Лабораторная работа №7

Вариант 7

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import scipy.stats as sts
import seaborn as sns
import math

%matplotlib inline
plt.style.use('fast')
```

```
def corr(x, y):
    Pearson = sts.pearsonr(x, y)
    Spearman = sts.spearmanr(x, y)
    Kendall = sts.spearmanr(x, y)

    print(f"""Значение выборочного коэффициента корреляции Пирсона: {Pearson.statistic:.02f})
Значение p_value: {Pearson.pvalue:.06f}

Значение рангового коэффициента корреляции Спирмена: {Spearman.statistic:.02f})
Значение p_value: {Spearman.pvalue:.06f}

Значение рангового коэффициента корреляции Кендалла : {Kendall.statistic:.02f})
Значение p_value: {Kendall.pvalue:.06f}""")
```

Задача 1

```
df = pd.read_csv("Lab_07/Вариант 7.1.csv", sep=';')
labels = ["X, пробег автомобиля", "Y, стоимость  ежемесячного тех. обслуживания"]
x, y = df["X"], df["Y"]
df
```

	X	Y
0	8.161	20.643
1	16.206	39.540
2	13.260	25.385
3	17.592	31.199
4	16.015	32.735
...	...	...
95	3.809	11.297
96	13.137	15.771
97	18.851	22.425
98	5.867	9.694
99	13.563	22.380

100 rows × 2 columns

```
plt.plot(x, y, '+g')
plt.xlabel(labels[0])
plt.ylabel(labels[1])
plt.title("Диаграмма рассеяния")
```

```
Text(0.5, 1.0, 'Диаграмма рассеяния')
```

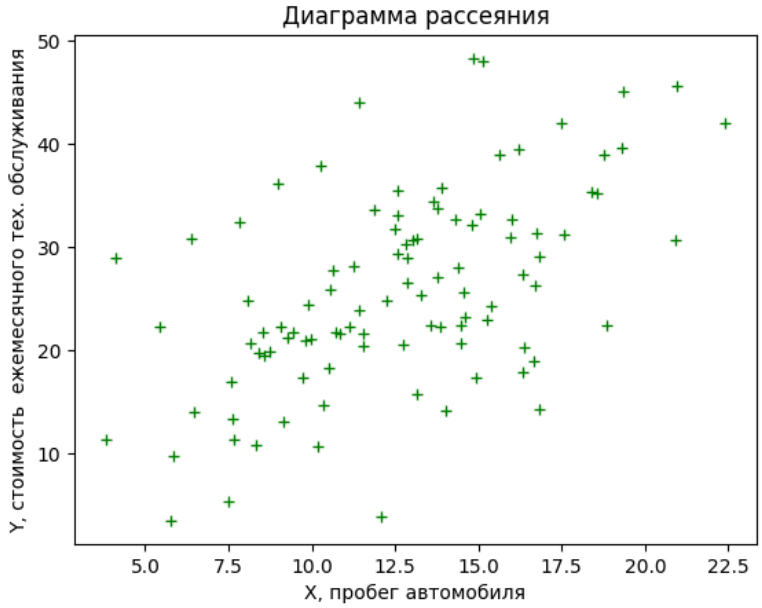
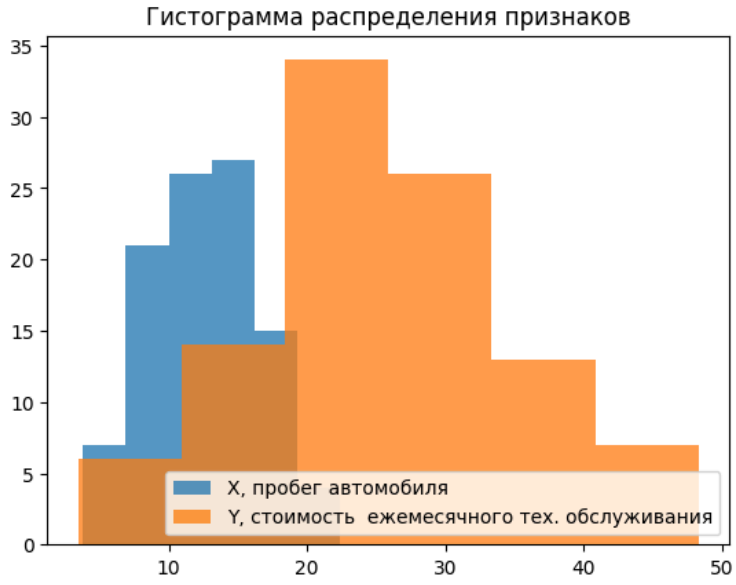


Диаграмма имеет вид облака точек формы прямой. Судя по характеру диаграммы рассеяния можно предположить о слабой прямой линейной зависимости признаков X и Y. Т.к. плотность точек выше, вдоль прямой вида y=kx+b, при этом точки достоточно сильно рассеяны, что говорит о слабой корреляции.

