

▼ Импорт необходимых библиотек

```
import pandas as pd
import numpy as np
import scipy.stats as sts
import math
from scipy.stats import chi2_contingency
from scipy.stats import ttest_ind

import matplotlib.pyplot as plt
import seaborn as sns

def prob_func (n, k):
    return (1/n)**k * ((n-1)/n)**(10-k)

# Построить функцию правдоподобия для n = 1 до 100
n_vals = np.arange (1, 101)
prav_vals = [prob_func (n, 10) for n in n_vals]
plt.plot (n_vals, prav_vals)
plt.xlabel('Общее количество таксистов, n')
plt.ylabel('Функция правдоподобия')
plt.show()

# Найдите оценку n, которая максимизирует функцию правдоподобия
n_max_likelihood = n_vals[np.argmax(prob_func)]
print(f'Оценка n, которая максимизирует функцию правдоподобия, равна {n_max_likelihood}.')
```



▼ Задача 3

▼ a)

```
primer = np.random.exponential(scale =1, size = (10000,20))
```

```
sr = primer.mean(axis=1)
```

```
z = sts.norm.isf(0.025)
```

```
z
```

```
1.9599639845400545
```

Построим ас. доверительный интервал для математического ожидания

```
theoretical = np.array([sr - z * np.sqrt(primer.var(ddof = 1, axis = 1)/20), sr + z*np.sqrt(primer.var(ddof = 1, axis = 1)
hashes = 0
for item, jtem in theoretical:
    if item<1 and 1<jtem:
        hashes+=1
hashes/10000

0.9045
```

это вероятность накрытия с ас. доверительным интервалом

```
np.random.seed(192)
n_in=0
for i in range(10000):
    sample = primer[i]
    sample_enh = np.random.choice(sample, size = (10000,20))
    up = sample_enh.mean(axis = 1)
    upper=np.percentile(up, 97.5)
    lower=np.percentile(up, 2.5)
    if lower<1<upper:
        n_in+=1
n_in/10000

0.903
```

ура снова получили что-то похожее

```
np.random.seed(156)
n_in=0
for i in range(10000):
    sample=primer[i]
```

```

sample_enh=np.random.choice(sample,size=(10000,20))
up=(sample_enh.mean(axis = 1) - sample.mean())/(np.sqrt(sample_enh.var(ddof=1,axis=1)/20))
upper=np.percentile(up, 97.5)
lower=np.percentile(up, 2.5)
if sample.mean()-upper*np.sqrt(sample.var(ddof=1)/20)<1<sample.mean()-lower*np.sqrt(sample.var(ddof=1))/20:
    n_in+= 1
n_in/10000

0.6865

```

Получили тоже высокое значение, значит вероятность порядочная

▼ Задача 4

▼ a)

Сначала приведем данные к удобному виду

```

df_raw = pd.read_csv('/content/22-23_hse_probability - Exam (1).csv')
df_raw = df_raw[['Last name', 'Unnamed: 72']]
df_raw = df_raw[(df_raw != 0).all(axis = 1)]
df = df_raw.dropna()

df.rename({'Unnamed: 72': 'Exam score'}, axis=1, inplace=True)
df = df.reset_index(drop=True)

sogl = ['б', 'в', 'г', 'д', 'ж', 'з', 'й', 'к', 'л', 'м', 'н', 'п', 'р', 'с', 'т', 'ф', 'х', 'ц', 'ч', 'ш', 'щ']

df['Sogl?'] = df['Last name'].apply(lambda x: 1 if x[0].lower() in sogl else 0)
df

```

<ipython-input-73-4e51e91dce9a>:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: <https://pandas.pydata.org/pandas-docs/stable/user>
df.rename({'Unnamed: 72': 'Exam score'}, axis=1, inplace=True)

