# ЛАБОРАТОРНАЯ РАБОТА 2. ЛИНЕЙНАЯ РЕГРЕССИЯ

## Цель работы

Изучить приемы исследования корреляционной зависимости, построения парной и множественной линейной регрессии.

## Задание

**1.** Загрузить набор данных для своего варианта, ознакомиться с его содержимым.

**2.** Построить график корреляционного поля для каждого фактора.

**3.** Построить уравнение парной линейной регрессии для каждого фактора.

**4.** Проверить значимость каждого из полученных уравнений регрессии. Показать уравнения регрессии с заданным в варианте доверительным интервалом на графиках.

**5.** Построить прогнозы по каждому из уравнений парной регрессии для заданных в варианте значений факторов.

**6.** Построить уравнение множественной линейной регрессии и получить корреляционную матрицу.

**7.** Построить прогноз по уравнению множественной регрессии для заданных в варианте значений факторов.

## Указания к выполнению работы

### Парная линейная регрессия

Рассмотрим построение парной линейной регрессии на встроенном наборе данных `cars`.

```
d <- cars
```

Будем рассматривать зависимость длины тормозного пути (переменная `dist`) от скорости (переменная `speed`).

Построим график зависимости длины тормозного пути от скорости автомобиля:

```
qplot(data=d, speed, dist)
```

Чтобы настроить внешний вид графика, необходимо использовать функцию `ggplot` (рис. 11).

```
ggplot() +
  geom_point(aes(x=d$speed, y=d$dist), size = 2) + theme_bw(base_size =
18) +
  xlab("Скорость, миль/ч") + ylab("Длина тормозного пути, футы") +
  labs(title = "Корреляционное поле")
```

Данная функция имеет множество других настроек, с которыми можно ознакомиться в справке [6]. Оценим модель линейной регрессии длины тормозного пути на скорость автомобиля.

Для этого командой `lm` поместим в переменную `model` модель линейной регрессии, указав `dist` в качестве зависимой переменной, и через значок ~ переменную `speed` в качестве регрессора:

```
model <- lm(data=d, dist~speed) # базовый пакет stats
```


*Рис. 11*. График зависимости длины тормозного пути `dist` от скорости автомобиля `speed`

Тип `lm` представляет собой список из 12 элементов (рис. 12).

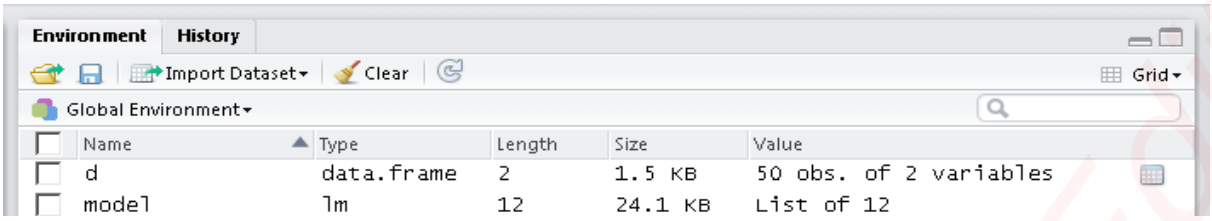


*Рис. 12*. Переменная-список `lm` в в таблице среды `Environment`

Посмотрим на коэффициенты уравнения линейной регрессии:

```
model$coefficients
```

Результат в консоли:

```
> model$coefficients
(Intercept)       speed
 -17.579095    3.932409
```

`(Intercept)` — это константа в уравнении регрессии, `speed` — коэффициент регрессии.

Таким образом, уравнение регрессии имеет вид:

$$dist_i^m = -17.579 + 3.9324 \cdot speed_i$$

Также можно посмотреть значения вектора ошибок модели — разницу между реальной длиной тормозного пути `dist` и полученной по модели $dist_i^m$. Выведем первые 10 значений этого вектора с точностью две цифры после запятой:

```
model$residuals[1:10]
options(digits = 3)
```

```
    1     2     3     4     5     6     7     8     9    10
 3.85 11.85 -5.95 12.05  2.12 -7.81 -3.74  4.26 12.26 -8.68
```

Более полный набор расчетов по модели можно получить командой `summary`:

```
summary(model) # базовый пакет base
```

```
Call:
lm(formula = dist ~ speed, data = d)

Residuals:
    Min      1Q  Median      3Q     Max
-29.069  -9.525  -2.272   9.215  43.201

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -17.5791     6.7584  -2.601   0.0123 *
speed         3.9324     0.4155   9.464 1.49e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 15.38 on 48 degrees of freedom
Multiple R-squared:  0.6511,    Adjusted R-squared:  0.6438
F-statistic: 89.57 on 1 and 48 DF,  p-value: 1.49e-12
```

Помимо коэффициентов регрессии, R выводит:

– стандартные ошибки коэффициентов (`Std. Error`);

– наблюдаемые значения t-критерия при проверке значимости коэффициентов регрессии (`t value`);

– P-значения для коэффициентов регрессии (`P-value`).

