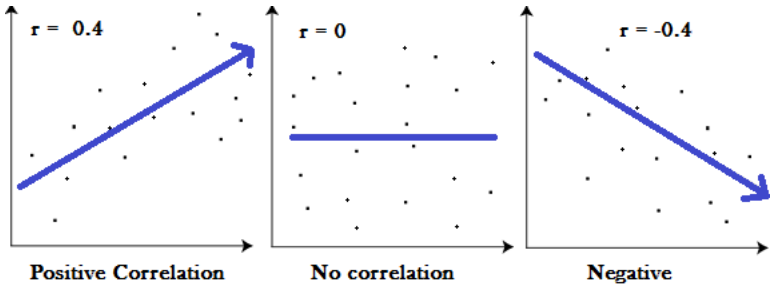
KARPOV.COURSES >>>
КОНСПЕКТ

> Конспект > 6 урок > СТАТИСТИКА

> Понятие корреляции

Коэффициент корреляции в своём статистическом смысле обозначает силу и характер взаимосвязи между двумя количественными переменными. Взаимосвязь может быть положительной (когда одна переменная растёт, другая тоже растёт), либо отрицательной (когда одна переменная растёт, другая уменьшается), либо отсутствовать.



Допустим, нам нужно рассчитать коэффициент корреляции для двух количественных величин X и Y. Соответственно, среднее по выборке для каждой из этих величин будет $\overline{X}$ и $\overline{Y}$. В таком случае по следующей формуле у нас получится коэффициент **ковариации**:

$$cov(x,y) = \frac{\sum(X_i - \bar{X})(Y_i - \bar{Y})}{N-1}$$

Здесь N обозначает размер нашей выборки, а X_i и Y_i – значения каждого индивидуального наблюдения *i* в выборке.

Иными словами, мы берём наблюдение, вычитаем его значение по переменной X из среднего по этой переменной, делаем то же самое для переменной Y, перемножаем результаты. Так делаем для каждого из наблюдений выборки, суммируем то, что получилось, и делим на количество наблюдений в выборке минус один.

Проблема: ковариация изменяется в произвольном разбросе значений => сложно интерпретировать силу взаимосвязи. Коэффициент корреляции считается на основе ковариации (cov):

$$r_{xy} = \frac{cov}{\delta_x \delta_y}$$

δ – это стандартное отклонение соответствующей переменной. Это преобразование приводит к тому, что значения коэффициента корреляции вне зависимости от шкалы исходных переменных находятся в диапазоне от -1 (сильная отрицательная) до +1 (сильная положительная) → проще интерпретация результатов.

В учебниках формулу часто можно встретить в таком виде:

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

Это называется коэффициентом корреляции **Пирсона.**

Приложение, в котором можно самому изучить, как работает корреляция: https://rpsychologist.com/d3/correlation/

> Проверка гипотез

- H_0– коэффициент корреляции равен нулю
- H_1 – коэффициент корреляции не равен нулю

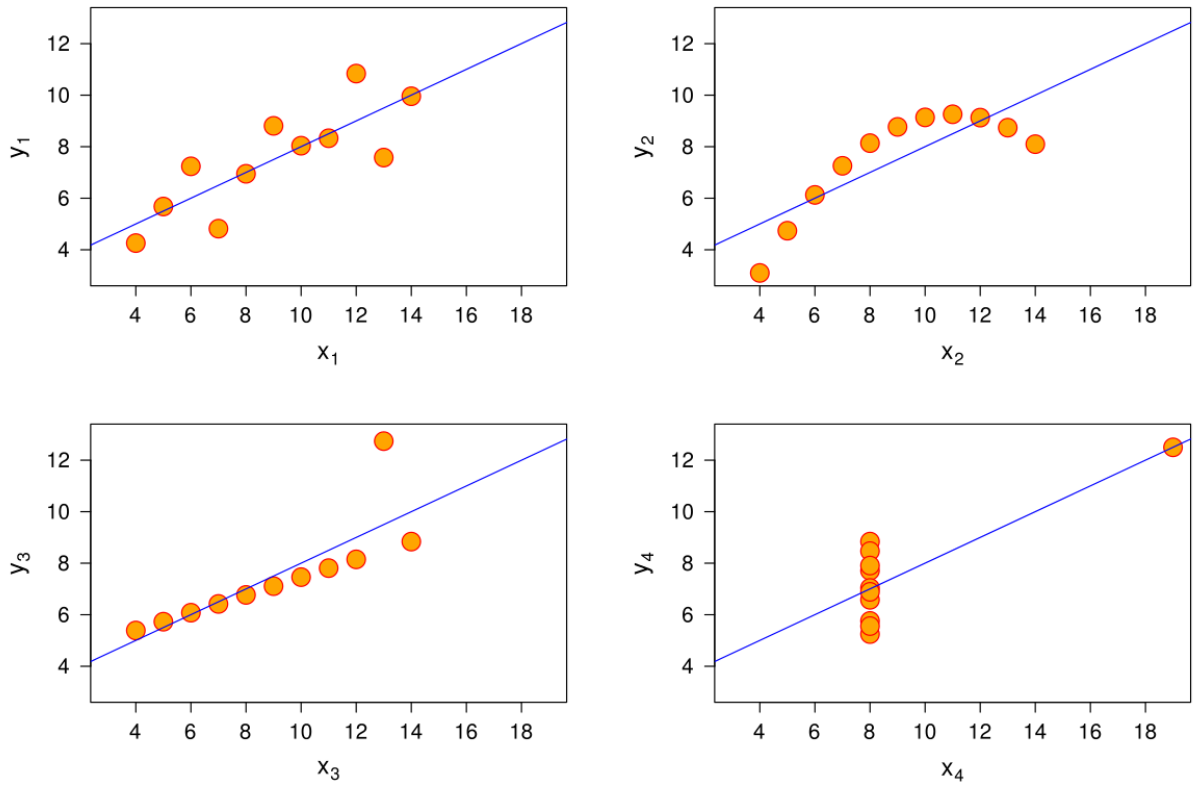
Значимость рассчитывается с использованием t-распределения с количеством степеней свободы $df = N - 2$.

