Розеттский камень

Пуассон, фея и три мексиканских негодяя

2019-09-23

Оглавление

Глава 1

Напутственное слово

Глава 2

Коан об установке софта

В этом коане мы рассмотрим установку и настройку программ для работы на языках программирования R и Python, а также установку и настройку программы Stata.

###Язык программирования R > R - это открытая среда программирования, помогающая в работе со статистическими данными. Для программирования на R подойдет программа RStudio.

Рассмотрим установку RStudio на Mac OS и Windows.

#####Инструкция по установке RStudio для Windows / Mac OS:

1. Загрузите и установите язык программирования R с официального сайта.

   - Версия для Windows: Выберите “Download R for Windows” ▸ “base” ▸ “Download R 3.x.x for Windows”.

   - Версия для Mac OS: Выберите “Download R for (Mac) OS X” ▸ “Latest Release” ▸ “R 3.x.x”.

2. Загрузите программу RStudio с официального сайта разработчика (выберите подходящую версию из предложенных опций). Возможностей бесплатной версии будет вполне достаточно для работы.

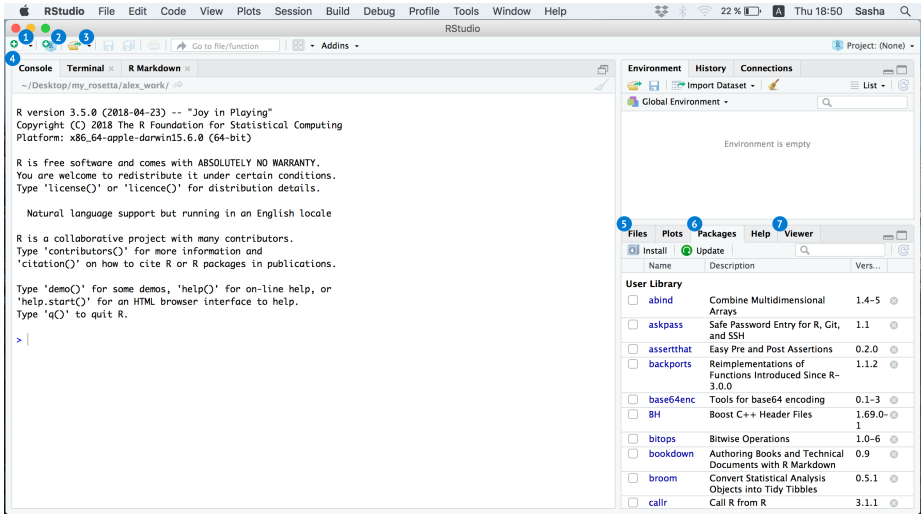
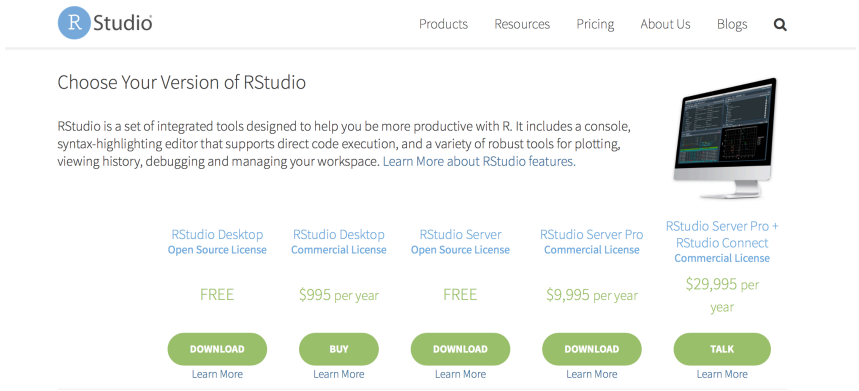


Рис. 2.1: *Интерфейс программы*



Готово, Вы можете использовать RStudio на вашем компьютере.

#####Начало работы

1. **New file** - Создание нового файла.
2. **New project** - Создание нового проекта.
3. **Open file** - Открытие существующего файла.
4. **Console** - Консоль, в которой набирается код.
5. **Files** - Список файлов, доступных для работы.
6. **Packages** - Список установленных пакетов, т.е. расширений. Также можно ознакомиться с ним, введя в консоль команду *installed.packages()*.

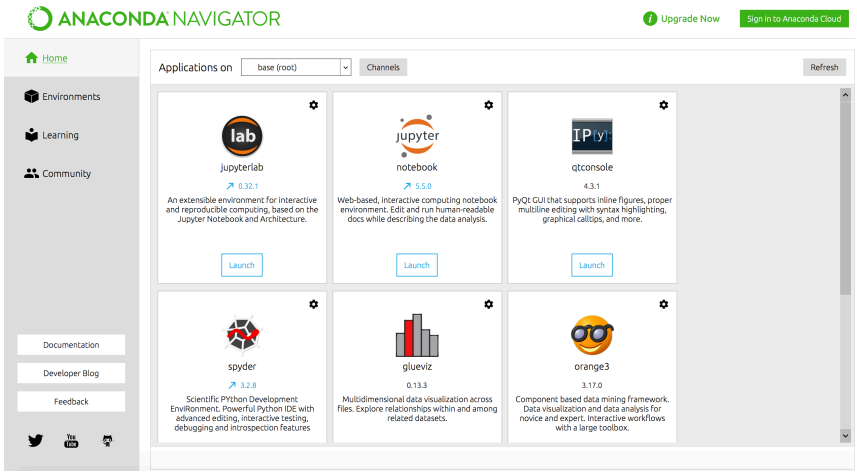
7. **Viewer** - Отображение введенного кода.

###Язык программирования Python > Python - это ещё одна открытая среда программирования, помогающая в работе со статистическими данными. Для программирования на Python подойдет программа Jupyter Notebook.

#####Установка

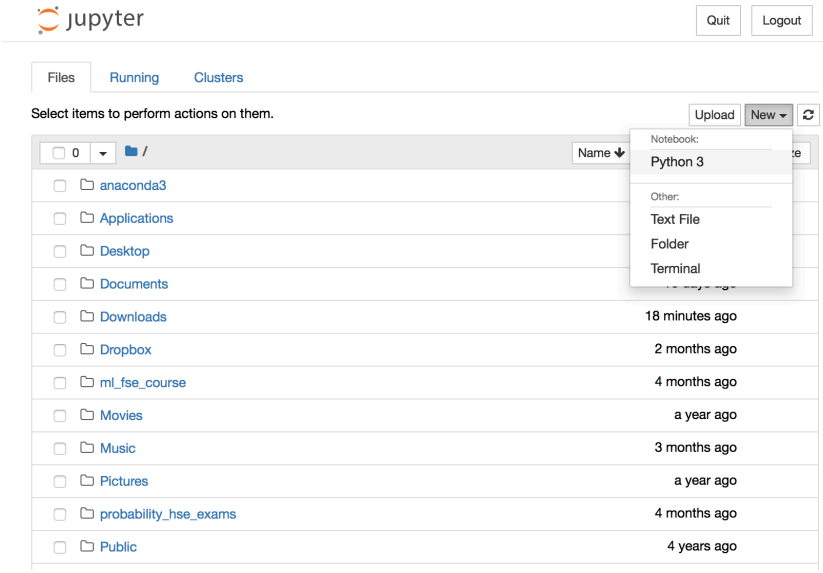
1. Загрузите и установите Anaconda с официального сайта.

2. После загрузки и установки откройте Anaconda Navigator, через который Вы сможете открыть программу Jupyter Notebook. Navigator.bb



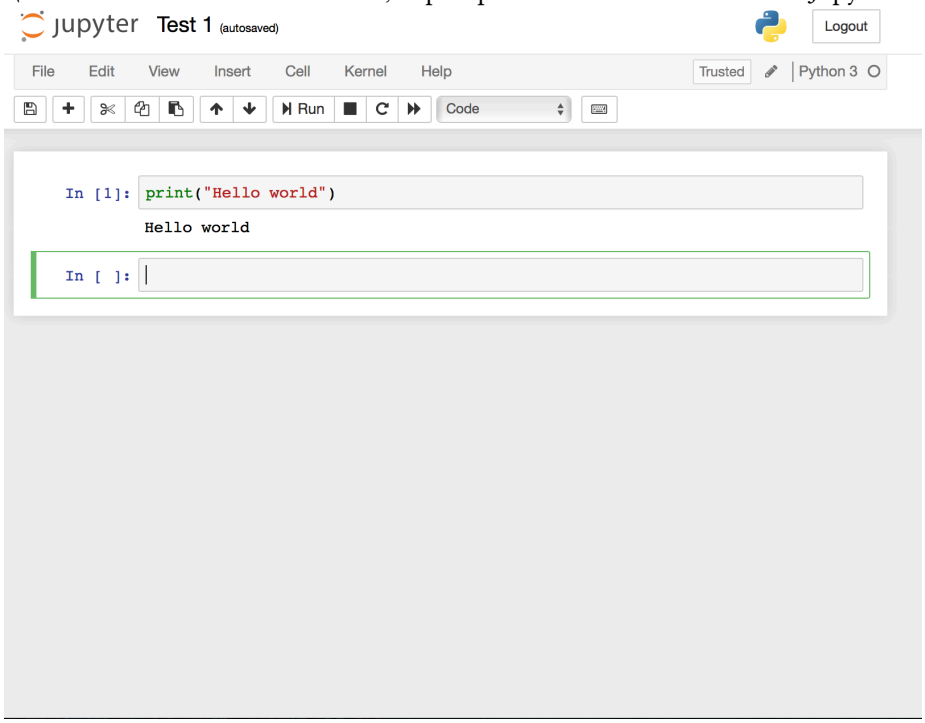
#####Начало работы

Открыв Jupyter Notebook, вы попадете на страницу, содержащую ваши сохраненные файлы. Чтобы создать новый файл, нажмите “New” ▸ “Notebook: Python



3". File in Jupyter.bb

Затем, в открывшемся окне, появится новый файл. Теперь все готово к работе. Вы можете вводить свой код и затем, используя комбинацию клавиш "Shift" + "Enter", проверять его исполнение. in Jupyter.bb



Interface.bb

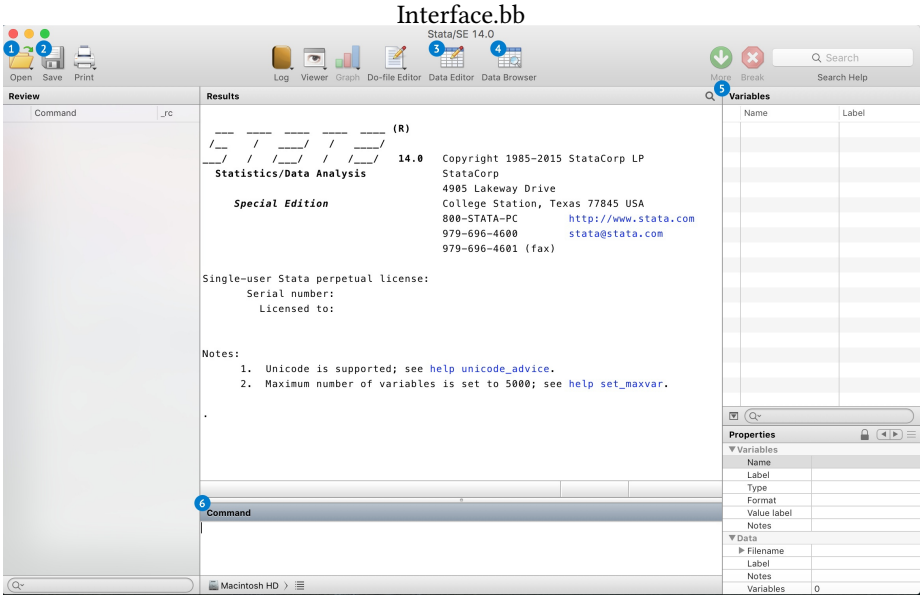


Рис. 2.2: *Интерфейс Stata*

###Программа STATA > Stata, в отличие от R и Python, является программой, а не языком программирования. Она также помогает в работе со статистическими данными.

#####Установка:

Для установки Stata необходимо загрузить актуальную версию с сайта компании-разработчика. Подойдут как Stata SE, так и Stata MP.

#####Начало работы:

1. **Open File** - открыть файл.
2. **Save** - сохранить файл.
3. **Data Editor** - редактирование данных.
4. **Data Browser** - просмотр данных.
5. **Variables** - список переменных.
6. **Command** - командная строка, в которой вводится код.

Глава 3

Коан о простой линейной регрессии

3.1. r

Построим простую линейную регрессию в R и проведем несложные тесты.

