automatization\_notebook\_04

2023-10-14

# Чтение данных

В вашем варианте нужно использовать датасет healthcare-dataset-stroke-data.

df <- read\_csv("data/raw/healthcare-dataset-stroke-data.csv")

# Выведите общее описание данных

glimpse(df) # структура данных

## Rows: 5,110  
## Columns: 12  
## $ id <dbl> 9046, 51676, 31112, 60182, 1665, 56669, 53882, 10434~  
## $ gender <chr> "Male", "Female", "Male", "Female", "Female", "Male"~  
## $ age <dbl> 67, 61, 80, 49, 79, 81, 74, 69, 59, 78, 81, 61, 54, ~  
## $ hypertension <dbl> 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1~  
## $ heart\_disease <dbl> 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0~  
## $ ever\_married <chr> "Yes", "Yes", "Yes", "Yes", "Yes", "Yes", "Yes", "No~  
## $ work\_type <chr> "Private", "Self-employed", "Private", "Private", "S~  
## $ Residence\_type <chr> "Urban", "Rural", "Rural", "Urban", "Rural", "Urban"~  
## $ avg\_glucose\_level <dbl> 228.69, 202.21, 105.92, 171.23, 174.12, 186.21, 70.0~  
## $ bmi <chr> "36.6", "N/A", "32.5", "34.4", "24", "29", "27.4", "~  
## $ smoking\_status <chr> "formerly smoked", "never smoked", "never smoked", "~  
## $ stroke <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~

# Очистка данных

1. Уберите переменные, в которых пропущенных значений больше 20% или уберите субъектов со слишком большим количеством пропущенных значений. Или совместите оба варианта. Напишите обоснование, почему вы выбрали тот или иной вариант:

**Обоснование**: в анализируемом наборе данных всего 2 переменных с пропущенными значениями, из них в переменной smoking\_status 30% пропущенных значений. Таким образом, мы можем убрать данную переменную и сохранить всех субъектов.

1. Переименуйте переменные в человекочитаемый вид (что делать с пробелами в названиях?);
2. В соответствии с описанием данных приведите переменные к нужному типу (numeric или factor);
3. Отсортируйте данные по возрасту по убыванию;
4. Сохраните в файл outliers.csv субъектов, которые являются выбросами (например, по правилу трёх сигм) — это необязательное задание со звёздочкой;
5. Присвойте получившийся датасет переменной “cleaned\_data”.

df <- df |>   
 # приведение переменных к нужному типу  
 mutate(across(where(is.character) & !bmi, \(x) as.factor(x))) |>   
 mutate(across(c(hypertension, heart\_disease, stroke), \(x) factor(recode(x,   
 "0" = "No",  
 "1" = "Yes")))) |>  
 mutate(bmi = as.numeric(if\_else(bmi == "N/A", NA, bmi))) |>  
 mutate(smoking\_status = if\_else(smoking\_status == "Unknown", NA, smoking\_status)) |>  
 mutate(smoking\_status = fct\_drop(smoking\_status)) |>   
 # Редактирование названий переменных  
 rename\_all(\(x)str\_to\_title(x)) |>   
 rename\_all(\(x) str\_replace\_all(x, "\_", " ")) |>   
 #сортировка по возрасту по убыванию  
 arrange(desc(Age))  
  
# работа с пропущенными значениями в данных  
df |>  
 select(where(\(x)sum(is.na(x))/length(x)>0.2)) |>   
 colnames()

## [1] "Smoking status"

# В переменной smoking\_status более 20% пропущенных значений  
sum(is.na(df$`Smoking status`))/nrow(df)

## [1] 0.3021526

cleaned\_data <- df |>   
 select(where(\(x)sum(is.na(x))/length(x)<=0.2))  
  