Звездочками или точками в столбце справа R показывает значимость или незначимость коэффициентов: `***` — значимы на уровне значимости менее 0.001; `**` — значимы на уровне значимости 0.001; `*` — значимы на уровне значимости 0.01; `.` — значимы на уровне значимости 0.05 и т. д. Эти обозначения приведены в разделе `Signif. codes`.

Коэффициент детерминации (`Multiple R-squared`) равен 0.6511; скорректированный коэффициент детерминации (`Adjusted R-squared`) равен 0.6438. Наблюдаемое значения F-критерия проверки значимости уравнения в целом и P-значение:

```
F-statistic: 89.57 on 1 and 48 DF,  p-value: 1.49e-12
```

Таким образом, уравнение регрессии получилось значимым.

Проведем на графике полученную линию регрессии с 95% доверительными интервалами (рис. 13):

```
qplot(data = d, speed, dist) + stat_smooth(method="lm", level = 0.95) +
  theme_bw(base_size = 18)
```

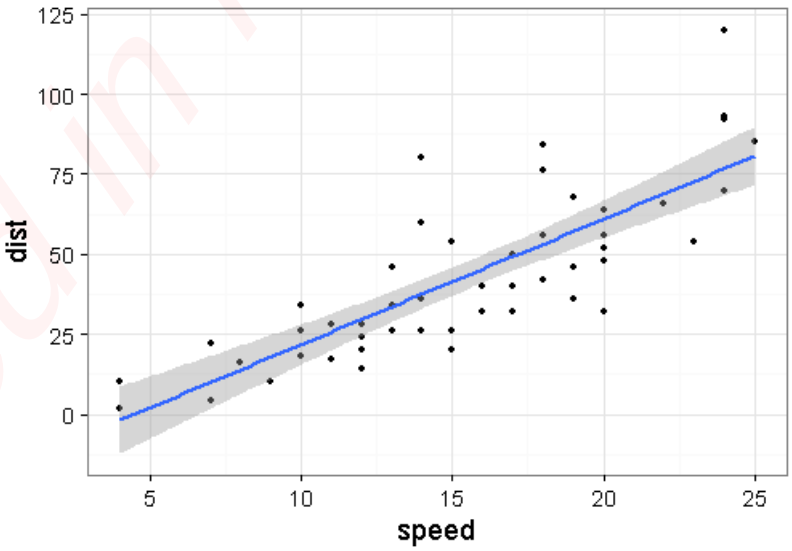


*Рис. 13*. График линейной регрессии с доверительными интервалами

Рассчитаем также 95% доверительные интервалы для параметров линейной регрессии:

```
> confint(model, level = 0.95) # базовый пакет stats

                2.5 %    97.5 %
(Intercept) -31.167850 -3.990340
speed         3.096964  4.767853
```

Рассчитанные по модели значения $dist_i^m = -17.579 + 3.9324 \cdot speed_i$ можно получить командой

```
fitted(model) # базовый пакет stats
```

Необъясненная сумма квадратов отклонений:

```
RSS <- deviance(model) # базовый пакет stats
```

Можно также рассчитать полную сумму квадратов, воспользовавшись уже известными функциями `sum` и `mean`:

```
TSS <- sum((y-mean(y))^2)
```

Для того чтобы построить прогноз по полученной модели, нужно задать значения регрессора и поместить их в новый `data.frame`.

```
# создаем новый набор данных
nd <- data.frame(speed=c(40,60))
```

Строим прогноз функцией `predict`:

```
> predict(model,nd)
       1        2
139.7173 218.3654
```

**Множественная линейная регрессия**

Рассмотрим встроенный набор данных по социально-экономическим показателям в 47 провинциях Швейцарии в 1888 г.

```
t <- swiss # встроенный набор данных по Швейцарии
```

Этот набор данных содержит 6 переменных по 47 наблюдений, каждая из которых измеряется в процентах (`help(swiss)`):

`Fertility` — рождаемость;

`Agriculture` — % мужчин, занятых в сельском хозяйстве;

`Examination` — % призывников, получивших высшую оценку на экзамене в армии;

`Education` — % призывников, имеющих образование помимо начального;

`Catholic` — % католиков среди населения;

`Infant.Mortality` — % детей, умерших до года.