Загрузим необходимые пакеты.

```
library(tidyverse) # для манипуляций с данными и построения графиков
library(skimr) # для красивого summary
library(rio) # для чтения .dta файлов
library(car) # для линейных гипотез
library(tseries) # для теста на нормальность
library(sjPlot) # еще графики
```

Импортируем данные.

```
df = rio::import("data/us-return.dta")
```

Исследуем наш датасет.

```
skim_with(numeric = list(hist = NULL, p25 = NULL, p75 = NULL)) # опустим некоторые описательные статистики
skim(df)
```

```
Skim summary statistics
 n obs: 2664
 n variables: 22

-- Variable type:character -----------------------------------------------------------
 variable missing complete    n min max empty n_unique
        B       0     2664 2664   0   6  2544       31
```

```
-- Variable type:numeric ------------------------------------------------------------------
 variable missing complete    n    mean      sd      p0     p50    p100
        A    2544      120 2664 60.5    34.79    1      60.5    120
    BOISE    2544      120 2664  0.017   0.097  -0.27    0.015    0.38
   CITCRP    2544      120 2664  0.012   0.081  -0.28    0.011    0.32
    CONED    2544      120 2664  0.019   0.05   -0.14    0.019    0.15
   CONTIL    2544      120 2664 -0.0011  0.15   -0.6     0        0.97
   DATGEN    2544      120 2664  0.0075  0.13   -0.34    0.017    0.53
      DEC    2544      120 2664  0.02    0.099  -0.36    0.024    0.39
    DELTA    2544      120 2664  0.012   0.096  -0.26    0.013    0.29
   GENMIL    2544      120 2664  0.017   0.065  -0.15    0.011    0.19
   GERBER    2544      120 2664  0.016   0.088  -0.29    0.015    0.23
      IBM    2544      120 2664  0.0096  0.059  -0.19    0.002    0.15
   MARKET    2544      120 2664  0.014   0.068  -0.26    0.012    0.15
    MOBIL    2544      120 2664  0.016   0.08   -0.18    0.013    0.37
    MOTOR    2544      120 2664  0.018   0.097  -0.33    0.017    0.27
    PANAM    2544      120 2664  0.0035  0.13   -0.31    0        0.41
     PSNH    2544      120 2664 -0.0042  0.11   -0.48    0        0.32
   rkfree    2544      120 2664  0.0068  0.0022  0.0021  0.0066   0.013
   RKFREE    2544      120 2664  0.0068  0.0022  0.0021  0.0066   0.013
    TANDY    2544      120 2664  0.025   0.13   -0.25    0.022    0.45
   TEXACO    2544      120 2664  0.012   0.08   -0.19    0.01     0.4
    WEYER    2544      120 2664  0.0096  0.085  -0.27   -0.002    0.27
```

Переименуем столбцы.

```
df = rename(df, n = A, date = B)
```

```
df = na.omit(df) # уберем пустые строки
```

Будем верить в CAPM :) Оценим параметры модели для компании MOTOR. Соответственно, зависимая переменная - разница доходностей акций MOTOR и безрискового актива, а регрессор - рыночная премия.

```
df = mutate(df, y = MOTOR - RKFREE, x = MARKET - RKFREE)
```

Строим нашу модель и проверяем гипотезу об адекватности регрессии.

```
ols = lm(y ~ x, data = df)
summary(ols)
```

```
Call:
lm(formula = y ~ x, data = df)

Residuals:
     Min       1Q   Median       3Q      Max
```

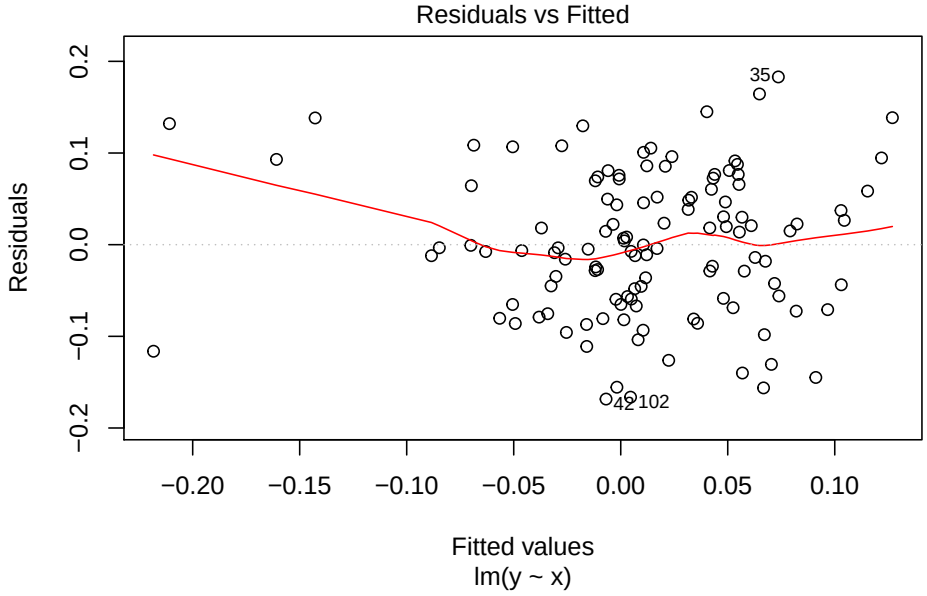
```
-0.168421 -0.059381 -0.003399  0.061373  0.182991

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.005253   0.007200   0.730    0.467
x           0.848150   0.104814   8.092 5.91e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.07844 on 118 degrees of freedom
Multiple R-squared:  0.3569,    Adjusted R-squared:  0.3514
F-statistic: 65.48 on 1 and 118 DF,  p-value: 5.913e-13
```

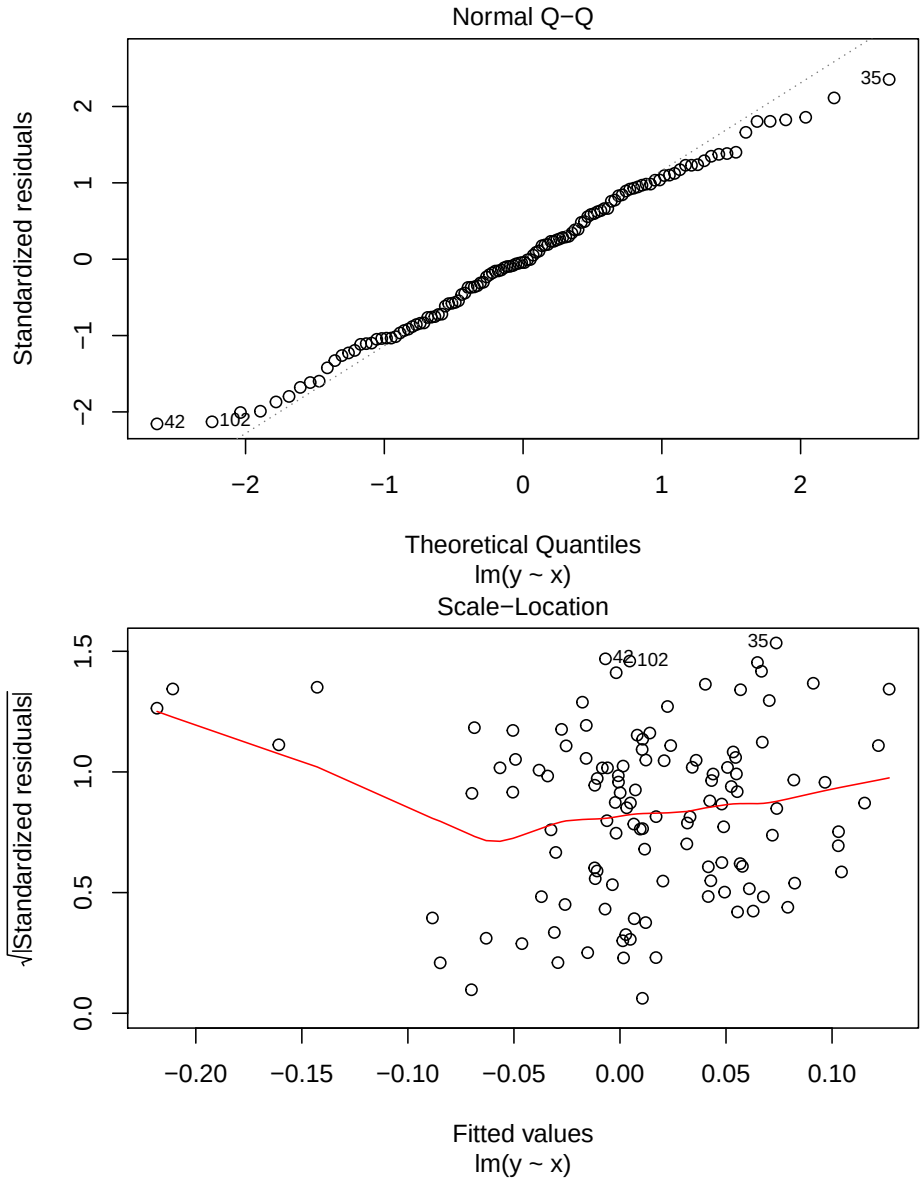
```
coeff = summary(ols)$coeff # отдельно табличка с коэффициентами
coeff
```

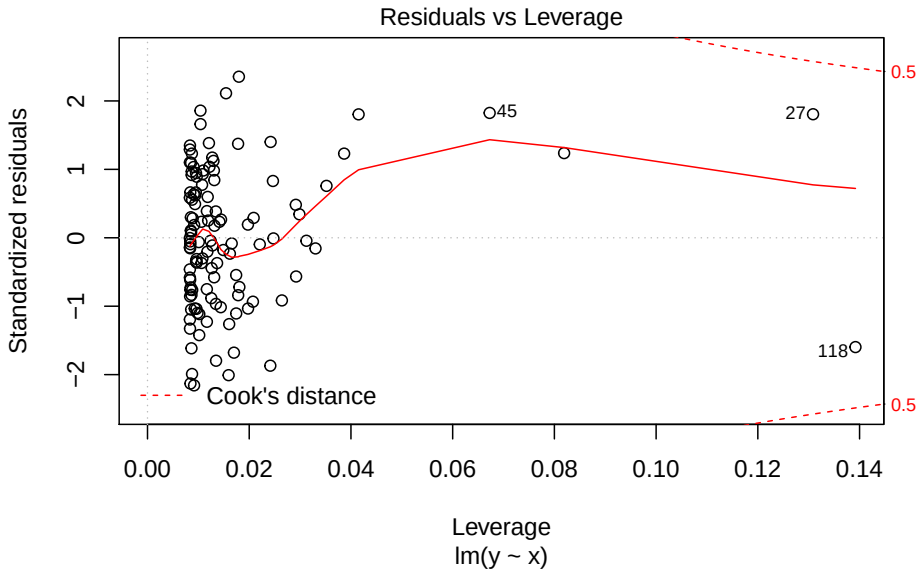
```
              Estimate  Std. Error    t value     Pr(>|t|)
(Intercept) 0.005252865 0.007199935 0.7295713 4.670981e-01
x           0.848149581 0.104813757 8.0919681 5.913330e-13
```

Вызовом одной функции получаем кучу полезных графиков. Можем визуально оценить наличие гетероскедастичности, нормальность распределения остатков, наличие выбросов.

```
plot(ols)
```




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(y ~ x)
Scale−Location
√|Standardized residuals|
Fitted values
lm(y ~ x)
35
42
102



Строим доверительный интервал для параметров модели.

```
est = cbind(Estimate = coef(ols), confint(ols))
```

Проверим гипотезу о равенстве коэффициента при регрессоре единице.

```
linearHypothesis(ols, c("x = 1"))
```

Linear hypothesis test

Hypothesis:
x = 1

Model 1: restricted model
Model 2: y ~ x

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1	119	0.73900				
2	118	0.72608	1	0.012915	2.0989	0.1501

Посмотрим на остатки :) Протестируем остатки регрессии на нормальность с помощью теста Харке-Бера.

$H_0 : S = 0, K = 3$, где S — коэффициент асимметрии (Skewness), K — коэффициент эксцесса (Kurtosis)

```
jarque.bera.test(resid(ols))
```

Jarque Bera Test

data: resid(ols)
X-squared = 1.7803, df = 2, p-value = 0.4106

И тест Шапиро-Уилка.

$H_0 : \epsilon_i \sim N(\mu, \sigma^2)$

```
shapiro.test(resid(ols))
```

Shapiro-Wilk normality test

data: resid(ols)
W = 0.99021, p-value = 0.5531

Оба теста указывают на нормальность распределения остатков регрессии.