```
plt.hist(x, bins=int(math.log2(len(x))), density=False, alpha=0.75)
plt.hist(y, bins=int(math.log2(len(y))), density=False, alpha=0.75)
plt.title("Гистограмма распределения признаков")
plt.legend(labels, loc="lower right")
```

```
<matplotlib.legend.Legend at 0x2c397b03b50>
```



Т.к. предполагаемая зависимость линейная, можно использовать коэффициет корреляции Пирсона.

```
corr(x, y)
```

```
Значение выборочного коэффициента корреляции Пирсона: 0.55)
Значение p_value: 0.000000

Значение рангового коэффициента корреляции Спирмена: 0.53)
Значение p_value: 0.000000

Значение рангового коэффициента корреляции Кендалла : 0.53)
Значение p_value: 0.000000
```

Коэффициенты корреляции входят в промежуток [0.4; 0.6], что говорит об умеренной линейной зависимости признаков. Коэффициенты корреляции положительны, что говорит об прямой зависимости признаков.

Нулевая гиппотеза H0 - корреляция в выборке статистически не значима (значение коэффициента корреляции получено случайно, корреляция отсутствует на генеральной совокупности). Для каждого коэффициента корреляции p_value << уровня занчимости alpha=0.05, следовательно, гиппотеза H0 о статистической незначимости отвергается.

Предположения частично подтвердились.

Вывод: стоимость ежемесячного технического обслуживания автомобиля прямо умеренно линейно зависит от пробега автомобиля.

Задача 2

```
df = pd.read_csv("Lab_07/Вариант 7.2.csv", sep=';')
labels = ["X", "Y"]
x, y = df["X"], df["Y"]
df
```

	X	Y
0	1	79
1	5	95
2	1	84
3	3	88
4	2	53
5	2	87
6	0	83
7	1	57
8	5	50
9	7	82
10	6	71
11	8	56
12	6	93
13	5	90
14	8	53
15	7	53
16	1	85
17	7	81

```
plt.plot(x, y, '+g')
plt.xlabel(labels[0])
plt.ylabel(labels[1])
plt.title("Диаграмма рассеяния")
```