# работа с выбросами  
  
find\_outliers <- function(data, column) {  
 data |>   
 filter(if\_any({{ column }}, \(x) x > mean(x, na.rm = TRUE) + 3 \* sd(x, na.rm = TRUE) | x < mean(x, na.rm = TRUE) - 3 \* sd(x, na.rm = TRUE)))  
}  
  
outliers <- cleaned\_data |>   
 find\_outliers(where(is.numeric) & !Id)  
  
write\_csv(outliers, "outliers.csv") #Ссохранение в файл outliers.csv субъектов, которые являются выбросами  
  
cleaned\_data <- anti\_join(cleaned\_data, outliers, by = "Id")  
nrow(cleaned\_data)

## [1] 5003

summary(cleaned\_data)

## Id Gender Age Hypertension Heart disease  
## Min. : 67 Female:2926 Min. : 0.08 No :4534 No :4737   
## 1st Qu.:17757 Male :2076 1st Qu.:25.00 Yes: 469 Yes: 266   
## Median :36969 Other : 1 Median :44.00   
## Mean :36547 Mean :42.98   
## 3rd Qu.:54632 3rd Qu.:61.00   
## Max. :72940 Max. :82.00   
##   
## Ever married Work type Residence type Avg glucose level  
## No :1743 children : 687 Rural:2460 Min. : 55.12   
## Yes:3260 Govt\_job : 641 Urban:2543 1st Qu.: 77.08   
## Never\_worked : 22 Median : 91.58   
## Private :2856 Mean :104.53   
## Self-employed: 797 3rd Qu.:112.96   
## Max. :240.86   
##   
## Bmi Stroke   
## Min. :10.30 No :4764   
## 1st Qu.:23.50 Yes: 239   
## Median :27.90   
## Mean :28.51   
## 3rd Qu.:32.80   
## Max. :52.30   
## NA's :198

#Также мы видим одного человека, у которого указан пол Other, чтобы не было градации, состоящей из одного значения - удалим пациента  
  
cleaned\_data <- cleaned\_data |>   
 filter(!Gender == "Other") |>   
 mutate(Gender = fct\_drop(Gender))

# Сколько осталось переменных?

cleaned\_data |>   
 ncol()

## [1] 11

# Сколько осталось случаев?

cleaned\_data |>   
 nrow()

## [1] 5002

# Есть ли в данных идентичные строки?

# Посмотрим сначала, есть ли пациенты с одинаковым id в базе, если да, то тогда уже будем разибаться с ними подробнее (дубли, повторные визиты)  
if (nrow(distinct(cleaned\_data, Id) == nrow(cleaned\_data))) {  
 print("Дубликатов не обнаружено")  
} else {  
 print("Обнаружены дубликаты")  
}

## [1] "Дубликатов не обнаружено"

# Сколько всего переменных с пропущенными значениями в данных и сколько пропущенных точек в каждой такой переменной?

# Количество переменных с пропущенными значениями:  
cleaned\_data |>  
 select(where(\(x)sum(is.na(x))/length(x)>0)) |>   
 ncol()

## [1] 1

# Количество пропущенных значений в каждой такой переменной:  
cleaned\_data |>   
 select(where(\(x)sum(is.na(x))/length(x)>0)) |>  
 summarise(across(everything(), \(x)sum(is.na(x)), .names = "count\_{.col}")) |>   
 pivot\_longer(cols = everything(), names\_to = "variable", values\_to = "count")

## # A tibble: 1 x 2  
## variable count  
## <chr> <int>  
## 1 count\_Bmi 198

# Описательные статистики

## Количественные переменные

1. Рассчитайте для всех количественных переменных для каждой группы (stroke):

1.1) Количество значений;

1.2) Количество пропущенных значений;

1.3) Среднее;

1.4) Медиану;

1.5) Стандартное отклонение;

1.6) 25% квантиль и 75% квантиль;

1.7) Интерквартильный размах;

1.8) Минимум;

1.9) Максимум;

1.10) 95% ДИ для среднего - задание со звёздочкой.