Сделаем прогноз модели по данным вне обучаемой выборки.

```
set.seed(7)

newData = data.frame(x = df$x + 0.5*rnorm(length(df$x))) #пошумим
yhat = predict(ols, newdata = newData, se = TRUE)
```

3.2. python

Много полезных функций для статистических расчетов можно найти в пакете Statsmodels.

```
import pandas as pd # для работы с таблицами
```

Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'pandas'

Detailed traceback:
File "<string>", line 1, in <module>

```
import numpy as np # математика, работа с матрицами
import matplotlib.pyplot as plt # графики
```

Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'matplotlib

Detailed traceback:
File "<string>", line 1, in <module>

```
import statsmodels.api as sm
```

Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'statsmode

Detailed traceback:
File ”<string>”, line 1, in <module>

```
import statsmodels.formula.api as smf
```

Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'statsmodels'

Detailed traceback:
File ”<string>”, line 1, in <module>

```
import statsmodels.graphics.gofplots as gf
```

Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'statsmodels'

Detailed traceback:
File ”<string>”, line 1, in <module>

```
from statsmodels.stats.outliers_influence import summary_table
```

Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'statsmodels'

Detailed traceback:
File ”<string>”, line 1, in <module>

```
import seaborn as sns # еще более классные графики
```

Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'seaborn'

Detailed traceback:
File ”<string>”, line 1, in <module>

```
from scipy.stats import shapiro # еще математика
```

Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'scipy'

Detailed traceback:
File ”<string>”, line 1, in <module>

```
import statsmodels.discrete.discrete_model
```

Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'statsmodels'

Detailed traceback:
File ”<string>”, line 1, in <module>

При желании, можем кастомизировать графики :)

```
plt.style.use('seaborn')
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

```
plt.rc('font', size=14)
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

```
plt.rc('figure', titlesize=15)
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

```
plt.rc('axes', labelsize=15)
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

```
plt.rc('axes', titlesize=15)
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

Загрузим данные.

```
df = pd.read_stata('data/us-return.dta')
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'pd' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

Избавимся от наблюдений с пропущенными значениями.

```
df.dropna(inplace=True)
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'df' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

```
df.reset_index(drop=True, inplace=True)
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'df' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

Переименуем столбцы.

```
df = df.rename(columns={'A':'n', 'B': 'date'})
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'df' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
df['y'] = df['MOTOR'] - df['RKFREE']
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'df' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
df['x'] = df['MARKET'] - df['RKFREE']
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'df' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

Строим модель и читаем саммари :)

```
regr = smf.ols('y~x', data = df).fit()
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'smf' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
regr.summary()
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'regr' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

Получить прогноз.

```
df['yhat'] = regr.fittedvalues
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'regr' is not defined
```

Detailed traceback:
File ”<string>”, line 1, in <module>

Красивые графики для остатков, выборосов и прочих радостей, как в R, придется строить ручками. Зато приятно поиграть с оформлением :)

```
fig, ax = plt.subplots()
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

```
ax.plot(df['x'],regr.fittedvalues, color='g', alpha =0.8)
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'ax' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

```
ax.scatter(df['x'],regr.fittedvalues+regr.resid, color = 'g', alpha = 0.8, s = 40)
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'ax' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

```
ax.vlines(df['x'],regr.fittedvalues,regr.fittedvalues+regr.resid, color = 'gray', alpha = 0.5)
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'ax' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

```
plt.title('Линия регрессии и остатки')
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

```
plt.xlabel('RKFREE')
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

```
plt.ylabel('MARKET')
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
plt.show()
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

Строим доверительный интервал.

```
regr.conf_int()
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'regr' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

И проведем F-test.

```
hypotheses = '(x = 1)'
regr.f_test(r_matrix = hypotheses)
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'regr' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

Тест Шапиро. Такой же, как и в R. Для удобства можно поместить в табличку.

```
W, p_value = shapiro(regr.resid)
#pd.DataFrame(data = {'W': [round(W,3)], 'p_value': [round(p_value,3)]})
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'shapiro' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

Генерируем новые данные и строим предсказание.

```
import random
random.seed(7)

newData = df['x'] + 0.5*np.random.normal(len(df))
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'df' is not defined

Detailed traceback:
```

File ”<string>”, line 1, in <module>

```
prediction = regr.predict(newData)
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'regr' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

А теперь жесть! Построим графички, похожие на autoplot R.

```
fig_1 = plt.figure(1)
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

```
fig_1.axes[0] = sns.residplot(df['x'], df['y'],
                              lowess=True,
                              scatter_kws={'alpha': 0.6},
                              line_kws={'color': 'red', 'lw': 2, 'alpha': 0.8})
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'sns' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

```
fig_1.axes[0].set_title('Residuals vs Fitted')
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'fig_1' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

```
fig_1.axes[0].set_xlabel('Fitted values')
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'fig_1' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

```
fig_1.axes[0].set_ylabel('Residuals')


# можем добавить метки потенциальных аутлаеров
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'fig_1' is not defined

Detailed traceback:

File ”<string>”, line 1, in <module>

```
abs_resid = abs(regr.resid).sort_values(ascending=False)
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'regr' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

```
abs_resid_top3 = abs_resid[:3]
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'abs_resid' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

```
for i in abs_resid_top3.index:
    fig_1.axes[0].annotate(i,
                           xy=(regr.fittedvalues[i],
                               regr.resid[i]))
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'abs_resid_top3' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

```
norm_residuals = regr.get_influence().resid_studentized_internal # сохраним стьюдентизированные остатки
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'regr' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

```
QQ = gf.ProbPlot(norm_residuals)
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'gf' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

```
fig_2 = QQ.qqplot(line='45', alpha=0.5, color='b', lw=1)
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'QQ' is not defined

Detailed traceback:
File ”<string>”, line 1, in <module>

```
fig_2.axes[0].set_title('Normal Q-Q')
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'fig_2' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

```
fig_2.axes[0].set_xlabel('Theoretical Quantiles')
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'fig_2' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

```
fig_2.axes[0].set_ylabel('Standardized Residuals');

#и снова метки
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'fig_2' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

```
abs_norm_resid = np.flip(np.argsort(abs(norm_residuals)), 0)
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'norm_residuals' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

```
abs_norm_resid_top3 = abs_norm_resid[:3]
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'abs_norm_resid' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

```
for r, i in enumerate(abs_norm_resid_top3):
    fig_2.axes[0].annotate(i,
                           xy=(np.flip(QQ.theoretical_quantiles, 0)[r],
                               norm_residuals[i]))
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'abs_norm_resid_top3' is not define

Detailed traceback:
  File "<string>", line 1, in <module>

```
fig_3 = plt.figure(3)
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

```
plt.scatter(regr.fittedvalues, np.sqrt(abs(norm_residuals)), alpha=0.5)
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
sns.regplot(regr.fittedvalues, np.sqrt(abs(norm_residuals)),
            scatter=False,
            ci=False,
            lowess=True,
            line_kws={'color': 'red', 'lw': 1, 'alpha': 0.6})
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'sns' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
fig_3.axes[0].set_title('Scale-Location')
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'fig_3' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
fig_3.axes[0].set_xlabel('Fitted values')
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'fig_3' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
fig_3.axes[0].set_ylabel('$\sqrt{|Standardized Residuals|}$')

# и еще раз!)
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'fig_3' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
abs_sq_norm_resid = np.flip(np.argsort(np.sqrt(abs(norm_residuals)), 0))
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'norm_residuals' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
abs_sq_norm_resid_top3 = abs_sq_norm_resid[:3]
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'abs_sq_norm_resid' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
for i in abs_sq_norm_resid_top3:
    fig_3.axes[0].annotate(i, xy=(regr.fittedvalues[i],
                                  np.sqrt(abs(norm_residuals)[i])))
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'abs_sq_norm_resid_top3' is not def

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
leverage = regr.get_influence().hat_matrix_diag # сохраняем элементы матрицы-шляпницы
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'regr' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
cook_dist = regr.get_influence().cooks_distance[0] # и расстояние Кука
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'regr' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
fig_4 = plt.figure(4)
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
plt.scatter(leverage, norm_residuals, alpha=0.5)
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
sns.regplot(leverage, norm_residuals,
            scatter=False,
            ci=False,
            lowess=True,
            line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8})
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'sns' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

```
fig_4.axes[0].set_xlim(0, 0.20)
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'fig_4' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

```
fig_4.axes[0].set_ylim(-3, 5)
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'fig_4' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

```
fig_4.axes[0].set_title('Residuals vs Leverage')
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'fig_4' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

```
fig_4.axes[0].set_xlabel('Leverage')
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'fig_4' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

```
fig_4.axes[0].set_ylabel('Standardized Residuals')
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'fig_4' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

```
leverage_top3 = np.flip(np.argsort(cook_dist), 0)[:3]
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'cook_dist' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

```
for i in leverage_top3:
    fig_4.axes[0].annotate(i,
                           xy=(leverage[i],
```

```
                                norm_residuals[i]))
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'leverage_top3' is not defined

Detailed traceback:
  File ”<string>”, line 1, in <module>

```
plt.show()
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
  File ”<string>”, line 1, in <module>

3.3. stata

Загружаем данные.

use data/us-return.dta

Любуемся и даем новые названия столбцам.

```stata
summarize
ren A n
ren B date
```

```
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
           A |       120        60.5    34.78505         1        120
           B |         0
       MOBIL |       120    .0161917    .0803075      -.178       .366
      TEXACO |       120    .0119417    .0797036      -.194       .399
         IBM |       120    .0096167     .059024      -.187        .15
-------------+--------------------------------------------------------
         DEC |       120     .01975    .0991438      -.364       .385
      DATGEN |       120    .0074833    .1275399      -.342       .528
      CONED |       120    .0185083    .0502719      -.139       .151
        PSNH |       120   -.0042167    .1094712      -.485       .318
       WEYER |       120    .0096333    .0850664      -.271        .27
-------------+--------------------------------------------------------
       BOISE |       120     .016675    .0974882      -.274       .379
```

```
      MOTOR |       120    .0181583    .0972656      -.331        .27
      TANDY |       120    .0250083     .127566      -.246       .454
      PANAM |       120    .0035167    .1318054      -.313       .406
      DELTA |       120    .0116917    .0959317       -.26       .289
------------+---------------------------------------------------------
     CONTIL |       120     -.0011    .1506992       -.6       .974
      CITCRP |       120    .0118583    .0809719      -.282       .318
     GERBER |       120       .0164    .0877379      -.288       .234
     GENMIL |       120    .0165833    .0650403      -.148        .19
     MARKET |       120    .0139917    .0683532       -.26       .148
------------+---------------------------------------------------------
     RKFREE |       120    .0068386    .0021869     .00207     .01255
     rkfree |       120    .0068386    .0021869     .00207     .01255

```

Убираем пропущенные значения и создаем новые переменные.

```stata
drop if n == .
gen y = MOTOR - RKFREE
gen x = MARKET - RKFREE
```

```
(2,544 observations deleted)

```

Строим модель и проверяем гипотезу об адекватности регрессии. Тут же получаем доверительные интервалы для ко

```stata
reg y x
```

```
      Source |       SS           df       MS      Number of obs   =       120
-------------+----------------------------------   F(1, 118)       =     65.48
       Model |  .402913404         1  .402913404   Prob > F        =    0.0000
    Residual |  .726081541       118  .006153233   R-squared       =    0.3569
-------------+----------------------------------   Adj R-squared   =    0.3514
       Total |  1.12899494       119  .009487352   Root MSE        =    .07844

------------------------------------------------------------------------------
           y |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
```

```
           x |   .8481496   .1048138     8.09   0.000     .6405898    1.055709
       _cons |   .0052529   .0071999     0.73   0.467     -.009005    .0195107
------------------------------------------------------------------------------
```

Проверим гипотезу о равенстве коэффициента при регрессоре единице.

```stata
test x = 1
```

```
 ( 1)  x = 1

       F(  1,   118) =    2.10
            Prob > F =    0.1501
```

Сделаем предсказание по выборке и сохраним остатки.

```stata
predict u_hat, resid
predict y_hat
```

```
(option xb assumed; fitted values)
```

Протестируем остатки регрессии на нормальность с помощью теста Харке-Бера.
На самом деле, это не совсем тест Харке-Бера. Оригинальный вариант ассимптотический и в нем нет пог

```stata
sktest u_hat
```

```
                    Skewness/Kurtosis tests for Normality
                                                  ------ joint ------
    Variable |       Obs  Pr(Skewness)  Pr(Kurtosis) adj chi2(2)   Prob>chi2
-------------+-------------------------------------------------------------
       u_hat |       120     0.8841        0.1027        2.74         0.2539
```

И тест Шапиро-Уилка. Тут все аналогично R.

```stata
swilk u_hat
```

```
                Shapiro-Wilk W test for normal data

    Variable |       Obs       W           V       z       Prob>z
-------------+--------------------------------------------------------
       u_hat |       120    0.99021       0.942   -0.133   0.55310
```

Гипотеза о нормальности остатков не отвергается.