```
Text(0.5, 1.0, 'Диаграмма рассеяния')
```

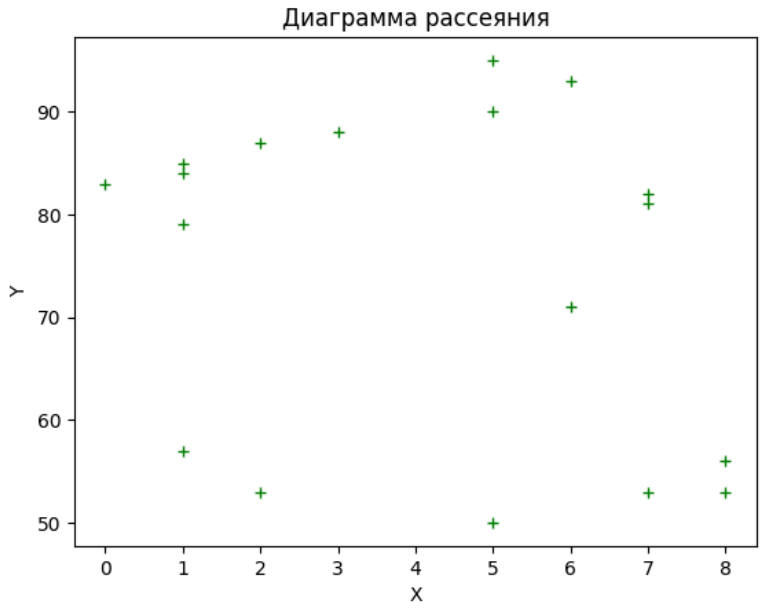
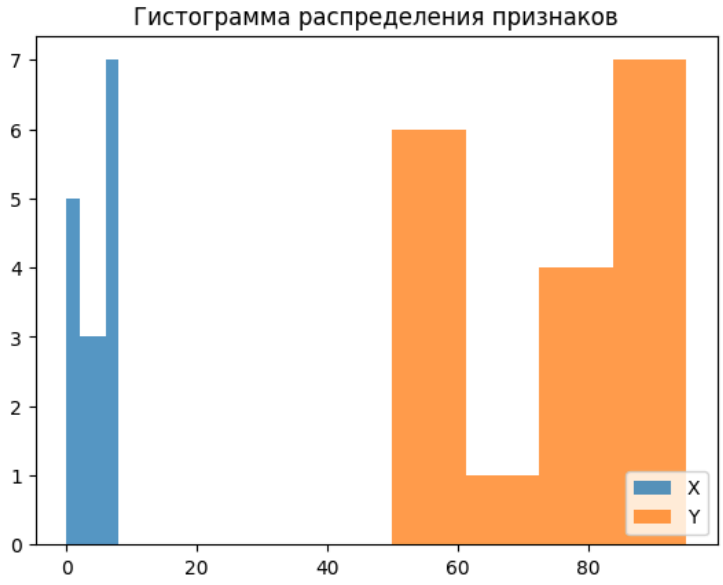


Диаграмма рассеяния выглядит как облако точек формы окружности. Судя по характеру диаграммы рассеяния можно утверждать о полной отсутствии зависимости между признаками.

```
plt.hist(x, bins=int(math.log2(len(x))), density=False, alpha=0.75)
plt.hist(y, bins=int(math.log2(len(y))), density=False, alpha=0.75)
plt.title("Гистограмма распределения признаков")
plt.legend(labels, loc="lower right")
```

```
<matplotlib.legend.Legend at 0x2c3973c5cd0>
```




```
corr(x, y)
```

```
Значение выборочного коэффициента корреляции Пирсона: -0.25)
Значение p_value: 0.310971

Значение рангового коэффициента корреляции Спирмена: -0.26)
Значение p_value: 0.288816

Значение рангового коэффициента корреляции Кендалла : -0.26)
Значение p_value: 0.288816
```

Коэффициенты корреляции входят в промежуток [-0.4; -0.2], что говорит очень слабой линейной зависимости признаков. Коэффициенты корреляции отрицательны, что говорит об обратной зависимости признаков.

Нулевая гиппотеза H0 - корреляция в выборке статистически не значима (значение коэффициента корреляции получено случайно, корреляция отсутствует на генеральной совокупности). Для каждого коэффициента корреляции p_value >> уровня занчимости alpha=0.05, следовательно, гиппотеза H0 о статистической незначимости принимается.

Предположения полностью подтвердились.

Вывод: признаки X и Y не зависят друг от друга.

Задача 3

```
df = pd.read_csv("Lab_07/Вариант 7.3.csv", sep=',')
labels = ["Ответы на 1 вопрос", "Ответы на второй вопрос"]
x, y = df["Вопрос 1"], df["Вопрос 2"]
df
```

	Вопрос 1	Вопрос 2
0	B	d
1	B	c
2	B	a
3	A	c
4	A	d
...	...	...
60	B	c
61	A	b
62	B	d
63	A	b
64	A	b

65 rows × 2 columns