statistics\_num <- list(  
 `Количество значений` = \(x) length(x) |> as.character(),  
 `Количество пропущенных значений` = \(x) sum(is.na(x)) |> as.character(),  
 `Среднее значение` = \(x) mean(x, na.rm = TRUE) |> round(2) |> as.character(),  
 `Медиана` = \(x) median(x, na.rm = TRUE) |> round(2) |> as.character(),  
 `Стандартное отклонение` = \(x) sd(x, na.rm = TRUE) |> round(2) |> as.character(),  
 `Q1 - Q3` = \(x) paste0(quantile(x, 0.25, na.rm = TRUE) |> round(2), " - ", quantile(x, 0.75, na.rm = TRUE) |> round(2)),  
 `Интерквартильный размах` = \(x) (quantile(x, 0.75, na.rm = TRUE) - quantile(x, 0.25, na.rm = TRUE)) |> round(2) |> as.character(),   
 `Мин. - макс.` = \(x) paste0(min(x, na.rm = TRUE) |> round(2), " - ", max(x, na.rm = TRUE) |> round(2)),  
 `95% ДИ для среднего` = \(x) paste0(round(mean(x, na.rm = TRUE) - qt(0.975, n()-1) \* sd(x, na.rm = TRUE)/sqrt(n()),2), " - ", round(mean(x, na.rm = TRUE) + qt(0.975, n()-1) \* sd(x, na.rm = TRUE)/sqrt(n()),2))  
)  
  
cleaned\_data |>   
 select(`Stroke`, where(is.numeric) & !Id) |>  
 group\_by(`Stroke`) |>  
 summarise(across(where(is.numeric), statistics\_num )) |>  
 pivot\_longer(!`Stroke`) |>  
 pivot\_wider(names\_from = `Stroke`, values\_from = value) |>   
 separate(name, into = c("Переменная", "Статистика"), sep = "\_") |>   
 flextable() |>   
 theme\_vanilla() |>   
 add\_header\_row(values = c(" ", "Stroke"), colwidths = c(2,2)) |>   
 merge\_v("Переменная") |>   
 align(align = "center", part = "all") |>  
 set\_table\_properties(width = 1, layout = "autofit")

|  | | **Stroke** | |
| --- | --- | --- | --- |
| **Переменная** | **Статистика** | **No** | **Yes** |
| Age | Количество значений | 4763 | 239 |
| Количество пропущенных значений | 0 | 0 |
| Среднее значение | 41.75 | 67.65 |
| Медиана | 43 | 71 |
| Стандартное отклонение | 22.34 | 12.79 |
| Q1 - Q3 | 24 - 59 | 59 - 78 |
| Интерквартильный размах | 35 | 19 |
| Мин. - макс. | 0.08 - 82 | 1.32 - 82 |
| 95% ДИ для среднего | 41.11 - 42.38 | 66.02 - 69.28 |
| Avg glucose level | Количество значений | 4763 | 239 |
| Количество пропущенных значений | 0 | 0 |
| Среднее значение | 103.37 | 127.61 |
| Медиана | 91.16 | 104.12 |
| Стандартное отклонение | 41.65 | 58.13 |
| Q1 - Q3 | 77 - 112.1 | 78.81 - 190.96 |
| Интерквартильный размах | 35.1 | 112.16 |
| Мин. - макс. | 55.12 - 240.86 | 56.11 - 240.59 |
| 95% ДИ для среднего | 102.19 - 104.55 | 120.2 - 135.02 |
| Bmi | Количество значений | 4763 | 239 |
| Количество пропущенных значений | 158 | 40 |
| Среднее значение | 28.44 | 30.13 |
| Медиана | 27.8 | 29.3 |
| Стандартное отклонение | 7.18 | 6.02 |
| Q1 - Q3 | 23.4 - 32.8 | 26.35 - 33.1 |
| Интерквартильный размах | 9.4 | 6.75 |
| Мин. - макс. | 10.3 - 52.3 | 16.9 - 48.9 |
| 95% ДИ для среднего | 28.24 - 28.65 | 29.36 - 30.9 |

## Категориальные переменные

1. Рассчитайте для всех категориальных переменных для каждой группы (stroke):

1.1) Абсолютное количество;

1.2) Относительное количество внутри группы;

1.3) 95% ДИ для доли внутри группы - задание со звёздочкой.