Условия применения:

1. Связь линейна и монотонна (нарастает или убывает в одном направлении, не меняя его)
2. Отсутствуют выбросы
3. Переменные нормально распределены

В случае нарушения этих допущений могут быть полезны коэффициенты корреляции Спирмена и Кэндалла, которые вместо реальных значений анализируют их ранги.

Note: Ни то, ни другое не является панацеей, и при особенно сильных нарушениях допущений (особенно первого) они также могут быть неадекватны. Внимательно изучайте свои данные и их характер! Один из классических примеров необходимости проверки допущений (и визуализации данных перед анализом) – **квартет Энскомба**. У всех четырёх наборов данных одинаковый коэффициент корреляции Пирсона, хотя их характер очевидно различен.



```
import numpy as np
import scipy.stats as st
import pandas as pd

# через numpy (только Пирсона, без p-значений)
np.corrcoef(x, y)  # у опционален, можно дать массив с несколькими колонками, функция строит матрицу корреляций# через scipy (даёт значение
st.pearsonr(x, y)
st.spearmanr(x, y)
st.kendalltau(x, y)

# через pandas (сравнение pandas Series)
df.corr()
df.corr(method='spearman')
df.corr(method='kendall')
```

Функцию `np.corrcoef()` имеет смысл использовать только в том случае, если у вас несколько переменных и вы хотите увидеть попарные корреляции каждой из них. На выходе тогда будет корреляционная матрица, по левой диагонали которой будут единицы (так как там переменная коррелируется сама с собой):

	Connectivity	Digital Public Services	Human Capital	Integration of Digital Technology	Use of Internet
Connectivity	1.00	0.64	0.71	0.65	0.77
Digital Public Services	0.64	1.00	0.58	0.64	0.62
Human Capital	0.71	0.58	1.00	0.66	0.72
Integration of Digital Technology	0.65	0.64	0.66	1.00	0.60
Use of Internet	0.77	0.62	0.72	0.60	1.00

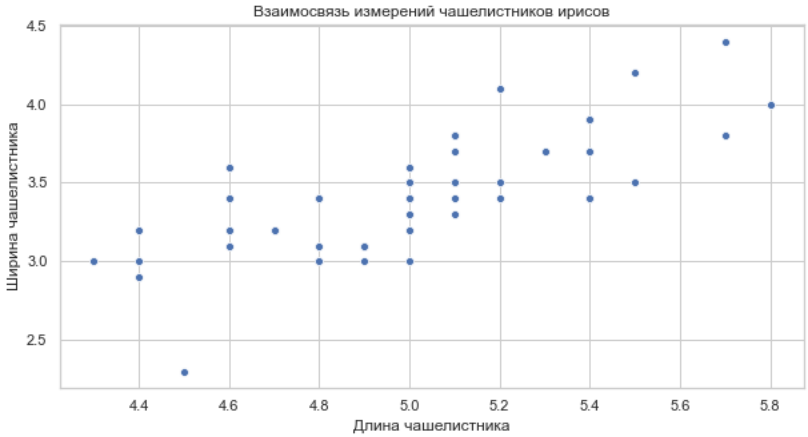
Дополнительно: существует более современный и продвинутый вариант квартета, называемый Датазавровой дюжиной, также существует аналог коэффициента корреляции, способный оценивать крайне нелинейные взаимосвязи в данных.