	Last name	Exam score	Sogl?
0	Репенкова	16.0	1
1	Сафина	19.0	1
2	Сидоров	26.0	1
3	Солоухин	21.0	1
4	Старощук	22.0	1
...
291	Петрова	10.0	1
292	Полищук	25.0	1
293	Савенкова	4.0	1
294	Сенников	19.0	1

```
sg = df[df['Sogl?']==1]['Exam score']
gl = df[df['Sogl?']==0]['Exam score']
```

```
sg_sr = sg.mean()
sg_var = sg.var()
sg_len = len(sg)
```

```
gl_sr = gl.mean()
```

```

gl_var = gl.var()
gl_len = len(gl)

yel_stat = (sg_sr - gl_sr) / (math.sqrt(gl_var/gl_len + sg_var/sg_len))
yel_stat

0.8646536949457263

d = ((gl_var/gl_len + sg_var/sg_len)**2) / (gl_var**2 / (gl_len**2 * (gl_len-1)) + sg_var**2 / (sg_len**2 * (sg_len-1)))
d

54.91646258007764

t_crit = sts.t.ppf(0.975, d)
t_crit

2.0041133054512987

```

Заметим, что наблюдаемое значение сильно меньше критического значения, значит начальная гипотеза не отвергается

▼ 6)

```

n_iterations = 10000

bootstrap_samples1 = np.random.choice(gl, size=(n_iterations, gl_len), replace=True)
bootstrap_samples2 = np.random.choice(sg, size=(n_iterations, sg_len), replace=True)

bootstrap_sr = np.mean(bootstrap_samples1, axis = 1) - np.mean(bootstrap_samples2, axis = 1)
p_value = np.sum(np.abs(bootstrap_sr) >= np.abs((gl_sr) - sg_sr)) / n_iterations

p_value

```

0.543

P-value слишком большое, значит ни при каком адекватном уровне значимости гипотеза не отвергается

▼ В)

```
t_stat = sg_sr - gl_sr
t_stat /= math.sqrt(sg_var / sg_len + gl_var / gl_len)

n = 10000
bootstrap_tstats = np.zeros(n)
for i in range (n):
    resample1 = np.random.choice(sg, size=sg_len, replace=True)
    resample2 = np.random.choice(gl, size=gl_len, replace=True)

    resample_tstat = np.mean(resample1) - np.mean(resample2)
    resample_tstat /= np.sqrt((np.var(resample1, ddof=1) / len(resample1)) + (np.var(resample2, ddof=1) / len(resample2)))
    bootstrap_tstats[i] = resample_tstat

p_value = np.sum (np.abs (bootstrap_tstats) >= np.abs (t_stat)) / n

p_value

0.5548
```

P-value слишком большое, значит ни при каком адекватном уровне значимости гипотеза не отвергается

▼ Г)

```
obs_diff = np.mean (sg) - np.mean (gl)
```

```

enh = np.concatenate([sg, gl])
n = 10000
permuted_diffs = np.zeros(n)
for i in range(n):
    np.random.shuffle(enh)
    permuted_diffs[i] = np.mean (enh [: len (sg)]) - np.mean (enh [len (sg):])

p_value = np.sum (np.abs (permuted_diffs) >= np.abs (obs_diff))/n
p_value

0.3659

```

P-value слишком большое, значит ни при каком адекватном уровне значимости гипотеза не отвергается

▼ Задача 5

```

mediana = df['Exam score'].median()

sg_mormed = sg[sg>=mediana].count()
sg_lesmed = sg[sg<mediana].count()
gl_mormed = gl[gl>=mediana].count()
gl_lesmed = gl[gl<mediana].count()

p_gl = gl_mormed / (gl_mormed + gl_lesmed)
p_sg = sg_mormed / (sg_mormed + sg_lesmed)

sg_mormed, sg_lesmed, gl_mormed, gl_lesmed, p_gl, p_sg

(145, 108, 21, 22, 0.4883720930232558, 0.5731225296442688)

```

▼ a)


```

a = sg_mormed
b = sg_lesmed
c = gl_mormed
d = gl_lesmed

OR = (a*d)/(b*c)
SE = np.sqrt(1/a + 1/b + 1/c + 1/d)

lower = np.exp(np.log(OR)-1.96*SE)
upper = np.exp(np.log(OR)+1.96*SE)
CI = [lower, upper]

sta = (a*d - b*c)**2 * (a+b+c+d) / ((a+b)*(c+d)*(a+c)*(b+d))
p_value = 1 - sts.chi2.cdf(sta, 1)
CI, p_value

([0.7359044730930567, 2.688275802603522], 0.3005395652910908)