statistics\_cat <- function(var){  
 label <- rlang::englue("{{var}}")  
   
 cleaned\_data |>   
 select(`Stroke`, {{var}}) |>  
 count(`Stroke`, {{var}}) |>   
 group\_by(`Stroke`) |>  
 mutate(`% по группе` = percent((n / sum(n)), 1)) |>   
 ungroup() |>   
 mutate(n = as.character(n)) |>   
 pivot\_longer(!c(`Stroke`, {{var}})) |>   
 pivot\_wider(names\_from = `Stroke`, values\_from = value) |>   
 mutate(`Переменная` = label) |>   
 relocate(`Переменная`, .before = 1) |>   
 rename("Статистика" = name) |>   
 rename("Категория" = {{var}})   
}  
  
  
names\_new <- cleaned\_data |> select(where(is.factor)& !Stroke) |>names()  
  
data\_cat <- map\_dfr(.x = vars(Gender, Hypertension, `Heart disease`, `Ever married`, `Work type`, `Residence type`), .f = statistics\_cat)  
  
data\_cat |>   
 flextable() |>   
 theme\_vanilla() |>   
 add\_header\_row(values = c(" ", "Stroke"), colwidths = c(3,2)) |>  
 merge\_v(c("Переменная", "Категория")) |>   
 align(align = "center", part = "all") |>  
 set\_table\_properties(width = 1, layout = "autofit")

|  | | | **Stroke** | |
| --- | --- | --- | --- | --- |
| **Переменная** | **Категория** | **Статистика** | **No** | **Yes** |
| Gender | Female | n | 2788 | 138 |
| % по группе | 59% | 58% |
| Male | n | 1975 | 101 |
| % по группе | 41% | 42% |
| Hypertension | No | n | 4354 | 179 |
| % по группе | 91% | 75% |
| Yes | n | 409 | 60 |
| % по группе | 9% | 25% |
| Heart disease | No | n | 4540 | 196 |
| % по группе | 95% | 82% |
| Yes | n | 223 | 43 |
| % по группе | 5% | 18% |
| Ever married | No | n | 1713 | 29 |
| % по группе | 36% | 12% |
| Yes | n | 3050 | 210 |
| % по группе | 64% | 88% |
| Work type | children | n | 685 | 2 |
| % по группе | 14% | 1% |
| Govt\_job | n | 609 | 32 |
| % по группе | 13% | 13% |
| Never\_worked | n | 22 |  |
| % по группе | 0% |  |
| Private | n | 2714 | 141 |
| % по группе | 57% | 59% |
| Self-employed | n | 733 | 64 |
| % по группе | 15% | 27% |
| Residence type | Rural | n | 2351 | 108 |
| % по группе | 49% | 45% |
| Urban | n | 2412 | 131 |
| % по группе | 51% | 55% |

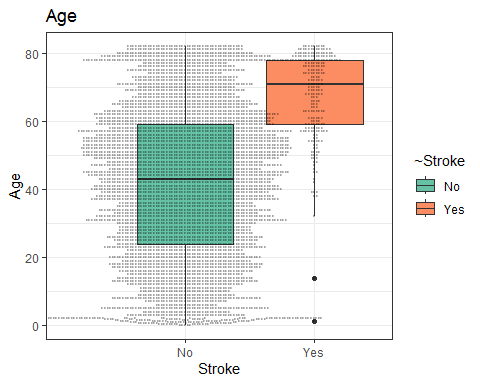
# Визуализация

## Количественные переменные

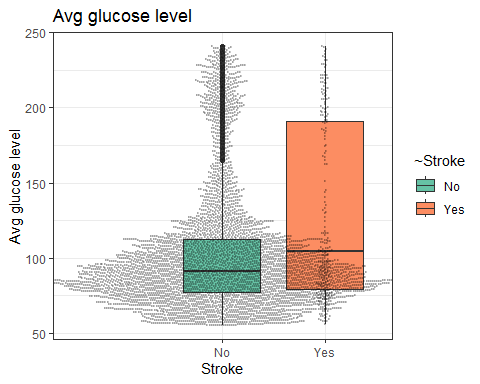
1. Для каждой количественной переменной сделайте боксплоты по группам. Расположите их либо на отдельных рисунках, либо на одном, но читаемо;
2. Наложите на боксплоты beeplots - задание со звёздочкой.
3. Раскрасьте боксплоты с помощью библиотеки RColorBrewer.