QQ - график

```stata
qnorm u_hat
```


График предсказанных значений против остатков.

```stata
rvfplot, yline(0)
```


График диагональных элементов матрицы-шляпницы против квадрата остатков (по сравнению с R оси поменялись м

```stata
lvr2plot
```


График предсказанных значений против стандартизиованных остатков. Размер точек на графике зависит от расстоя

```stata
predict D, cooksd
predict standard, rstandard

graph twoway scatter standard y_hat [aweight=D], msymbol(oh) yline(0)
```

```stata
set seed 7

set obs 120
gen x_new = x+ 0.5 * rnormal()
gen y_hat_new =  .8481496 * x_new + .0052529
```

```
 translator Graph2png not found
r(111);



number of observations (_N) was 120, now 120

```

<!--chapter:end:02-simplereg.Rmd-->

Модель бинарного выбора {#binchoice}

> Сейчас попробуем подружиться с моделями бинарного выбора на основе данных `bwght.dta`, где зави

r

Загрузим необходимы пакеты.

```r
library(rio) # импорт и экспорт данных в разных форматах
library(tidyverse) # графики и манипуляции с данными
library(skimr) # описательные статистики
library(mfx) # нахождение предельных эффектов
library(margins) # визуализация предельных эффектов
```

```
Error in library(margins): there is no package called 'margins'
```

```r
library(lmtest) # проведение тестов
library(plotROC) # построение ROC-кривой
```

```
Error in library(plotROC): there is no package called 'plotROC'
```

```r
library(caret) # confusion-матрица
library(texreg) # вывод результатов регрессии в tex и html
```

Импортируем исследуемые данные.

```r
data = import("data/bwght.dta")
```

```
Error in import("data/bwght.dta"): No such file
```

Сгенерируем переменную `smoke`, отражающее состояние отдельного индивида: курильщик, если `smoke = 1`, не ку
иначе.

```r
data = mutate(data, smoke=(cigs>0))
```

```
Error in UseMethod("mutate_"): no applicable method for 'mutate_' applied to an object of class "function"
```

Рассмотрим описательные статистики по всем переменным: решение курить, семейный доход, налог на сигареты, ц

```r
skim(data)
```

Заметим существование пропущенных переменных у `fatheduc`, `motheduc`. Будем анализировать только те значен

```r
data_2 = filter(data, !is.na(fatheduc), !is.na(motheduc))
```

```
Error in UseMethod("filter_"): no applicable method for 'filter_' applied to an object of class "function"
```

```

```r
skim(data_2)
```

```
Error in skim(data_2): object 'data_2' not found
```
Построим модель линейной вероятности. Сохраним результат под `lin_prob_model`.

```r
lin_prob_model = lm(smoke ~ 1 + faminc + cigtax + cigprice + fatheduc + motheduc + parity + white, data=data_2)
```

```
Error in is.data.frame(data): object 'data_2' not found
```

```r
summary(lin_prob_model)
```

```
Error in summary(lin_prob_model): object 'lin_prob_model' not found
```
Посмотрим на число совпадений прогноза и исходных значений. Для этого оценим предсказанные знач

```r
predictions_lin_prob_model = predict(lin_prob_model)
```

```
Error in predict(lin_prob_model): object 'lin_prob_model' not found
```
Генерируем `smoke_ols` как 1, если вероятность по модели больше 0.5 и 0, если она меньше 0.5.

```r
smoke_ols = 1 * (predictions_lin_prob_model>0.5)
```

```
Error in eval(expr, envir, enclos): object 'predictions_lin_prob_model' not found
```
Число совпадений данных и прогноза модели линейной вероятности:
```

```r
sum (smoke_ols == data_2$smoke)
```

```
Error in eval(expr, envir, enclos): object 'smoke_ols' not found
```

Известно, что модель линейной вероятности обладает значительными недостатками, в частности: нереалистичное з
и пробит-модели.
Немного о логит-модели: предполагается, что существует скрытая (латентная) переменная, для которой строится мо

\[
\begin{equation*}
Y_i =
 \begin{cases}
   1, &\text{если ${y_i}^* \geqslant 0$}\\
   0, &\text{если ${y_i}^* < 0$}
 \end{cases}
\end{equation*}
\]

 $$\varepsilon_i \sim logistic, \\f(t) = \frac{e^{-t}}{(1 + e^{-t})^2}$$

Построим логит-модель и сохраним результат оцененной модели как `logit_model`.

```r
logit_model = glm(smoke ~ 1 + faminc + cigtax + cigprice + fatheduc + motheduc + parity + white, x=TRUE, data=data_2, family
```

```
Error in is.data.frame(data): object 'data_2' not found
```

```r
summary(logit_model)
```

```
Error in summary(logit_model): object 'logit_model' not found
```

Так как коэффициенты логит- и пробит- моделей плохо интерпретируются, поскольку единицы измерения латентно

Для предельного эффекта в средних значениях факторов:

```r
logitmfx(smoke ~ 1 + faminc + cigtax + cigprice + fatheduc + motheduc + parity + white, data=data_2, atmean=TRUE)
```

```

```

```
Error in is.data.frame(data): object 'data_2' not found
```

```r
margins = margins(logit_model)
```

```
Error in margins(logit_model): could not find function ”margins”
```

```r
plot(margins)
```

```
Error in plot(margins): object 'margins' not found
```
Интерпретация предельных эффектов следующая (на примере переменной семейного дохода): при увел

Визуализируем предельный эффект для семейного дохода:

```r
cplot(logit_model, ”faminc”, what=”effect”, main=”Average Marginal Effect of Faminc”)
```

```
Error in cplot(logit_model, ”faminc”, what = ”effect”, main = ”Average Marginal Effect of Faminc”): could not find
```
Для определения качества модели построим классификационную матрицу. Для этого сначала вычислим
модели, `predictions_logit_model`. Так как результат не бинарный, то введём порог отсечения, равный 0.5

```r
predictions_logit_model = predict(logit_model)
```

```
Error in predict(logit_model): object 'logit_model' not found
```

```r
smoke_logit_model = (predictions_logit_model>0.5)
```

```
Error in eval(expr, envir, enclos): object 'predictions_logit_model' not found
```

Построим классификационную матрицу. При возникновении ошибок аргументов, в частности, при несовпадении их

```r
confusionMatrix(as.factor(smoke_logit_model), as.factor(data_2$smoke))
```

```
Error in is.factor(x): object 'smoke_logit_model' not found
```

Качество модели также можно проанализировать с помощью ROC-кривой, отражающей зависимость доли верных положительно классифицируемых наблюдений (`sensitivity`) от доли specifity)`.

Построим ROC-кривую для логит-модели:

```r
basicplot = ggplot(data_2, aes(m=predictions_logit_model, d=data_2$smoke)) + geom_roc()
```

```
Error in ggplot(data_2, aes(m = predictions_logit_model, d = data_2$smoke)): object 'data_2' not found
```

```r
basicplot + annotate("text", x = .75, y = .25,
        label = paste("AUC =", round(calc_auc(basicplot)$AUC, 2)))
```

```
Error in eval(expr, envir, enclos): object 'basicplot' not found
```

Площадь под кривой обозначается как AUC. Он показывает качество классификации. Соответственно, чем выше AU

Теперь рассмотрим логит-модель, не учитывающую переменную `white`. Сохраним эту логит-модель под названием `logit_model_new`.

```r
logit_model_new = glm(smoke ~ 1 + faminc + cigtax + cigprice + fatheduc + motheduc + parity, x=TRUE, data=data_2, family=b
```

```
Error in is.data.frame(data): object 'data_2' not found
```

```
Сравним модели `logit_model` и `logit_model_new` с помощью теста максимального правдоподобия (like
```r
lrtest(logit_model,logit_model_new)
```

```
Error in lrtest(logit_model, logit_model_new): object 'logit_model' not found
```
`p-value = 0.08` в LR-тесте. Следовательно, основная гипотеза о том, что переменная `white` не влияет на р

Сейчас посмотрим на пробит-модель. Скрытая переменная в этой модели распределена стандартно нор
\[
f(t) = \frac{1 \cdot e^{\frac{-t^2}{2}}}{\sqrt{2 \cdot \pi}}
\]

Построим пробит-модель.

```r
probit_model = glm(smoke ~ 1 + faminc + cigtax + cigprice + fatheduc + motheduc + parity + white, data=data_2, f
```

```
Error in is.data.frame(data): object 'data_2' not found
```

```r
summary(probit_model)
```

```
Error in summary(probit_model): object 'probit_model' not found
```
Вычисление предельных эффектов и их интерпретация, построение классификационной матрицы и RO
кривой и LR-тест проводятся аналогично выполненным в логит-модели.
Выведем сравнительную таблицу для построенных моделей.

```r
screenreg(list(lin_prob_model, logit_model, probit_model),
          custom.model.names = c("Модель линейной вероятности", "Логит-
модель", "Пробит-модель"))
```

```
Error in "list" %in% class(l)[1]: object 'lin_prob_model' not found
```
```

```

## python

Попробуем повторить эти шаги, используя **python**.

Импортируем пакеты:

```python
import numpy as np
import pandas as pd # чтение файлов
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'pandas'

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
import matplotlib.pyplot as plt # построение графиков
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'matplotlib'

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
from statsmodels.formula.api import logit, probit, ols # построение логит-, пробит -
и линейной регрессий
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'statsmodels'

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
import statistics # описательные статистики
import sklearn
```
```

```

```

```
Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'sklearn'

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
from sklearn import metrics # для работы с классификационными матрицами
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'sklearn'

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
from sklearn.metrics import roc_curve, auc  # ROC-curve и AUC
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'sklearn'

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
from scipy.stats.distributions import chi2 # хи-квадрат-статистика
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'scipy'

Detailed traceback:
  File "<string>", line 1, in <module>
```

Загрузим данные:

```python
data = pd.read_stata("data/bwght.dta")
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'pd' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```
Уберём пропущенные данные.Выведем описательные статистики по данным.

```python
data_2 = data.dropna()
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'data' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
data_2.describe()
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'data_2' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```
Создадим бинарную переменную `smoke`:

```python
data_2['smoke'] = 1 * (data_2['cigs']>0)
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'data_2' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```
Построим модель линейной вероятности:

```python
lin_prob_model = ols("smoke ~ 1 + faminc + cigtax + cigprice + fatheduc + motheduc + parity + white", data_2).fit()
```

```

```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'ols' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
lin_prob_model.summary()
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'lin_prob_model' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

Создадим переменную `predictions__lin_prob_model`, равную прогнозным значениям модели линейной

```python
predictions_lin_prob_model = lin_prob_model.predict(data_2)
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'lin_prob_model' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
data_2['smoke_ols'] = 1 * (predictions_lin_prob_model>0.5)
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'predictions_lin_prob_model' is not

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
sum(data_2['smoke']==data_2['smoke_ols'])
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'data_2' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

Построим логит-модель.

```python
logit_model = logit("smoke ~ 1 + faminc + cigtax + cigprice + fatheduc + motheduc + parity + white", data_2).fit()
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'logit' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
logit_model.summary()
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'logit_model' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

Посчитаем предельные эффекты в средних значениях переменных для логистической регрессии.

```python
me_mean = logit_model.get_margeff(at='mean')
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'logit_model' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
me_mean.summary()
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'me_mean' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

Посмотрим на точность классификации построенной логит-модели. Для этого вычислим прогнозные зн

```python
predictions_logit_pred = logit_model.predict(data_2) # прогнозирование значений
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'logit_model' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
data_2['smoke_logit_model'] = 1 * (predictions_logit_pred>0.5)
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'predictions_logit_pred' is not defin

Detailed traceback:
  File "<string>", line 1, in <module>
```

Построим классификационную матрицу.

```python
sklearn.metrics.confusion_matrix(data_2['smoke'], data_2['smoke_logit_model'])
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'sklearn' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

Точность прогноза и классификационные данные.

```python
np.round(sklearn.metrics.accuracy_score(data_2['smoke'],data_2['smoke_logit_model']), 2)
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'sklearn' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
sklearn.metrics.classification_report(data_2['smoke'], data_2['smoke_logit_model'])
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'sklearn' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

Выведем ROC-кривую для логит-модели.

```python
fpr, tpr, thresholds = metrics.roc_curve(data_2['smoke'], predictions_logit_pred)
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'metrics' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
auc = metrics.roc_auc_score(data_2['smoke'], predictions_logit_pred)
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'metrics' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
```

plt.plot(fpr,tpr,label=”auc=”+str(np.round(auc, 2)))
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
  File ”<string>”, line 1, in <module>
```

```python
plt.legend(loc=4)
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
  File ”<string>”, line 1, in <module>
```

```python
plt.xlabel('1-Specifity')
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
  File ”<string>”, line 1, in <module>
```

```python
plt.ylabel('Sensitivity')
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
  File ”<string>”, line 1, in <module>
```

```python
plt.title('ROC-curve')
```
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
plt.show()
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

Построим новую логит-модель (`logit_model_new`) без учёта переменной `white`.

```python
logit_model_new = logit("smoke ~ 1 + faminc + cigtax + cigprice + fatheduc + motheduc + parity ", data_2).fit()
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'logit' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
logit_model_new.summary()
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'logit_model_new' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

Так как на момент написания коана готовой реализации функции теста отношения правдоподобия нет, то сделаем е

```python
L1 = logit_model.llf
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'logit_model' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
L2 = logit_model_new.llf
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'logit_model_new' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
def likelihood_ratio(llmin, llmax):
    return(2*(max(llmax, llmin) - min(llmax, llmin)))
LR = likelihood_ratio (L1, L2)
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'L1' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
np.round(chi2.sf(LR, 1), 2) # расчёт p-value для теста
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'chi2' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

Основная гипотеза о незначимости фактора `white` не отвергается на 5% уровне значимости.
Построим пробит-модель.