> Как рисовать

График корреляции называется диаграммой рассеивания (scatterplot). В Python его можно нарисовать следующим образом:

```
sns.set(style='whitegrid', rc={'figure.figsize' : (10,5)})

sns.scatterplot(x = 'sepal_length', y = 'sepal_width', data = iris)
plt.title('Взаимосвязь измерений чашелистников ирисов')
plt.xlabel('Длина чашелистника')
plt.ylabel('Ширина чашелистника')
```



> Регрессия с одной независимой переменной

Задача одномерной линейной регрессии, по сути, та же, что и у коэффициента корреляции – оценить взаимосвязь между двумя количественными переменными. Различия между ними лежат в технической основе + в линейной регрессии более чётко выражено, какая переменная независимая (НП), а какая зависимая (ЗП). Иными словами, значения какой переменной влияют на другую переменную. НП традиционно лежит на оси X, а ЗП – на оси Y.

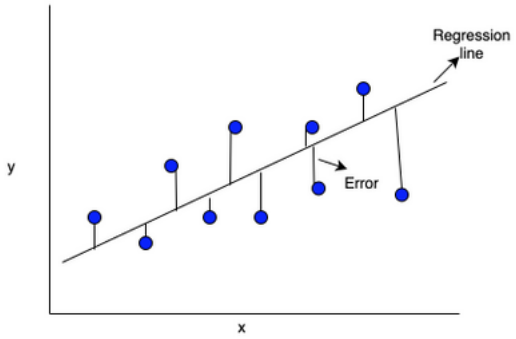
Note: Напоминаем, что это всё ещё ничего не говорит о каузальной взаимосвязи между переменными – о причинно-следственных связях можно судить только путём адекватной организации эксперимента!

Основное уравнение линейной регрессии:

$$Y = b_0 + b_1 X + \epsilon$$

- b_0 – свободный член регрессионного уравнения (Intercept), место, где регрессионная прямая пересекает ось Y. Интерпретация – какое значение принимает зависимая переменная, если независимая переменная равна нулю.
- b_1 – угол наклона регрессионной прямой (slope), отражает направление взаимосвязи между НП и ЗП.
- ϵ – ошибка (остатки уравнения регрессии)

Классический метод нахождения оптимальных параметров уравнения линейной регрессии – **метод наименьших квадратов** (МНК). Он заключается в минимизации суммы квадратов ошибок от регрессионной прямой. **Ошибка** (**остаток**) в данном случае – это разница между индивидуальным значением в выборке (y) и соответствующим ему местом на регрессионной прямой ($\hat{y}$)



Иными словами, этот метод направлен на то, чтобы прямая лежала максимально близко ко всем точкам. Остатки возводятся в квадрат для того, чтобы отрицательные остатки не вычитались из положительных (сумма “сырых” остатков в таком случае равна нулю).

Формулы коэффициентов:

$$b_1 = \frac{\delta_y}{\delta_x} r_{xy}$$

- δ – это стандартное отклонение соответствующей переменной
- r_{xy} – коэффициент корреляции

$$b_0 = \bar{Y} - b_1 \bar{X}$$

$\bar{X}$ и $\bar{Y}$ – средние значения соответствующих переменных.

Проверка гипотез

- H_0 – b_1 равен нулю
- H_1 – b_1 не равен нулю

Наше традиционное допущение: если бы мы повторяли эксперимент бесконечное число раз и H_0 была бы верна, то значения b_1 распределились бы нормальным образом вокруг 0. Соответственно, значимость мы оцениваем с помощью t-распределения, где t-статистика рассчитывается по формуле:

$$t = \frac{b_1}{se}, df = N - 2$$

- se – стандартная ошибка в уравнении линейной регрессии
- df – степени свободы
- N – размер выборки

Также важно знать про **коэффициент детерминации** (R^2) - это квадрат коэффициента корреляции Пирсона и он отображает, в какой степени дисперсия одной переменной обусловлена влиянием другой переменной. Он принимает значения от 0 до 1, и с его помощью можно оценивать качество одномерной регрессионной модели – чем больше R^2, тем лучше. Также его можно считать по следующей формуле:

$$R^2 = 1 - \frac{SS_{res}}{SS_{total}}$$

– сумма квадратов остатков (смотрите конспект про МНК), SS_{total} = общая сумма квадратов (вычитаем каждое наблюдение из среднего значения по ЗП, возводим в квадрат и суммируем). Соответственно, чем меньше SS_{res} относительно SS_{total}, тем ближе R^2 к единице и тем лучше модель.