library(RColorBrewer)  
#display.brewer.all(colorblindFriendly = TRUE)  
  
  
graph\_num <- function(num\_var){  
  
 ggplot(data = cleaned\_data, aes(x = Stroke, y = {{num\_var}})) +  
 geom\_boxplot(aes(fill = Stroke)) +  
 ggbeeswarm::geom\_beeswarm(color = "Black", size = 0.1, alpha = 0.2) +  
 labs(title = rlang::englue("{{num\_var}}")) +  
 scale\_fill\_brewer(aes(x = Stroke, y = {{num\_var}}, fill = Stroke), palette="Set2") +  
 theme\_bw()  
}  
  
  
graph\_num\_list <- map(.x = vars(Age, `Avg glucose level`, Bmi), .f = graph\_num)  
  
graph\_num\_list

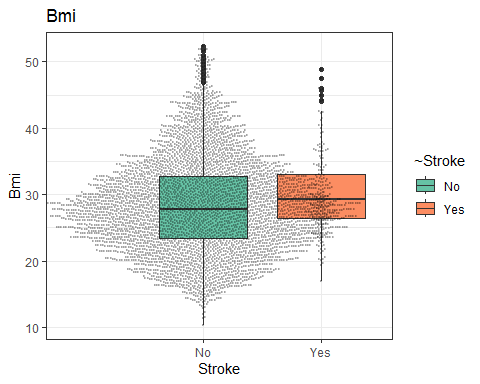
## [[1]]



##   
## [[2]]



##   
## [[3]]

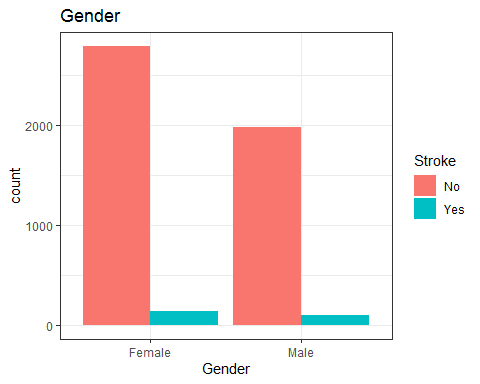


## Категориальные переменные

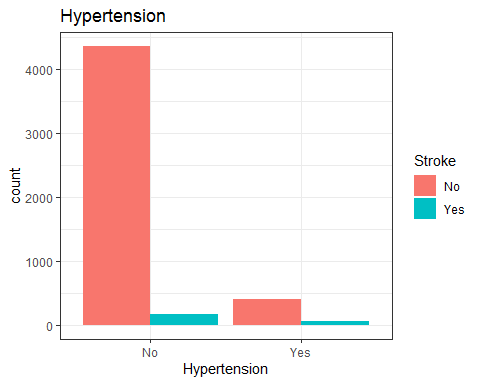
1. Сделайте подходящие визуализации категориальных переменных. Обоснуйте, почему выбрали именно этот тип.

#Для визуализации распределения частот номинальных переменных хорошо подходит гистограмма частот(барплот), построим ее.  
  
graph\_cat <- function(cat\_var){  
  
 ggplot(data = cleaned\_data, aes(x = {{cat\_var}}, fill = Stroke)) +  
 geom\_bar(position = "dodge") +  
 labs(title = rlang::englue("{{cat\_var}}")) +  
 theme\_bw()  
  