```python
```

probit_model = probit("smoke ~ 1 + faminc + cigtax + cigprice + fatheduc + motheduc + parity + white", data_2).fit()
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'probit' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```python
probit_model.summary()
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'probit_model' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

Расчёт предельных эффектов, точности классификации, визуализация ROC-кривой и проведение LR-теста проводятся аналогично операциям с логит-моделью.
Сравнение моделей.

```python
pd.DataFrame(dict(col1=lin_prob_model.params, col2=logit_model.params, col3=probit_model.params))
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'pd' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

stata

А сейчас познакомимся с тем, как **stata** работает с моделями бинарного выбора.

Импортируем данные.

```stata
use data/bwght.dta
```
```

```

```

```
file data/bwght.dta not found
r(601);

end of do-file
r(601);
```
Сгенерируем переменную `smoke`.

```stata
gen smoke = (cigs>0) if cigs != .
```

```
 file data/bwght.dta not found
r(601);


cigs not found
r(111);

end of do-file
r(111);
```
Рассмотрим описательные статистики dataframe.

```stata
sum smoke faminc cigtax cigprice fatheduc motheduc parity white
```

```
 file data/bwght.dta not found
r(601);


no variables defined
r(111);

end of do-file
r(111);
```
Уберём пропущенные наблюдения.

```stata
```

sum smoke faminc cigtax cigprice fatheduc motheduc parity white if fatheduc != . & motheduc != .
```

```
 file data/bwght.dta not found
r(601);

no variables defined
r(111);

end of do-file
r(111);
```
Построим модель линейной вероятности. Сохраним результат под `lin_prob_model`.

```stata
reg smoke faminc cigtax cigprice fatheduc motheduc parity white if fatheduc != . & motheduc != .
est store lin_prob_model
```

```
 file data/bwght.dta not found
r(601);


fatheduc not found
r(111);

end of do-file
r(111);
```
Посчитаем количество совпадений прогнозов и исходных значений.

```stata
predict predictions_lin_prob_model
gen smoke_ols = (predictions_lin_prob_model>0.5) if predictions_lin_prob_model != .
count if smoke_ols == smoke
tab smoke_ols smoke
```

```
 file data/bwght.dta not found
r(601);
```
```

```
last estimates not found
r(301);

end of do-file
r(301);
```

Построим логит-модель и сохраним результат оцененной модели как `logit_model`.

```stata
logit smoke faminc cigtax cigprice fatheduc motheduc parity white if fatheduc != . & motheduc != .
est store logit_model
```

```
 file data/bwght.dta not found
r(601);


fatheduc not found
r(111);

end of do-file
r(111);
```

Рассчитаем предельные эффекты в средних значениях переменных.

```stata
margins, dydx(*) atmeans
```

```
 file data/bwght.dta not found
r(601);


last estimates not found
r(301);

end of do-file
r(301);
```

Визуализируем предельные эффекты.

```stata
marginsplot
```


Посмотрим на точность классификации построенной логит-модели. Для этого применяется простая команда:

```stata
estat classification
```

```
 file data/bwght.dta not found
r(601);


last estimates not found
r(301);

end of do-file
r(301);
```

Построим ROC-кривую, показывающую качество классификации построенной логит-модели.

```stata
lroc
```


попробуем построить ещё одну логит-модель без учёта фактора `white` и сохраним новую модель под именем `logit_

```stata
logit smoke faminc cigtax cigprice fatheduc motheduc parity if fatheduc != . & motheduc != .
est store logit_model_new
```

```
 file data/bwght.dta not found
r(601);


fatheduc not found
r(111);

end of do-file
r(111);
```

Сравним `logit_model` и `logit_model_new` с помощью LR (likelihood-ratio test):

```stata
lrtest logit_model logit_model_new
```

```
 file data/bwght.dta not found
r(601);


estimation result logit_model not found
r(111);

end of do-file
r(111);
```

`p-value = 0.08` в LR-тесте. Следовательно, основная гипотеза о том, что переменная `white` не влияет на р

Построим пробит-модель и сохраним результат оцененной модели как `probit_model`.

```stata
probit smoke faminc cigtax cigprice fatheduc motheduc parity white if fatheduc != . & motheduc != .
est store probit_model
```

```
 file data/bwght.dta not found
r(601);


fatheduc not found
r(111);

end of do-file
r(111);
```

Сравним коэффициенты построенных моделей: модели линейной вероятности, логит- и пробит-моделей.

```stata
est tab lin_prob_model logit_model probit_model
```

```
 file data/bwght.dta not found
r(601);
```

```
estimation result lin_prob_model not found
r(111);

end of do-file
r(111);
```

<!--chapter:end:03-binchoice.Rmd-->

Модели множественного выбора {#multchoice}

Загрузим необходимые пакеты.

```r
library(tidyverse) # для манипуляций с данными и построения графиков
library(skimr) # для красивого summary
library(rio) # для чтения .dta файлов
library(margins) # для расчета предельных эффектов
```

```
Error in library(margins): there is no package called 'margins'
```

```r
library(mlogit)
```

```
Error in library(mlogit): there is no package called 'mlogit'
```

```r
library(skimr)
library(nnet)
library(questionr)
```

```
```

Error in library(questionr): there is no package called 'questionr'
```

```r
library(MASS)
library(survival)
library(lattice)
```

r

Импортируем датасет. В нем находятся данные по клиентам пенсионных фондов. Нас интересует перем
в зависимости от ответа респондента на вопрос о предпочтительном способе инвестирования пенсионн

```r
df = rio::import("data/pension.dta")
```

```r
skim_with(numeric = list(hist = NULL, p25 = NULL, p75 = NULL)) #посмотрим на данные

skim(df)
```

```
Skim summary statistics
 n obs: 226
 n variables: 19

-- Variable type:numeric -------------------------------------------------------------------
 variable missing complete   n     mean      sd   p0     p50 p100
      age       0      226 226   60.7      4.29   53   60      73
    black       0      226 226    0.12     0.33    0    0       1
   choice       0      226 226    0.62     0.49    0    1       1
     educ       7      219 226   13.52     2.55    8   12      18
   female       0      226 226    0.6      0.49    0    1       1
  finc100      10      216 226    0.12     0.33    0    0       1
  finc101      10      216 226    0.065    0.25    0    0       1
   finc25      10      216 226    0.21     0.41    0    0       1
   finc35      10      216 226    0.19     0.39    0    0       1
   finc50      10      216 226    0.25     0.43    0    0       1
   finc75      10      216 226    0.12     0.33    0    0       1
       id       0      226 226 2445.09  1371.27   38 2377.5  5014
  irain89       0      226 226    0.5      0.5     0    0.5     1
```
```

```
  married       0      226 226    0.73     0.44    0    1       1
  pctstck       0      226 226   46.68    39.44    0   50     100
  prftshr      20      206 226    0.21     0.41    0    0       1
   pyears       8      218 226   11.39     9.61    0    9      45
 stckin89       0      226 226    0.32     0.47    0    0       1
 wealth89       0      226 226  197.91   242.09 -580  127.85 1485
```

Создадим факторную перменную и упорядочим категории.

```r
df = mutate(df, y = factor(pctstck)) # факторная переменная
df = mutate(df, y = relevel(y, ref = 1)) # сменить базовую категорию
levels(df$y)
```

```
[1] "0"   "50"  "100"
```

Можно взглянуть на значения объясняемой переменной в разрезе какой-то другой переменной. Или посмотреть на картиночку.

```r
table(df$y, df$educ)
```

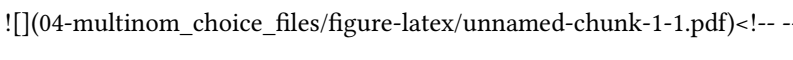
```
      8  9 10 11 12 13 14 15 16 17 18
  0   5  3  0  3 31  4  7  0 11  1  7
  50  1  1  0  3 34  4  6  2 14  5 14
  100 0  2  1  1 36  1  5  4  5  4  4
```

```r
tab = xtabs(~ y + educ, data = df)
prop.table(tab, 1)
```

```
    educ
y            8         9        10        11        12        13
```

```
  0   0.06944444 0.04166667 0.00000000 0.04166667 0.43055556 0.05555556
  50  0.01190476 0.01190476 0.00000000 0.03571429 0.40476190 0.04761905
  100 0.00000000 0.03174603 0.01587302 0.01587302 0.57142857 0.01587302
     educ
y            14         15         16         17         18
  0   0.09722222 0.00000000 0.15277778 0.01388889 0.09722222
  50  0.07142857 0.02380952 0.16666667 0.05952381 0.16666667
  100 0.07936508 0.06349206 0.07936508 0.06349206 0.06349206
```

```r
spineplot(tab, off = 0)
```



Построим модель множественного выбора (лог-линейная модель).

```r
multmodel= multinom(y ~ choice+age+educ+wealth89+prftshr, data = df, reflevel = '50')
```

```
# weights:  21 (12 variable)
initial  value 220.821070
iter  10 value 207.012642
iter  20 value 204.507792
final  value 204.507779
converged
```

```r
summary(multmodel)
```

```
Call:
multinom(formula = y ~ choice + age + educ + wealth89 + prftshr,
    data = df, reflevel = "50")

Coefficients:
    (Intercept)    choice        age       educ     wealth89    prftshr
50     3.777686 0.6269410 -0.10621691 0.18518113 -0.0003716626 -0.2717872
100    4.492971 0.6244954 -0.09482129 0.04644315 -0.0003548369  0.9809245
```

```
Std. Errors:
   (Intercept)    choice       age      educ     wealth89   prftshr
50     1.581691 0.3701263 0.02826469 0.06725443 0.0007365833 0.4988234
100    1.385291 0.3851273 0.02530600 0.07203058 0.0007896235 0.4396202

Residual Deviance: 409.0156
AIC: 433.0156
```

При необходимости можем построить модельку для подвыборки, например, только для замужних/женатых.

```r
multmodel_married = multinom(y ~ choice+age+educ+wealth89+prftshr, subset = married == 1, data = df, reflevel = '50')
```

```
# weights:  21 (12 variable)
initial  value 165.890456
iter  10 value 152.737765
iter  20 value 149.611359
final  value 149.611069
converged
```

```r
summary(multmodel_married)
```

```
Call:
multinom(formula = y ~ choice + age + educ + wealth89 + prftshr,
   data = df, subset = married == 1, reflevel = "50")

Coefficients:
   (Intercept)    choice       age      educ      wealth89   prftshr
50     4.907315 1.0040978 -0.1279041 0.19054837 -0.0006204112 0.1901337
100    5.135424 0.4658502 -0.1145570 0.09046898 -0.0002127724 1.2594092

Std. Errors:
   (Intercept)    choice       age      educ     wealth89   prftshr
50     1.836616 0.4462543 0.03282248 0.07841324 0.0008456801 0.5624022
100    1.551829 0.4583930 0.02890949 0.08508466 0.0008605946 0.5228806

Residual Deviance: 299.2221
AIC: 323.2221
```

```

Быстренько прикинули значимость коэффициентов.

```r
summary(multmodel)$coefficients/summary(multmodel)$standard.errors
```

```
   (Intercept)   choice       age      educ   wealth89    prftshr
50    2.388384 1.693857 -3.757937 2.7534413 -0.5045765 -0.5448566
100   3.243342 1.621530 -3.746989 0.6447699 -0.4493748  2.2313001
```

Сохраним прогнозы.

```r
fit_values = fitted(multmodel)
```

И посчитать относительное изменение отношения шансов:

\[
\frac{P(y_{i} = j)}{P(y_{i} = 1)} = exp(x_{i}\beta)
\] - показывает изменение отношения шансов при выборе альтернативы j вместо альтернативы 0, если x

```r
odds.ratio(multmodel)
```

```
Error in odds.ratio(multmodel): could not find function "odds.ratio"
```

Можем посчитать предельные эффекты в различных квартилях.

```r
summary(marginal_effects(multmodel))
```

```
Error in marginal_effects(multmodel): could not find function "marginal_effects"
```
```

```

Допустим, мы можем упорядочить наши альтернативы (например, от более рискованного способа распределения ре

```r
logit.polr = polr(y ~ choice+age+educ+wealth89+prftshr , data = df)
probit.polr = polr(y ~ choice+age+educ+wealth89+prftshr , data = df, method = 'probit')

### summary(logit.polr) не работает
```

```r
fit_prob = fitted(logit.polr)
fit_log = fitted(probit.polr)
```

stata

```stata
use data/pension.dta
```

sum

Variable	Obs	Mean	Std. Dev.	Min	Max
id	226	2445.093	1371.271	38	5014
pyears	218	11.38532	9.605498	0	45
prftshr	206	.2087379	.4073967	0	1
choice	226	.6150442	.487665	0	1
female	226	.6017699	.49062	0	1
married	226	.7345133	.4425723	0	1
age	226	60.70354	4.287002	53	73
educ	219	13.51598	2.554627	8	18
finc25	216	.2083333	.4070598	0	1
finc35	216	.1851852	.38935	0	1
finc50	216	.2453704	.4313061	0	1
finc75	216	.125	.3314871	0	1
finc100	216	.1203704	.32615	0	1
```

```
    finc101 |       216    .0648148    .2467707          0          1
   wealth89 |       226    197.9057    242.0919   -579.997   1484.997
------------+------------------------------------------------------------
      black |       226     .119469    .3250596          0          1
   stckin89 |       226    .3185841    .4669616          0          1
    irain89 |       226          .5    .5011099          0          1
    pctstck |       226    46.68142    39.44116          0        100
```

ren pctstck y

Построим модель множественного выбора (лог-линейная модель).