```
cross_tab = pd.crosstab(x, y, margins=True)
cross_tab
```

Вопрос 2	a	b	c	d	All
Вопрос 1					
A	5	21	6	6	38
B	7	0	10	10	27
All	12	21	16	16	65

Критерий $\chi 2$ считается надежным, только если в таблице сопряженности не слишком много клеток с небольшими частотами (количество клеток с частотами менее 5 не должно превышать 20%). Только одно значение в таблице меньше 5, следовательно, критерий $\chi 2$ можно считать надёжным.

Исследуемые признаки качественные. В ходе эксперимента испытуемым был предложен тест, в котором первый вопрос был направлен на изучение признака X, второй – на изучение признака Y.

В паре ответов (i, j) на два вопроса отвечал один и тот же человек. Следовательно, группы для признаков зависимы - по сути у нас одна группа испытуемых. Использование критерия хи-квадрат является неуместным, т.к. он применяется для двух и более независимых групп.

Вывод: использование критерия хи-квадрат для данной выборки является неуместным.

Задача 4

```
df = pd.read_csv("Lab_07/Данные_клиентов.txt", sep="\t")
labels = ["Возраст", "Продолжительность разговора, сек."]
df = df[[labels[0], labels[1]]]
df.columns = ["x", "y"]
df
```

	x	y
0	56	261
1	57	149
2	37	226
3	40	151
4	56	307
...	...	...
41183	73	334
41184	46	383
41185	56	189
41186	44	442
41187	74	239

41188 rows × 2 columns

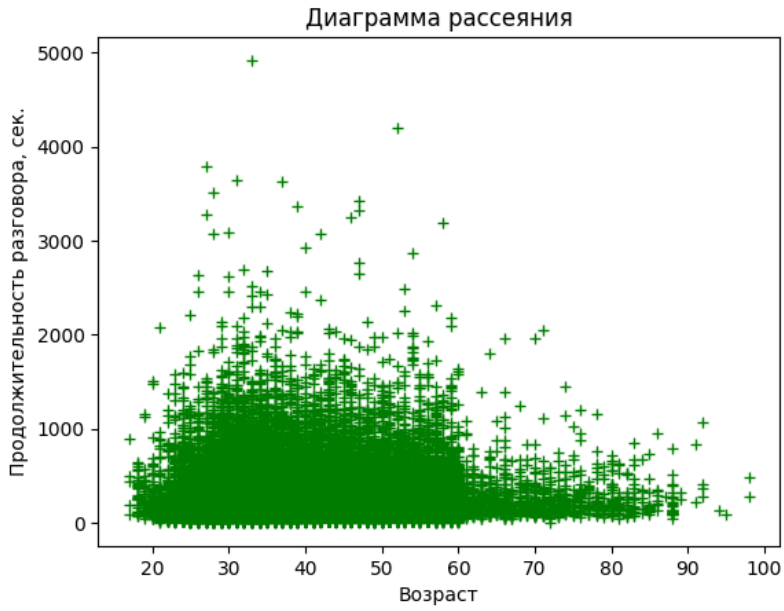
```
print(f"Среди значений x есть значения NaN: {True in df.x.isnull()}")
print(f"Среди значений y есть значения NaN: {True in df.y.isnull()}")
```

```
Среди значений x есть значения NaN: False
Среди значений y есть значения NaN: False
```

Среди значений исследуемых признаков нет значений NaN. Если бы они были - пришлось бы либо их убрать, либо заменить медианным значением для данного признака.