}  
  
  
graph\_cat\_list <- map(.x = vars(Gender, Hypertension, `Heart disease`, `Ever married`, `Work type`, `Residence type`), .f = graph\_cat)  
  
graph\_cat\_list

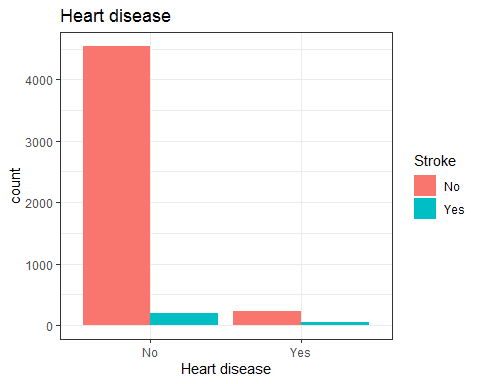
## [[1]]



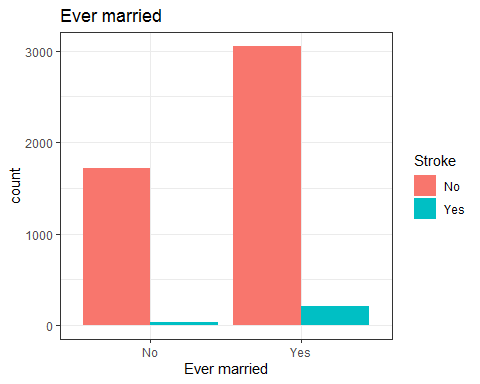
##   
## [[2]]



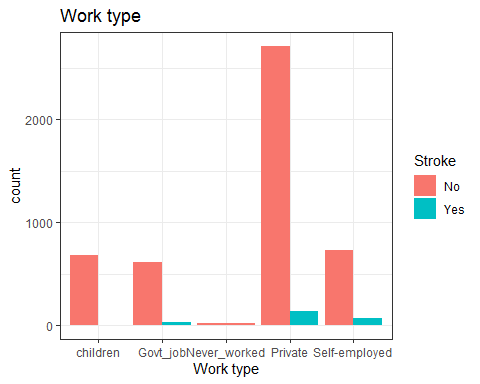
##   
## [[3]]



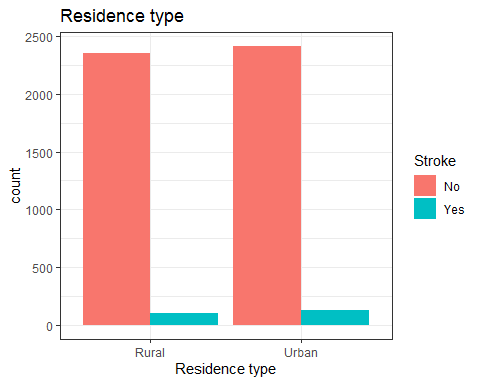
##   
## [[4]]



##   
## [[5]]



##   
## [[6]]



# Статистические оценки

## Проверка на нормальность

1. Оцените каждую переменную на соответствие нормальному распределению с помощью теста Шапиро-Уилка. Какие из переменных являются нормальными и как как вы это поняли?

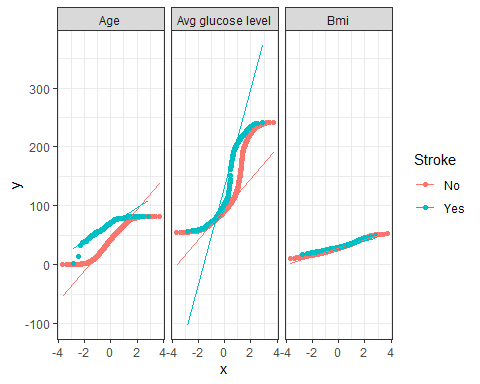
shapiro\_test <- function(x) {  
 shapiro.test(x) |>  
 tidy() |>  
 select(method, statistic, `p value` = p.value)  
}  
  
# Применение теста Шапиро-Уилка ко всем количественным переменным и объединение результатов в таблицу  
cleaned\_data |>  
 group\_by(Stroke) |>   
 select(where(is.numeric) & !Id) |>  
 pivot\_longer(everything() & !Stroke) |>  
 group\_by(Stroke, name) |>  
 summarise(shapiro\_results = list(shapiro\_test(value))) |>  
 arrange(name) |>   
 unnest(shapiro\_results)