```

P-value слишком большое, значит ни при каком адекватном уровне значимости гипотеза не отвергается, 1 входит в доверительный интервал

▼ б)

```

RR = (a*(b+d)) / (b*(a+c))
RR_enh = np.log(RR)

SE_RR = math.sqrt(c/(a*(a+c)) + d/(b*(b+d)))

CI_RR = [math.exp(RR_enh - SE_RR * 1.96), math.exp(RR_enh + SE_RR * 1.96)]

stes = (a + b + c + d) * ((a*d - b*c)** 2 / ((a + b)* (c + d) * (a + c)* (b + d)))
p_value = 1 - sts.chi2.cdf(stes, 1)
CI_RR, p_value

```

```
([0.954415742940724, 1.158300990405983], 0.3005395652910908)
```

p-value опять же довольно велик, значение входит в интервал, а значит гипотезу нулевую не отвергаем

▼ Задача 6

▼ a)

df

```
df['dlina']=df['Last name'].apply(lambda x:len(x))
df
```

	Last name	Exam score	Sogl?	dlina	
0	Репенкова	16.0	1	9	
1	Сафина	19.0	1	6	
2	Сидоров	26.0	1	7	
3	Солоухин	21.0	1	8	
4	Старощук	22.0	1	8	
...	
291	Петрова	10.0	1	7	
292	Полищук	25.0	1	7	
293	Савенкова	4.0	1	9	
294	Сенников	19.0	1	8	
295	Эшмеев	16.0	0	6	

Если оценить с помощью первого начального момента, то получим, что $E(Y) = Y_{\text{среднее}}$

Поэтому оценка Y это $Y_{\text{среднее}}/F(\text{оценка за экзамен})_{\text{среднее}}$

То есть:

```
teta = df['Exam score'].mean()/df['dlina'].mean()
teta
```

2.3109965635738834

```
df.corr()
```

<ipython-input-88-2f6f6606aa2c>:1: FutureWarning: The default value of numeric_only in D

```
df.corr()
```

	Exam score	Sogl?	dlina
Exam score	1.000000	0.053219	0.033327
Sogl?	0.053219	1.000000	0.072414

Корреляция где-то 3%, то есть очень слабая, значит эффекта особо нет

▼ 6)