```stata
mlogit y choice age educ wealth89 prftshr,  baseoutcome(0)
```

```
Iteration 0:   log likelihood = -219.86356
Iteration 1:   log likelihood = -204.58172
Iteration 2:   log likelihood =  -204.5078
Iteration 3:   log likelihood = -204.50778
Iteration 4:   log likelihood = -204.50778

Multinomial logistic regression                 Number of obs     =        201
                                                LR chi2(10)       =      30.71
                                                Prob > chi2       =     0.0007
Log likelihood = -204.50778                     Pseudo R2         =     0.0698

------------------------------------------------------------------------------
           y |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
0            |  (base outcome)
-------------+----------------------------------------------------------------
50           |
      choice |   .6269473   .3706065     1.69   0.091    -.0994281    1.353323
         age |  -.1062189   .0434194    -2.45   0.014    -.1913193   -.0211185
        educ |   .1851821    .070641     2.62   0.009     .0467283    .3236359
    wealth89 |  -.0003717   .0007432    -0.50   0.617    -.0018283     .001085
     prftshr |  -.2718087   .4988312    -0.54   0.586     -1.2495    .7058825
       _cons |   3.777798   2.790118     1.35   0.176    -1.690732    9.246328
-------------+----------------------------------------------------------------
100          |
      choice |   .6244907   .3859169     1.62   0.106    -.1318925    1.380874
         age |  -.0948282   .0450488    -2.11   0.035    -.1831222   -.0065341
        educ |   .0464378   .0767858     0.60   0.545    -.1040595    .1969352
    wealth89 |  -.0003548    .000797    -0.45   0.656     -.001917    .0012074
```

```
    prftshr |   .9809114   .4396226     2.23   0.026      .119267    1.842556
      _cons |   4.493463   2.967396     1.51   0.130    -1.322526    10.30945
------------------------------------------------------------------------------
```

Можем посмотреть на прогнозы.

```stata
predict p1 p2 p3, p
```

```
(25 missing values generated)
```

И посчитать относительное изменение отношения шансов:

\[
\frac{P(y_{i} = j)}{P(y_{i} = 1)} = exp(x_{i}\beta)
\] - показывает изменение отношения шансов при выборе альтернативы j вместо альтернативы 0, если x изменился н
В stata, в отличие от R, отношение шансов называется relative-risk ratio.

```stata
mlogit, rrr
```

```
Multinomial logistic regression                 Number of obs     =        201
                                                LR chi2(10)       =      30.71
                                                Prob > chi2       =     0.0007
Log likelihood = -204.50778                     Pseudo R2         =     0.0698

------------------------------------------------------------------------------
           y |        RRR   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
0            |  (base outcome)
-------------+----------------------------------------------------------------
50           |
      choice |   1.871888   .6937337     1.69   0.091     .9053551    3.870264
         age |   .8992278   .0390439    -2.45   0.014     .8258688     .979103
        educ |   1.203438    .085012     2.62   0.009     1.047837    1.382144
    wealth89 |   .9996284   .0007429    -0.50   0.617     .9981733    1.001086
     prftshr |       .762   .3801094    -0.54   0.586     .2866481    2.025633
       _cons |   43.71966    121.983     1.35   0.176     .1843845    10366.43
```

```
-------------+----------------------------------------------------------------
100          |
      choice |   1.867295   .7206205     1.62   0.106     .8764352    3.978377
         age |   .9095292   .0409732    -2.11   0.035     .8326664    .9934872
        educ |   1.047533   .0804356     0.60   0.545     .9011717    1.217665
    wealth89 |   .9996452   .0007968    -0.45   0.656     .9980848    1.001208
     prftshr |   2.666886   1.172423     2.23   0.026     1.126671    6.312652
       _cons |   89.43064   265.3761     1.51   0.130     .2664612    30015.02
------------------------------------------------------------------------------
```

Можем посчитать предельные эффекты в разных точках.

```stata
margins, predict(outcome(50)) dydx(choice age educ wealth89 prftshr) atmeans

margins, predict(outcome(50)) dydx(choice age educ wealth89 prftshr) at((p25) *)
```

```
Conditional marginal effects                    Number of obs     =        201
Model VCE    : OIM

Expression   : Pr(y==50), predict(outcome(50))
dy/dx w.r.t. : choice age educ wealth89 prftshr
at           : choice          =     .6069652 (mean)
               age             =     60.52736 (mean)
               educ            =     13.56219 (mean)
               wealth89        =     205.5467 (mean)
               prftshr         =     .2089552 (mean)

------------------------------------------------------------------------------
             |            Delta-method
             |      dy/dx   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      choice |    .077144   .0757102     1.02   0.308    -.0712453    .2255333
         age |   -.014281   .0089754    -1.59   0.112    -.0318725    .0033105
        educ |   .0380169   .0140813     2.70   0.007     .0104182    .0656157
    wealth89 |  -.0000474   .0001544    -0.31   0.759      -.00035    .0002551
     prftshr |  -.1715698   .0989457    -1.73   0.083    -.3654998    .0223602
------------------------------------------------------------------------------


Conditional marginal effects                    Number of obs     =        201
```

```
Model VCE    : OIM

Expression   : Pr(y==50), predict(outcome(50))
dy/dx w.r.t. : choice age educ wealth89 prftshr
at           : choice          =           0 (p25)
               age             =          57 (p25)
               educ            =          12 (p25)
               wealth89        =        65.1 (p25)
               prftshr         =           0 (p25)

------------------------------------------------------------------------------
             |            Delta-method
             |      dy/dx   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      choice |   .0853087   .0708501     1.20   0.229    -.0535549    .2241723
         age |  -.0154741   .0095391    -1.62   0.105    -.0341705    .0032222
        educ |   .0380373   .0133192     2.86   0.004     .0119321    .0641426
    wealth89 |   -.000052    .000152    -0.34   0.732      -.00035     .000246
     prftshr |  -.1534241     .10697    -1.43   0.151    -.3630814    .0562333
------------------------------------------------------------------------------
```

Допустим, мы можем упорядочить наши альтернативы (например, от более рискованного способа распределения ре

```stata
oprobit y choice age educ wealth89 prftshr

ologit y choice age educ wealth89 prftshr
```

```
Iteration 0:   log likelihood = -219.86356
Iteration 1:   log likelihood = -212.89234
Iteration 2:   log likelihood = -212.88817
Iteration 3:   log likelihood = -212.88817

Ordered probit regression                       Number of obs     =        201
                                                LR chi2(5)        =      13.95
                                                Prob > chi2       =     0.0159
Log likelihood = -212.88817                     Pseudo R2         =     0.0317

------------------------------------------------------------------------------
           y |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
```

```
      choice |   .2932272    .167064     1.76   0.079    -.0342122    .6206666
         age |  -.0453065   .0195009    -2.32   0.020    -.0835275   -.0070854
        educ |   .0269375   .0315643     0.85   0.393    -.0349273    .0888024
    wealth89 |  -.0001694   .0003431    -0.49   0.622    -.0008419    .0005031
     prftshr |   .4864833   .2030406     2.40   0.017     .088531    .8844355
-------------+----------------------------------------------------------------
       /cut1 |  -2.578052   1.277878                     -5.082648   -.0734562
       /cut2 |  -1.561798   1.272756                     -4.056353    .9327576
------------------------------------------------------------------------------


Iteration 0:   log likelihood = -219.86356
Iteration 1:   log likelihood = -212.75117
Iteration 2:   log likelihood = -212.72813
Iteration 3:   log likelihood = -212.72813

Ordered logistic regression                     Number of obs     =        201
                                                LR chi2(5)        =      14.27
                                                Prob > chi2       =     0.0140
Log likelihood = -212.72813                     Pseudo R2         =     0.0325

------------------------------------------------------------------------------
           y |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      choice |   .4720438   .2757545     1.71   0.087     -.068425    1.012513
         age |  -.0776337   .0328659    -2.36   0.018    -.1420497   -.0132177
        educ |   .0475714   .0514763     0.92   0.355    -.0533203    .1484631
    wealth89 |   -.000277    .000561    -0.49   0.621    -.0013765    .0008224
     prftshr |   .8312158   .3506528     2.37   0.018     .1439489    1.518483
-------------+----------------------------------------------------------------
       /cut1 |  -4.376271   2.144494                     -8.579402   -.1731395
       /cut2 |  -2.714186   2.129423                     -6.887779    1.459407
------------------------------------------------------------------------------
```

<!--chapter:end:04-multinom_choice.Rmd-->

Модели упорядоченного выбора и условный логит {#ordchoice}

Загрузим необходимые пакеты.

```r
library(tidyverse) # для манипуляций с данными и построения графиков
library(skimr) #для красивого summary
library(rio) # для чтения .dta файлов
library(margins)
```

```
Error in library(margins): there is no package called 'margins'
```

```r
library(mlogit)
```

```
Error in library(mlogit): there is no package called 'mlogit'
```

```r
library(nnet)
library(questionr)
```

```
Error in library(questionr): there is no package called 'questionr'
```

```r
library(MASS)
library(survival)

# log(6)
```

Импортируем датасет. В нем находятся данные по клиентам пенсионных фондов. Нас интересует переменная `pctst
в зависимости от ответа респондента на вопрос о предпочтительном способе инвестирования пенсионных накоплен

```r
df = rio::import("pension.dta")
```

```r
skim_with(numeric = list(hist = NULL, p25 = NULL, p75 = NULL)) #посмотрим на данные
```

```
#skim(df)
```

Создадим факторную перменную и упорядочим категории.

```r
df = rename(df,  alloc = pctstck) # переименуем
df = mutate(df, alloc_factor = factor(alloc)) # факторная переменная
df = mutate(df, y = relevel(df$alloc_factor, ref = 1)) # сменить базовую категорию
levels(df$y)
```

```
[1] "0"   "50"  "100"
```

Построим модель множественного выбора (лог-линейная модель).

```r
multmodel = multinom(y ~ choice+age+educ+wealth89+prftshr, data = df)
```

```
# weights:  21 (12 variable)
initial  value 220.821070
iter  10 value 207.012642
iter  20 value 204.507792
final  value 204.507779
converged
```

```r
summary(multmodel)
```

```
Call:
multinom(formula = y ~ choice + age + educ + wealth89 + prftshr,
    data = df)

Coefficients:
    (Intercept)    choice        age       educ      wealth89    prftshr
50     3.777686 0.6269410 -0.10621691 0.18518113 -0.0003716626 -0.2717872
100    4.492971 0.6244954 -0.09482129 0.04644315 -0.0003548369  0.9809245
```

```
Std. Errors:
   (Intercept)    choice       age      educ     wealth89   prftshr
50    1.581691 0.3701263 0.02826469 0.06725443 0.0007365833 0.4988234
100   1.385291 0.3851273 0.02530600 0.07203058 0.0007896235 0.4396202

Residual Deviance: 409.0156
AIC: 433.0156
```

Сохраним прогнозы.

```r
fit_values = fitted(multmodel)
head(fit_values)
```

```
        0       50       100
1 0.4040703 0.3308134 0.2651163
2 0.1534943 0.2619464 0.5845593
3 0.1651913 0.2342525 0.6005562
4 0.4300671 0.1504960 0.4194370
5 0.4878942 0.2797337 0.2323721
6 0.4642700 0.1265789 0.4091510
```

И посчитать относительное изменение отношения шансов:

$$
\frac{P(y_{i} = j)}{P(y_{i} = 1)} = exp(x_{i}\beta)
$$

показывает изменение отношения шансов при выборе альтернативы j вместо альтернативы 0, если x изменился на e

```r
odds.ratio(multmodel) # отношение шансов в stata называется relative-risk ratio
```

```
Error in odds.ratio(multmodel): could not find function "odds.ratio"
```

Можем посчитать предельные эффекты в различных квартилях.