## # A tibble: 6 x 5  
## # Groups: Stroke [2]  
## Stroke name method statistic `p value`  
## <fct> <chr> <chr> <dbl> <dbl>  
## 1 No Age Shapiro-Wilk normality test 0.970 2.35e-30  
## 2 Yes Age Shapiro-Wilk normality test 0.877 5.83e-13  
## 3 No Avg glucose level Shapiro-Wilk normality test 0.809 9.32e-60  
## 4 Yes Avg glucose level Shapiro-Wilk normality test 0.860 6.37e-14  
## 5 No Bmi Shapiro-Wilk normality test 0.984 3.35e-22  
## 6 Yes Bmi Shapiro-Wilk normality test 0.965 6.66e- 5

# По результатам теста распределения всех количественных переменных по переменной Stroke отличаются от нормального(р<0.001)

1. Постройте для каждой количественной переменной QQ-плот. Отличаются ли выводы от теста Шапиро-Уилка? Какой метод вы бы предпочли и почему?

cleaned\_data |>   
 group\_by(Stroke) |>   
 select( where(is.numeric) & !Id) |>   
 pivot\_longer(everything()& !Stroke) |>   
 ggplot(aes(sample = value, colour = Stroke)) +  
 stat\_qq() +  
 stat\_qq\_line() +  
 facet\_wrap(~name) +  
 theme\_bw()



# При построении графиков QQ-plot мы видим, что для переменной Bmi по градации Yes(Stroke) точки лежат близко к прямой линии, что указывает на нормальность распределения данных, для других переменных этого не наблюдается.  
# Выводы по результатам построения графика QQ-плот и теста Шапиро-Уилка схожи. Я бы предпочла тест Шапиро-Уилка, т к он позволяет формально оценить, соответствуют ли данные нормальному распределению на основе p-value

1. Ниже напишите, какие ещё методы проверки на нормальность вы знаете и какие у них есть ограничения.

**- Анализ моды, медианы, среднего значения, эксцесса, асимметрии.**

**- К графическим методам также относят построение гистограммы - график распределения данных с наложением кривой нормального распределения для визуальной оценки, анализ ящичковой диаграммы. Ограничение - визуальная оценка субъективна**

**- Тест Колмогорова-Смирнова. Ограничение - выборка должная быть достаточно большой**

## Сравнение групп

1. Сравните группы (переменная **stroke**) по каждой переменной (как количественной, так и категориальной). Для каждой переменной выберите нужный критерий и кратко обоснуйте его выбор в комментариях.

# Применение теста Манна-Уитни для каждой количественной переменной по stroke - выборки независимы, распределение данных на основании теста Шапиро-Уилка отличается от нормального.  
  
wilcox\_test <- function(x,y) {  
 wilcox.test(x~y) |>  
 tidy() |>  
 select(method, statistic, `p value` = p.value)  
}  
  
cleaned\_data |>  
 group\_by(Stroke) |>   
 select(where(is.numeric) & !Id, Stroke) |>  
 pivot\_longer(everything() & !Stroke) |>  
 group\_by(name) |>  
 summarise(wilcox\_results = list(wilcox\_test(value,Stroke))) |>  
 unnest(wilcox\_results) |>   
 flextable() |>   
 colformat\_double(digits = 3) |>   
 theme\_vanilla() |>   
 align(align = "center", part = "all") |>  
 set\_table\_properties(width = 1, layout = "autofit")

| **name** | **method** | **statistic** | **p value** |
| --- | --- | --- | --- |
| Age | Wilcoxon rank sum test with continuity correction | 187,867.500 | 0.000 |
| Avg glucose level | Wilcoxon rank sum test with continuity correction | 454,961.500 | 0.000 |
| Bmi | Wilcoxon rank sum test with continuity correction | 388,159.500 | 0.000 |