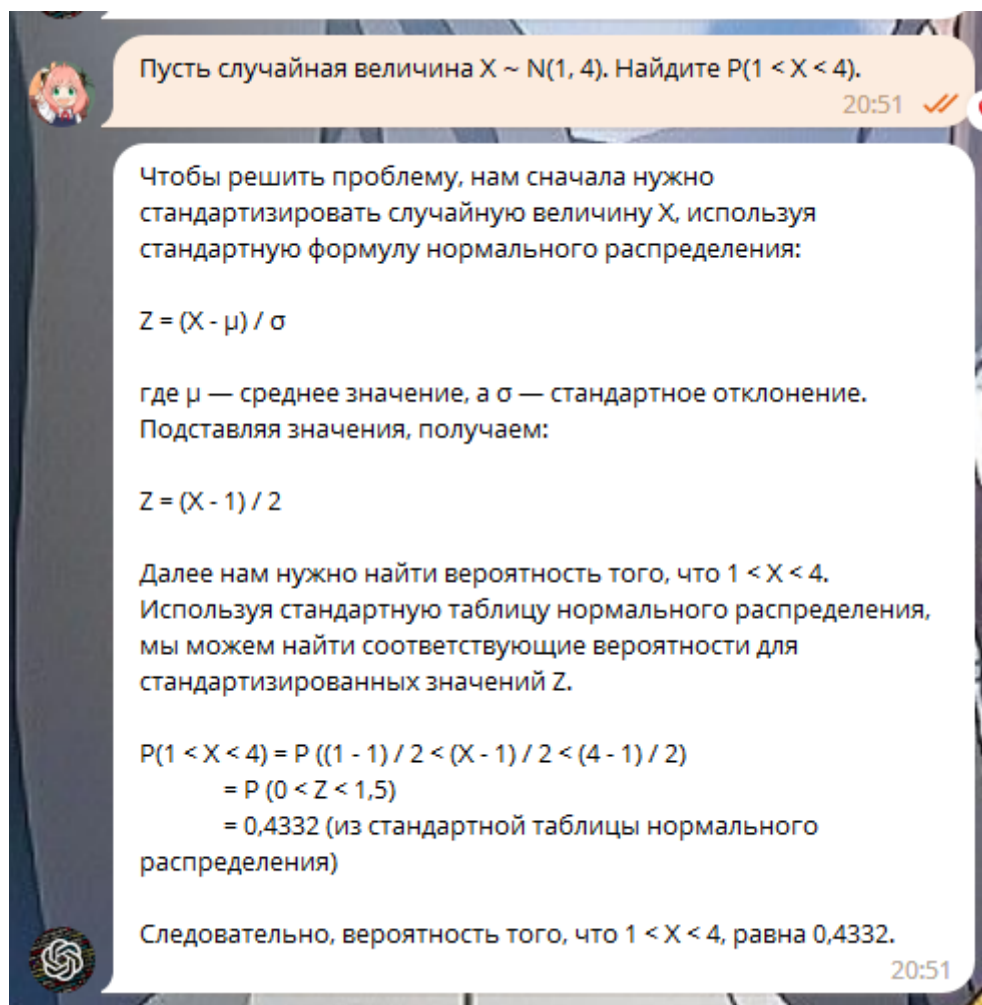
```
np.random.seed(8800)
permutations = []
for i in range(10000):
    y_enh = np.random.permutation(df['Exam score'])
    pipa = np.corrcoef(y_enh, df['dlina'])[0,1]
    permutations.append(pipa)
```

```
p_value = 2*min(np.sum(permutations[permutations>0.033327])/10000, np.sum(permutations[permutations<=0.033327])/10000)
p_value
```

```
lower = np.percentile(permutations, 2.5)
upper = np.percentile(permutations, 97.5)
lower, upper
```

Заметим, что значение входит в интервал, а p_value достаточно большой, значит гипотеза не отвергается

▼ Задача 7



Пусть случайная величина $X \sim N(1, 4)$. Найдите $P(1 < X < 4)$. 20:51 ✓

Чтобы решить проблему, нам сначала нужно стандартизировать случайную величину X , используя стандартную формулу нормального распределения:

$$Z = (X - \mu) / \sigma$$

где μ — среднее значение, а σ — стандартное отклонение. Подставляя значения, получаем:

$$Z = (X - 1) / 2$$

Далее нам нужно найти вероятность того, что $1 < X < 4$. Используя стандартную таблицу нормального распределения, мы можем найти соответствующие вероятности для стандартизированных значений Z .

$$\begin{aligned} P(1 < X < 4) &= P((1 - 1) / 2 < (X - 1) / 2 < (4 - 1) / 2) \\ &= P(0 < Z < 1,5) \\ &= 0,4332 \text{ (из стандартной таблицы нормального распределения)} \end{aligned}$$

Следовательно, вероятность того, что $1 < X < 4$, равна 0,4332. 20:51

▼ Задача 8

Самым полезным ресурсом за время учебы, пожалуй, были видосики с канала 3Blue1Brown. Некоторые видосики скидывал Борис Борисович, некоторые смотрел сам. Через них понятнее становилась связь между изучаемой теорией и какой-то практикой/бытовым применением. Например, полностью я понял значение и смысл ЦПТ и нормальности распределения величин только после упомянутых видео. Отдельно еще выделить можно видео о том, откуда в функции плотности нормального распределения число Π :).

В общем, много знаний и ассоциаций оттуда были получены, большое спасибо!

[Платные продукты Colab](#) - [Отменить подписку](#)

❗ 1 сек. выполнено в 20:54

● ✕