Посмотрим на этот набор данных:

```
> glimpse(t)
Observations: 47
Variables: 6
$ Fertility     (dbl) 80.2, 83.1, 92.5, 85.8, 76.9, 76.1, 83.8, 92.4...
$ Agriculture   (dbl) 17.0, 45.1, 39.7, 36.5, 43.5, 35.3, 70.2, 67.8...
$ Examination   (int) 15, 6, 5, 12, 17, 9, 16, 14, 12, 16, 14, 21, 1...
$ Education     (int) 12, 9, 5, 7, 15, 7, 7, 8, 7, 13, 6, 12, 7, 12,...
$ Catholic      (dbl) 9.96, 84.84, 93.40, 33.77, 5.16, 90.57, 92.85,...
$ Infant.Mortality (dbl) 22.2, 22.2, 20.2, 20.3, 20.6, 26.6, 23.6, 24.9...
```

Встроенный пакет `graphics` содержит функцию `pairs`, позволяющую получить все возможные диаграммы рассеяния на одном графике, а также выполнить их сглаживание с помощью опции `panel.smooth`:

```
pairs(swiss, panel = panel.smooth)
```

Результатом будет график, показанный на рисунке 14.

Функция `cor` позволяет как вычислить корреляцию между двумя выборками, так и получить корреляционную матрицу для всех переменных из набора данных:

```
> cor(swiss)
                 Fertility Agriculture Examination Education Catholic
Fertility            1.000      0.3531      -0.646   -0.6638    0.464
Agriculture          0.353      1.0000      -0.687   -0.6395    0.401
Examination         -0.646     -0.6865       1.000    0.6984   -0.573
Education           -0.664     -0.6395       0.698    1.0000   -0.154
Catholic             0.464      0.4011      -0.573   -0.1539    1.000
Infant.Mortality     0.417     -0.0609      -0.114   -0.0993    0.175
                 Infant.Mortality
Fertility                  0.4166
Agriculture               -0.0609
Examination               -0.1140
Education                 -0.0993
Catholic                   0.1755
Infant.Mortality           1.0000
```

Корреляционную матрицу можно получить и в других видах, например, с помощью функции `sjp.corr` из пакета `sjPlot` (рис. 15):

```
library("sjPlot")
sjp.corr(t)
```

Существует еще одна функция, позволяющая получить корреляционную матрицу, диаграммы рассеяния и сглаженные распределения одновременно (рис. 16):

```
library("GGally")
ggpairs(t) # функция из пакета GGally
```

Чтобы оценить регрессию рождаемости на остальные переменные, можно воспользоваться уже знакомой функцией `lm`, а регрессоры перечислить через знак «плюс»:

```
model2 <- lm(data=t, Fertility~Agriculture+Education+Catholic)
```

В данном случае регрессорами стали % занятых в с/х; % католического населения и % имеющих образование выше начального.