# Применение хи-квадрат критерия для каждой категориальной переменной  
  
# По переменной Work type наблюдается разреженная таблица сопряженности, объединим 2 редко встречающиеся градации в одну   
cleaned\_data <- cleaned\_data |>  
 mutate(`Work type` = recode(`Work type`, "children" = "children\_never\_worked", "Never\_worked" = "children\_never\_worked"))  
  
chisq\_test <- function(x,y) {  
 chisq.test(x,y) |>  
 tidy() |>  
 select(method, statistic, `p value` = p.value)  
}  
  
cleaned\_data |>  
 group\_by(Stroke) |>   
 select(where(is.factor)) |>  
 pivot\_longer(everything() & !Stroke) |>  
 group\_by(name) |>  
 summarise(chisq\_results = list(chisq\_test(value, Stroke))) |>  
 unnest(chisq\_results) |>   
 flextable() |>   
 colformat\_double(digits = 3) |>   
 theme\_vanilla() |>   
 align(align = "center", part = "all") |>  
 set\_table\_properties(width = 1, layout = "autofit")

| **name** | **method** | **statistic** | **p value** |
| --- | --- | --- | --- |
| Ever married | Pearson's Chi-squared test with Yates' continuity correction | 55.897 | 0.000 |
| Gender | Pearson's Chi-squared test with Yates' continuity correction | 0.031 | 0.860 |
| Heart disease | Pearson's Chi-squared test with Yates' continuity correction | 77.448 | 0.000 |
| Hypertension | Pearson's Chi-squared test with Yates' continuity correction | 71.142 | 0.000 |
| Residence type | Pearson's Chi-squared test with Yates' continuity correction | 1.422 | 0.233 |
| Work type | Pearson's Chi-squared test | 50.252 | 0.000 |

# Далее идут **необязательные** дополнительные задания, которые могут принести вам дополнительные баллы в том числе в случае ошибок в предыдущих

## Корреляционный анализ

1. Создайте корреляционную матрицу с визуализацией и поправкой на множественные сравнения. Объясните, когда лучше использовать корреляционные матрицы и в чём минусы и плюсы корреляционных исследований.

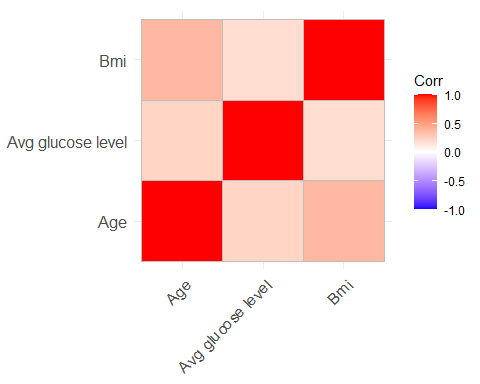
**Корреляционные матрицы помогают быстро оценить степень взаимосвязи между парами переменных, направление связи. Важно смотреть корреляционные матрицы при построении моделей для исключения мультиколлинеарности.**

**Из минусов можно выделить то, что с помощью такой корреляции мы можем оценить только линейные связи между переменными. Также мы не можем оценить причинно-следственные связи.**

# Создание корреляционной матрицы для количественных переменных.  
cleaned\_data\_num <- cleaned\_data |>  
 select( where(is.numeric) & !Id)  
  
psych::corr.test(cleaned\_data\_num,  
 adjust = "holm")

## Call:psych::corr.test(x = cleaned\_data\_num, adjust = "holm")  
## Correlation matrix   
## Age Avg glucose level Bmi  
## Age 1.00 0.22 0.37  
## Avg glucose level 0.22 1.00 0.18  
## Bmi 0.37 0.18 1.00  
## Sample Size   
## Age Avg glucose level Bmi  
## Age 5002 5002 4804  
## Avg glucose level 5002 5002 4804  
## Bmi 4804 4804 4804  
## Probability values (Entries above the diagonal are adjusted for multiple tests.)   
## Age Avg glucose level Bmi  
## Age 0 0 0  
## Avg glucose level 0 0 0  
## Bmi 0 0 0  
##   
## To see confidence intervals of the correlations, print with the short=FALSE option

# Визуализация  
ggcorrplot(cor(cleaned\_data\_num|>  
 drop\_na()))



## Моделирование

1. Постройте регрессионную модель для переменной **stroke**. Опишите процесс построения