Условия применения

1. Связь линейна и монотонна
2. Остатки распределены нормальным образом
3. Нет выбросов
4. Дисперсия ЗП однородна на всех уровнях НП (гомоскедастичность)

Приложение, в котором можно поиграться с этими допущениями: https://gallery.shinyapps.io/slr_diag/

```
from scipy import stats
import statsmodels.api as sm

# через scipy (только одномерная)
stats.linregress(x, y)

# через statsmodels (один из вариантов)# Y = одномерный массив с ЗП, X - массив с НП

X = sm.add_constant(X)  # добавить константу, чтобы был свободный член
model = sm.OLS(Y, X)  # говорим модели, что у нас ЗП, а что НП
results = model.fit()  # строим регрессионную прямую
print(results.summary())  # смотрим результат
```

```
# то же самое можно через формулуimport statsmodels.formula.api as smf

results = smf.ols('Y ~ X', data).fit()
print(results.summary())
```

Пример того, как может выглядеть результат регрессии:

```
In [184]: print(model.summary())
                            OLS Regression Results
==============================================================================
Dep. Variable:                      B   R-squared:                       0.000
Model:                            OLS   Adj. R-squared:                 -0.125
Method:                 Least Squares   F-statistic:                 0.0005452
Date:                Tue, 10 Oct 2017   Prob (F-statistic):              0.982
Time:                        11:44:28   Log-Likelihood:                -33.201
No. Observations:                  10   AIC:                             70.40
Df Residuals:                       8   BIC:                             71.01
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [95.0% Conf. Int.]
------------------------------------------------------------------------------
const        100.4481     32.127      3.127      0.014        26.363   174.533
A             -0.0076      0.327     -0.023      0.982        -0.762     0.746
==============================================================================
Omnibus:                        2.865   Durbin-Watson:                   2.352
Prob(Omnibus):                  0.239   Jarque-Bera (JB):                0.994
Skew:                          -0.191   Prob(JB):                        0.608
Kurtosis:                       1.504   Cond. No.                     1.33e+03
------------------------------------------------------------------------------
```

Нас в первую очередь интересует та часть вывода, которая обведена:

- **const** – это свободный член (b_0), в других моделях может выглядеть как Intercept
- Ниже располагается НП
- Значения, связанные с b_0, интерпретировать не нужно, нас интересует именно угол наклона
- **coef** – значение коэффициента, отрицательные значения означают отрицательную взаимосвязь, положительные - положительную
- **std err** – стандартная ошибка
- **t** – t-критерий
- **P > |t|** – p-значение
- Последним идёт 95%-ый доверительный интервал

Для интерпретации результатов достаточно coef и P > |t|.

Помимо этого:

1. В левой верхней части таблицы указаны некоторые формальные характеристики модели (название ЗП, метод, тип ковариации, время создания объекта и т.д.)
2. В правой верхней части таблицы находятся показатели качества модели (в первую очередь нам интересны R^2 и его скорректированная разновидность, о которой будет в следующем уроке)
3. В нижней части таблицы указаны диагностические характеристики модели. В частности, **Omnibus** и **Jarque-Bera** - это два разных теста нормальности остатков (**prob** - их p-значение), **skew** - коэффициент асимметрии, **kurtosis** - коэффициент эксцесса (насколько вытянутое или плоское распределение остатков), **Durbin-Watson** - тест автокорреляции остатков (должен быть между 0 и 4), **Cond. No.** - показатель мультиколлинеарности (должен быть меньше 2, об этом в следующем уроке)

Дополнительно:

Примеры того, как делать диагностику модели
тестами: https://www.statsmodels.org/dev/examples/notebooks/generated/regression_diagnostics.html

Пример того, как делать визуальную диагностику: https://zhiyzuo.github.io/Linear-Regression-Diagnostic-in-Python/

> Как рисовать

График фактически идентичен тому, который строится при обычной корреляции, но в случае линейной регрессии принято ещё рисовать соответствующую прямую.

```
sns.set(style='whitegrid', rc={'figure.figsize' : (10,5)})
sns.regplot(x = 'sepal_length', y = 'sepal_width', data = iris)
plt.title('Взаимосвязь измерений чашелистников ирисов')
plt.xlabel('Длина чашелистника')
plt.ylabel('Ширина чашелистника')
```