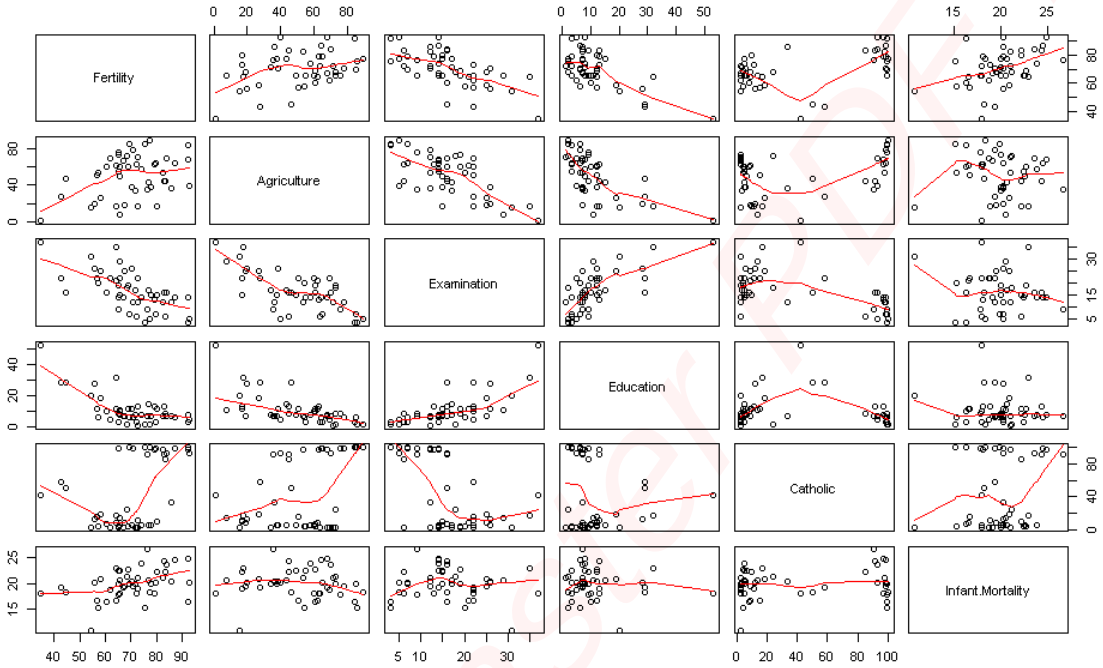


*Рис. 14*. Диаграммы рассеяния, полученные с помощью функции `pairs`

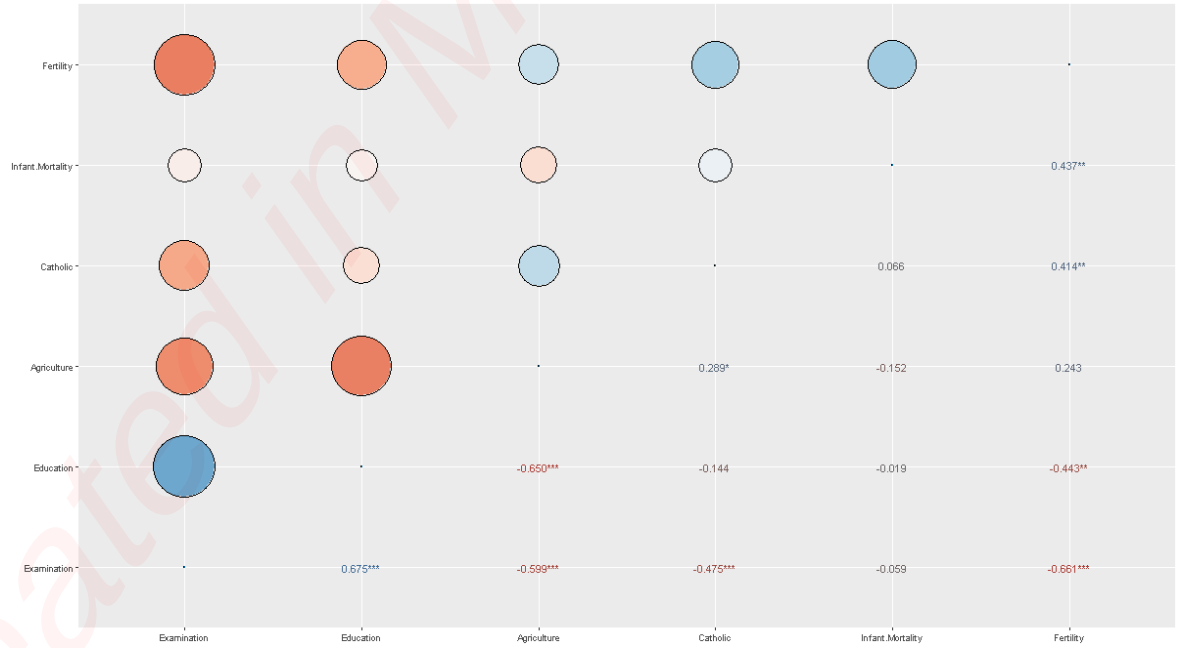


*Рис. 15*. Корреляционная матрица, полученная с помощью функции `sjp.corr`

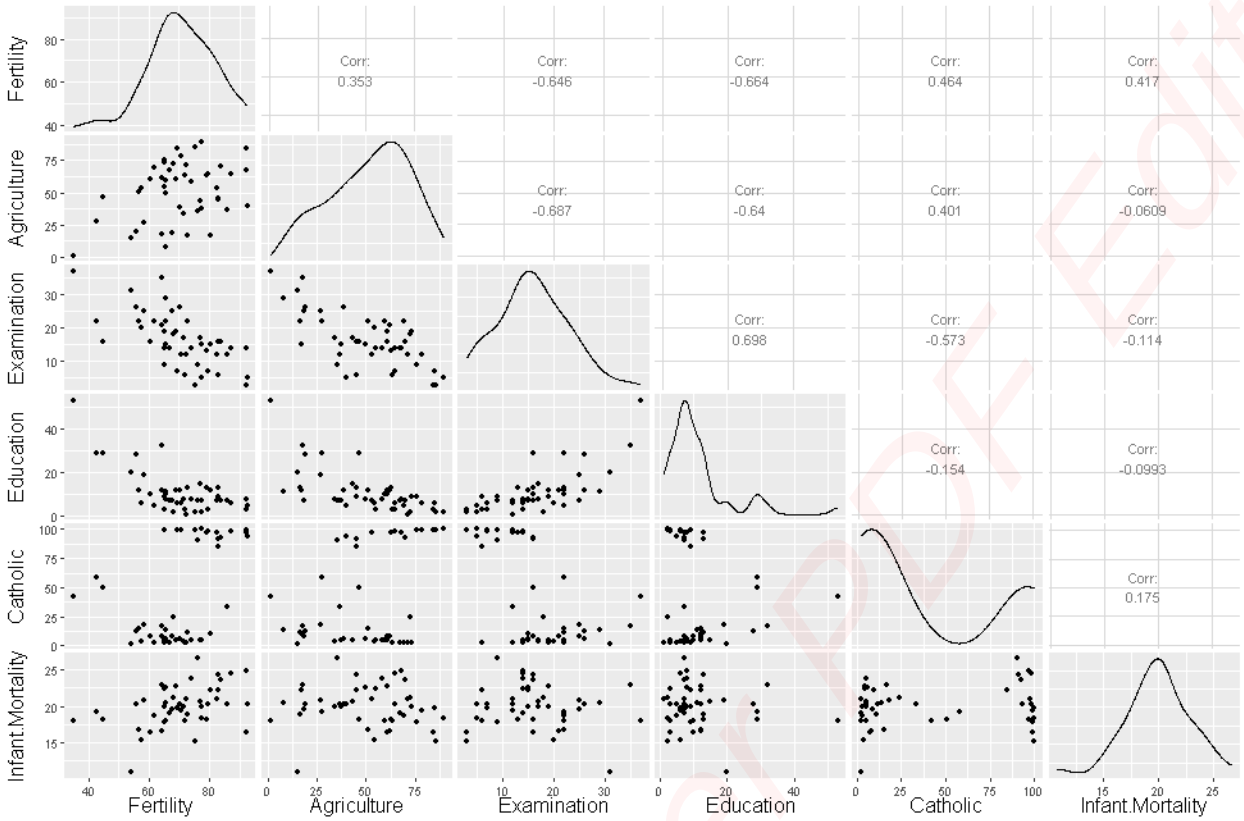


*Рис. 16*. Корреляционная матрица, диаграммы рассеяния и сглаженные распределения, полученные с помощью функции `ggpairs`

Получить оценки коэффициентов уравнения регрессии, а также проверить основные гипотезы поможет функция `summary`:

```
> summary(model2)

Call:
lm(formula = Fertility ~ Agriculture + Education + Catholic,
    data = t)

Residuals:
    Min      1Q  Median      3Q     Max
-15.178  -6.548   1.379   5.822  14.840

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 86.22502    4.73472  18.211  < 2e-16 ***
Agriculture -0.20304    0.07115  -2.854  0.00662 **
Education   -1.07215    0.15580  -6.881 1.91e-08 ***
Catholic     0.14520    0.03015   4.817 1.84e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 7.728 on 43 degrees of freedom
Multiple R-squared:  0.6423,    Adjusted R-squared:  0.6173
F-statistic: 25.73 on 3 and 43 DF,  p-value: 1.089e-09
```

Построим прогноз по аналогии с парной линейной регрессией. Отличие заключается лишь в том, что в наборе данных необходимо указать значения каждого фактора:

```
# создаем новый набор данных
nd2 <- data.frame(Agriculture=0.5,Catholic=0.5, Education=20)
> predict(model2,nd2)
       1
64.75316
```

Построение прогноза по нескольким точкам выполняется с помощью векторов значений:

```
# создаем новый набор данных
nd2 <- data.frame(Agriculture=c(0.5,0.8),Catholic=c(0.5, 0.65),
                  Education=c(20, 25))
# прогнозируем
predict(model2,nd2)
       1        2
64.75316 59.35330
```