```r
```

summary(marginal_effects(multmodel)) # mean как в стате
```

```
Error in marginal_effects(multmodel): could not find function ”marginal_effects”
```

Допустим, мы можем упорядочить наши альтернативы (например, от более рискованного способа расп

```r
ordered_logit = polr(y ~ choice+age+educ+wealth89+prftshr , data = df)
ordered_probit = polr(y ~ choice+age+educ+wealth89+prftshr , data = df, method = 'probit')

fit_prob = fitted(ordered_probit)
fit_log = fitted(ordered_logit)
ordered_probit
```

```
Call:
polr(formula = y ~ choice + age + educ + wealth89 + prftshr,
    data = df, method = ”probit”)

Coefficients:
      choice          age         educ     wealth89      prftshr
 0.2932276690 -0.0453064786  0.0269376562 -0.0001693805  0.4864824791

Intercepts:
    0|50    50|100
-2.578050 -1.561799

Residual Deviance: 425.7763
AIC: 439.7763
(25 observations deleted due to missingness)
```

```r
ln(5)
```

```
Error in ln(5): could not find function ”ln”
```
```

```r
cond_logit = clogit(y ~ choice+age+strata(educ)+wealth89+prftshr , data = df)
```

```
Error in coxph(formula = Surv(rep(1, 226L), y) ~ choice + age + strata(educ) + : Cox model doesn't support "mright" survival data
```

То же самое в стате

```stata
use pension.dta
```

```
end of do-file
```

```stata
sum
```

```
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
          id |       226    2445.093    1371.271        38       5014
      pyears |       218    11.38532    9.605498         0         45
     prftshr |       206    .2087379    .4073967         0          1
      choice |       226    .6150442     .487665         0          1
      female |       226    .6017699      .49062         0          1
-------------+--------------------------------------------------------
     married |       226    .7345133    .4425723         0          1
         age |       226    60.70354    4.287002        53         73
        educ |       219    13.51598    2.554627         8         18
      finc25 |       216    .2083333    .4070598         0          1
      finc35 |       216    .1851852      .38935         0          1
-------------+--------------------------------------------------------
      finc50 |       216    .2453704    .4313061         0          1
      finc75 |       216        .125    .3314871         0          1
     finc100 |       216    .1203704      .32615         0          1
     finc101 |       216    .0648148    .2467707         0          1
```

```
   wealth89 |       226    197.9057    242.0919   -579.997    1484.997
------------+------------------------------------------------------------
      black |       226     .119469    .3250596          0           1
    stckin89 |       226    .3185841    .4669616          0           1
     irain89 |       226          .5    .5011099          0           1
    pctstck |       226    46.68142    39.44116          0         100
```

```stata
ren pctstck alloc
```

Построим модель множественного выбора (лог-линейная модель).

mlogit alloc choice age educ wealth89 prftshr, baseoutcome(0) #маленькое отличие с R

> ичие с R
option # not allowed
r(198);

end of do-file
r(198);

Можем посмотреть на прогнозы.

predict p1 p2 p3, p

option # not allowed
r(198);

last estimates not found
r(301);

end of do-file
r(301);

И посчитать относительное изменение отношения шансов:

$$\frac{P(y_i = j)}{P(y_i = 1)} = exp(x_i\beta)$$

- показывает изменение отношения шансов при выборе альтернативы j вместо альтернативы 0, если x изменился на единицу

mlogit, rrr #relative-risk ratio

option # not allowed

```
r(198);


last estimates not found
r(301);

end of do-file
r(301);
```

Можем посчитать предельные эффекты в разных точках.

```
margins, predict(outcome(50)) dydx( choice age educ wealth89 prftshr) atmeans

margins, predict(outcome(50)) dydx( choice age educ wealth89 prftshr) at((p25) *)

 option # not allowed
r(198);


last estimates not found
r(301);

end of do-file
r(301);
```

```
oprobit alloc choice age educ wealth89 prftshr

ologit alloc choice age educ wealth89 prftshr

 option # not allowed
r(198);



Iteration 0:   log likelihood = -219.86356
Iteration 1:   log likelihood = -212.89234
Iteration 2:   log likelihood = -212.88817
Iteration 3:   log likelihood = -212.88817

Ordered probit regression                       Number of obs     =        201
                                                LR chi2(5)        =      13.95
                                                Prob > chi2       =     0.0159
Log likelihood = -212.88817                     Pseudo R2         =     0.0317

------------------------------------------------------------------------------
       alloc |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      choice |   .2932272    .167064     1.76   0.079    -.0342122    .6206666
```

```
       age |  -.0453065   .0195009    -2.32   0.020    -.0835275   -.0070854
      educ |   .0269375   .0315643     0.85   0.393    -.0349273    .0888024
  wealth89 |  -.0001694   .0003431    -0.49   0.622    -.0008419    .0005031
   prftshr |   .4864833   .2030406     2.40   0.017     .088531    .8844355
-------------+----------------------------------------------------------------
     /cut1 |  -2.578052   1.277878                     -5.082648   -.0734562
     /cut2 |  -1.561798   1.272756                     -4.056353    .9327576
------------------------------------------------------------------------------


Iteration 0:   log likelihood = -219.86356
Iteration 1:   log likelihood = -212.75117
Iteration 2:   log likelihood = -212.72813
Iteration 3:   log likelihood = -212.72813

Ordered logistic regression                     Number of obs     =        201
                                                LR chi2(5)        =      14.27
                                                Prob > chi2       =     0.0140
Log likelihood = -212.72813                     Pseudo R2         =     0.0325

------------------------------------------------------------------------------
      alloc |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
     choice |   .4720438   .2757545     1.71   0.087     -.068425    1.012513
       age |  -.0776337   .0328659    -2.36   0.018    -.1420497   -.0132177
      educ |   .0475714   .0514763     0.92   0.355    -.0533203    .1484631
  wealth89 |   -.000277    .000561    -0.49   0.621    -.0013765    .0008224
   prftshr |   .8312158   .3506528     2.37   0.018     .1439489    1.518483
-------------+----------------------------------------------------------------
     /cut1 |  -4.376271   2.144494                     -8.579402   -.1731395
     /cut2 |  -2.714186   2.129423                     -6.887779    1.459407
------------------------------------------------------------------------------
```

Посмотрим на conditional logit

ПОКА ЗАБИЛА

```
use crackers.dta


egen resp = group(id occ)

tabulate brand, generate(br)
rename br1 Sunshine
rename br2 Keebler
rename br3 Nabisco
```

```
clogit choice Sunshine Keebler Nabisco display feature price, group(resp)

 option # not allowed
r(198);


no; data in memory would be lost
r(4);

end of do-file
r(4);
```

Глава 4

Модели счетных данных

Загрузим необходимые пакеты.

```
library(tidyverse) # работа с данными и графики
library(skimr) # красивое summary
library(rio) # чтение .dta файлов
library(MASS) # отрицательное биномиальное
library(lmtest) # для проверки гипотез
library(pscl) # zero-inflation function
```

Error in library(pscl): there is no package called 'pscl'

```
library(margins) # для подсчета предельных эффектов
```

Error in library(margins): there is no package called 'margins'

```
library(sjPlot) # визуализация моделей
```

4.1. r

Импортируем данные.

```
df_fish = rio::import(file = "data/fish.dta")
```

Данные содержат информацию о количестве рыбы, пойманной людьми на отдыхе.

Camper - наличие/отсутсвие палатки. Child - количество детей, которых взяли на рыбалку. Persons - количество людей в группе. Count - количество пойманной рыбы

Посмотрим нам описательные статистики.

```
skim_with(numeric = list(hist = NULL, p25 = NULL, p75 = NULL))
skim(df_fish)
```

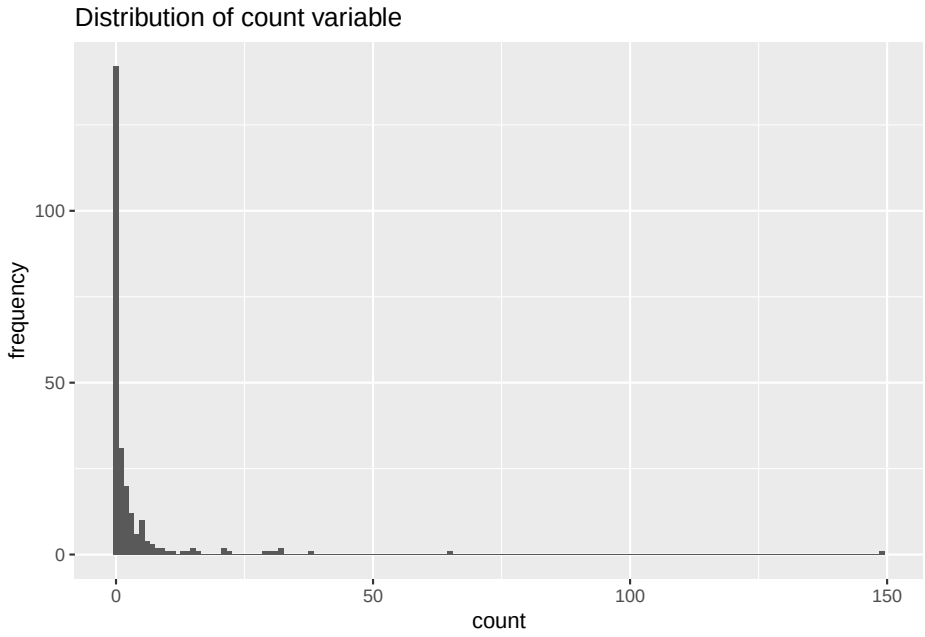
```
Skim summary statistics
 n obs: 250
 n variables: 4

-- Variable type:numeric -------------------------------------------------------------------
 variable missing complete   n mean    sd p0 p50 p100
   camper       0      250 250 0.59  0.49  0   1    1
    child       0      250 250 0.68  0.85  0   0    3
    count       0      250 250 3.3   11.64  0   0  149
  persons       0      250 250 2.53  1.11  1   2    4
```

Переменная camper принимает всего два значения, поэтому превратим ее в факторную переменную.

```
df_fish = mutate(df_fish, camper = factor(camper))
```

Наша задача - по имеющимся данным предсказать улов. Для начала посмотрим на распределение объясняемой переменной count.

```
ggplot(df_fish, aes(x = count)) +
  geom_histogram(binwidth = 1) +
  labs(x = 'count', y = 'frequency', title = 'Distribution of count variable')
```



Предположим, что переменная имеет распределение Пуассона. Будем исполь-

зовать пуассоновскую регрессию.

$$P(y = k) = exp(-\lambda)\lambda^k/k!$$

где $\lambda = \exp(b_1 + b_2 * x)$

```
poisson_model = glm(count ~ child + camper +  persons, family = "poisson", data = df_fish)
summary(poisson_model)
```

```
Call:
glm(formula = count ~ child + camper + persons, family = "poisson",
    data = df_fish)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-6.8096  -1.4431  -0.9060  -0.0406  16.1417

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.98183    0.15226  -13.02   <2e-16 ***
child       -1.68996    0.08099  -20.87   <2e-16 ***
camper1      0.93094    0.08909   10.45   <2e-16 ***
persons      1.09126    0.03926   27.80   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 2958.4  on 249  degrees of freedom
Residual deviance: 1337.1  on 246  degrees of freedom
AIC: 1682.1

Number of Fisher Scoring iterations: 6
```

Посчитаем средний предельный эффект для каждой переменной.

```
m = margins(poisson_model)
```

```
Error in margins(poisson_model): could not find function "margins"
```

```
summary(m)
```

```
Error in summary(m): object 'm' not found
```

```
cplot(poisson_model, x = 'persons', what = 'effect', title = 'Предельный эффект переменной camper')
```

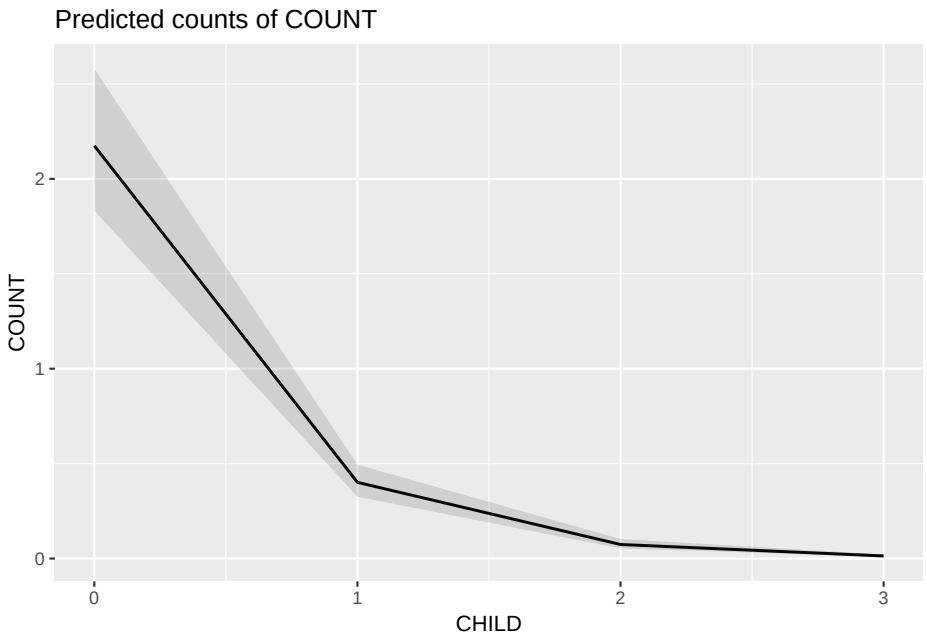
```
Error in cplot(poisson_model, x = "persons", what = "effect", title = "Предельный эффект переменной camper"): could not fir
```

```
margins(poisson_model, at = list(child = 0:1)) # или в какой-нибудь точке
```

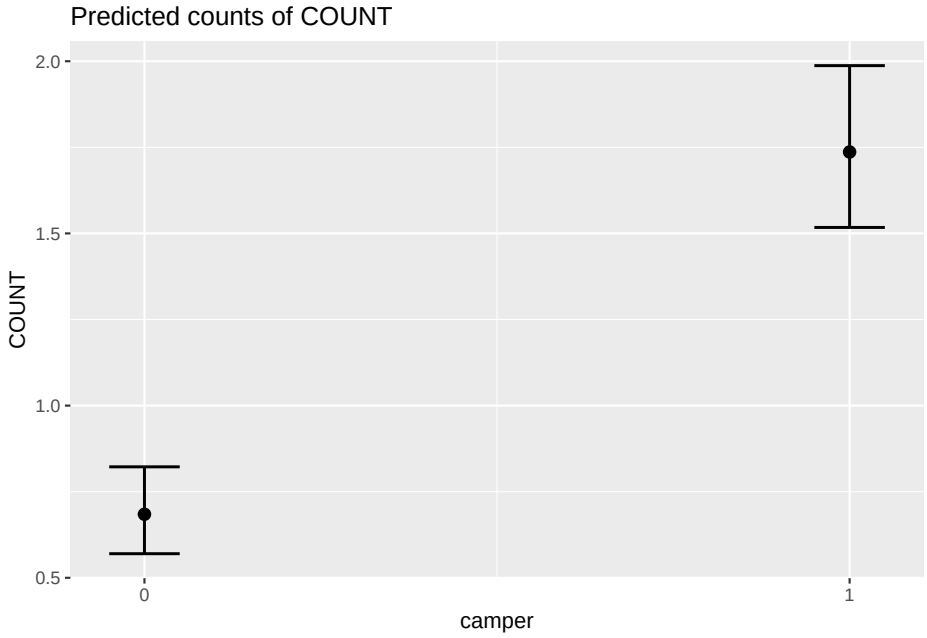
Error in margins(poisson_model, at = list(child = 0:1)): could not find function ”margins”

```
plot_model(poisson_model, type = 'pred')
```

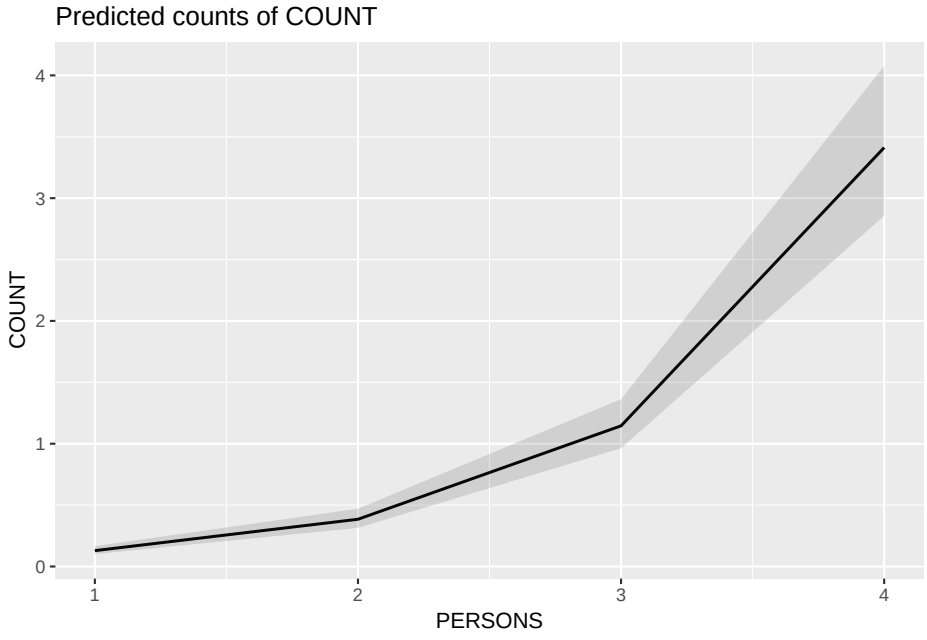
$child



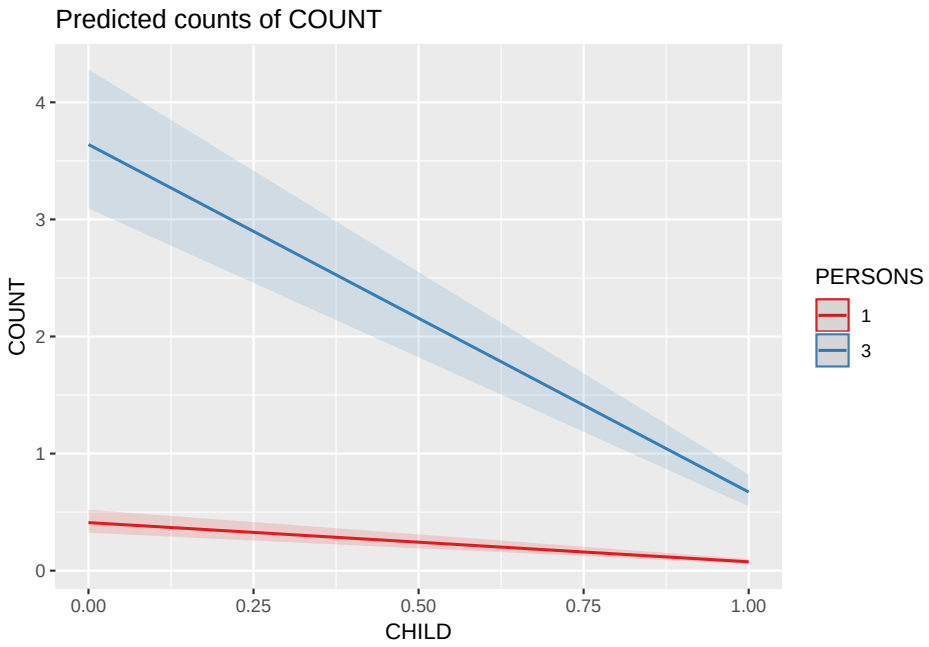
$camper


Predicted counts of COUNT
2.0
1.5
1.0
0.5
COUNT
0
1
camper

$persons


Predicted counts of COUNT
4
3
2
1
0
COUNT
1
2
3
4
PERSONS

```
plot_model(poisson_model, type = "pred", terms = c("child [0, 0, 1]", "persons [1,3]"))
```



Однако, заметим, что дисперсия и среднее значение объясняемой переменной не равны, как это предполагает распределение Пуассона.

```
df_fish %>%
  group_by(camper) %>%
  summarize(var = var(count), mean = mean(count))
```

```
# A tibble: 2 x 3
  camper   var  mean
  <fct>  <dbl> <dbl>
1 0       21.1  1.52
2 1      212.   4.54
```

Оценим регрессию, предполагая отрицательное биномиальное распределение остатков. В этом случае, дисперсия распределения зависит от некоторого параметра и не равна среднему.

```
nb1 = glm.nb(count ~ child + camper +  persons, data = df_fish)
summary(nb1)
```

```
Call:
glm.nb(formula = count ~ child + camper + persons, data = df_fish,
    init.theta = 0.4635287626, link = log)
```

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6673  -0.9599  -0.6590  -0.0319   4.9433

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -1.6250     0.3304  -4.918 8.74e-07 ***
child        -1.7805     0.1850  -9.623  < 2e-16 ***
camper1       0.6211     0.2348   2.645  0.00816 **
persons       1.0608     0.1144   9.273  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for Negative Binomial(0.4635) family taken to be 1)

    Null deviance: 394.25  on 249  degrees of freedom
Residual deviance: 210.65  on 246  degrees of freedom
AIC: 820.44

Number of Fisher Scoring iterations: 1

              Theta:  0.4635
          Std. Err.:  0.0712

 2 x log-likelihood:  -810.4440
```

Попробуем исключить из модели переменную camper и сравним качество двух моделей.

```
nb2 = update(nb1, . ~ . - camper)
waldtest(nb1, nb2)
```

```
Wald test

Model 1: count ~ child + camper + persons
Model 2: count ~ child + persons
  Res.Df Df      F   Pr(>F)
1    246
2    247 -1 6.9979 0.008686 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Можем посмотреть на результаты модели с “раздутыми нулями” (zero-inflated). Они предполагают большую частоту нулевых наблюдений.

```
zero_infl = zeroinfl(count ~  child + camper | persons, data = df_fish, dist = 'negbin')
```

```
Error in zeroinfl(count ~ child + camper | persons, data = df_fish, dist = "negbin"): could not find function "zeroinfl"
```

```
summary(zero_infl)
```

Error in summary(zero_infl): object 'zero_infl' not found

```
plot_model(zero_infl, type = 'pred')
```

Error in insight::model_info(model): object 'zero_infl' not found

4.2. python

Нужные пакетики:

```
import pandas as pd # для работы с таблицами
```

Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'pandas'

Detailed traceback:
File "<string>", line 1, in <module>

```
import numpy as np # математика, работа с матрицами
import matplotlib.pyplot as plt # графики
```

Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'matplotlib

Detailed traceback:
File "<string>", line 1, in <module>

```
import statsmodels.api as sm
```

Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'statsmodel

Detailed traceback:
File "<string>", line 1, in <module>

```
import statsmodels.formula.api as smf
```

Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'statsmodel

Detailed traceback:
File "<string>", line 1, in <module>

```
import statsmodels.graphics.gofplots as gf
```

Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'statsmodel

Detailed traceback:
File "<string>", line 1, in <module>

```
from statsmodels.stats.outliers_influence import summary_table
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'statsmodels'

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
import seaborn as sns # еще более классные графики
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'seaborn'

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
from scipy.stats import shapiro # еще математика
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'scipy'

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
import statsmodels.discrete.discrete_model
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'statsmodels'

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
from statsmodels.discrete.count_model import ZeroInflatedPoisson
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): ModuleNotFoundError: No module named 'statsmodels'

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
plt.style.use('ggplot')
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

Загружаем данные и смотрим описательные статистики.

```
df_fish = pd.read_stata('data/fish.dta')
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'pd' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
sns.distplot(df_fish['count'])
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'sns' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

```
plt.show()
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plt' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

Превращаем переменную camper в категориальную.

```
df_fish['camper'] = df_fish['camper'].astype('category')
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'df_fish' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

Строим Пуассоновскую регрессию.

```
pois = statsmodels.discrete.discrete_model.Poisson(endog = count, exog = np.array(child, camper, persons), dat
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'statsmodels' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

```
regr_pois = smf.glm('count ~ child + camper +  persons', data=df_fish,
                    family=sm.families.Poisson()).fit()
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'smf' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

```
regr_pois.summary()
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'regr_pois' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

Посмотрим, равны ли среднее значение и дисперсия, как это предполагает распределение Пуассона.

```
(df_fish
 .filter(['count', 'camper'])
 .groupby('camper')
 .agg(['mean', 'var']))
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'df_fish' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

И регрессию с остатками, имеющими отрицательное биномиальное распределение.

```
regr_bin = smf.glm('count ~ child + camper +  persons', data=df_fish,
                   family=sm.families.NegativeBinomial()).fit()
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'smf' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

```
regr_bin.summary()
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'regr_bin' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

Проверим гипотезу о равенстве 0 коэффициента при переменной camper. Проведем тест Вальда.

```
hyp = '(child = 0)'
regr_bin.wald_test(hyp)
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'regr_bin' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

Посчитаем средний предельный эффект для каждой переменной.

```
pred = regr_pois.fittedvalues
```

Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'regr_pois' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>

```
mean_mef_child = np.mean([regr_pois.params[1] * p for p in pred])
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'pred' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
mean_mef_camper = np.mean([regr_pois.params[2] * p for p in pred])
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'pred' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
data_1 = pd.DataFrame({'child': df_fish['child'], 'camper': 1, 'persons': df_fish['persons']})
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'pd' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
data_0 = pd.DataFrame({'child': df_fish['child'], 'camper': 0, 'persons': df_fish['persons']})
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'pd' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

```
mean_mef_persons = np.mean([(regr_pois.predict(data_1)[i]-regr_pois.predict(data_0)[i])
                            for i in range(len(df_fish))])
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'df_fish' is not defined

Detailed traceback:
  File "<string>", line 2, in <module>
```

```
plot_model(regr_pois, type = 'effect', terms = 'camper')
```

```
Error in py_call_impl(callable, dots$args, dots$keywords): NameError: name 'plot_model' is not defined

Detailed traceback:
  File "<string>", line 1, in <module>
```

И модель с раздутыми нулями. (которой нет)

4.3. stata

Загружаем данные и смотрим описательные статистики.

```
use data/fish.dta
summarize
```