```
plt.plot(df.x, df.y, '+g')
plt.xlabel(labels[0])
plt.ylabel(labels[1])
plt.title("Диаграмма рассеяния")
```

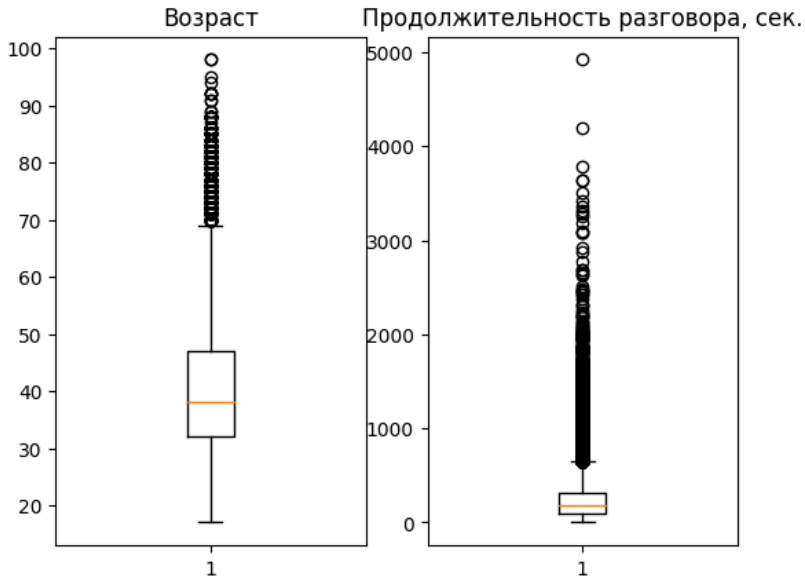
```
Text(0.5, 1.0, 'Диаграмма рассеяния')
```



Среди значений можно наблюдать некоторое число выбросов. Перед анализом зависимостей нужно удалить выбросы, т.к. коэффициенты корреляции чувствительны к выбросам. Для определения выбросов воспользуемся диаграммой "ящик с усами"

```
fig, ax = plt.subplots(1, 2)
ax[0].boxplot(df.x,)
ax[1].boxplot(df.y,)
ax[0].set_title(labels[0])
ax[1].set_title(labels[1])
```

```
Text(0.5, 1.0, 'Продолжительность разговора, сек.')
```



Избавимся от строк, где какой-либо из признаков выходит за границы "усов".

```
x_q1 = np.quantile(df.x, 0.25)
x_q2 = np.quantile(df.x, 0.75)
y_q1 = np.quantile(df.y, 0.25)
y_q2 = np.quantile(df.y, 0.75)

q = x_q2 + 1.5*(x_q2 - x_q1), y_q2 + 1.5*(y_q2 - y_q1)

print(q)
f_df = df[(df.x <= q[0]) & (df.y <= q[1])]

plt.plot(f_df.x, f_df.y, '+y')
plt.xlabel(labels[0])
plt.ylabel(labels[1])
plt.title("Диаграмма рассеяния исправленных данных")
```

```
(69.5, 644.5)
Text(0.5, 1.0, 'Диаграмма рассеяния исправленных данных')
```

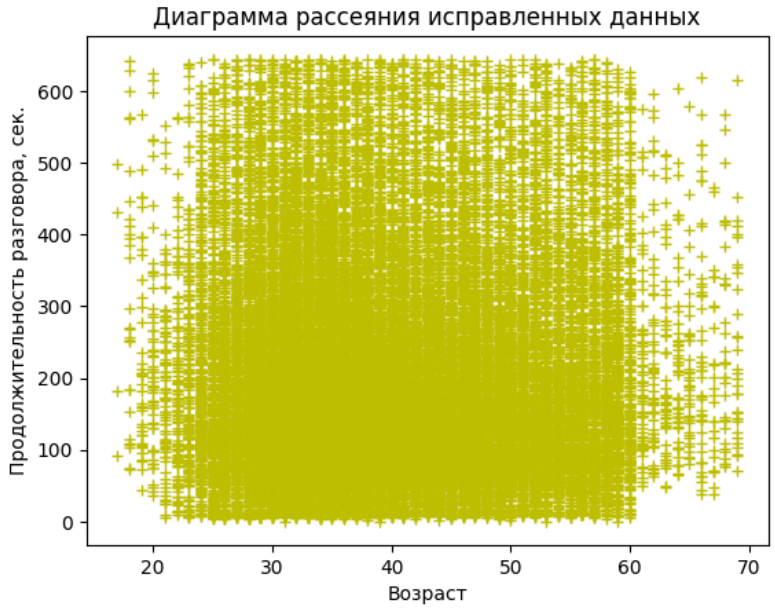


Диаграмма рассеяния имеет вид облака точек в форме прямоугольника. Такой характер диаграммы нам говорит, что выборка близка к декардову произведению - среди всех возрастов равномерно представлены люди, говорящие по телефону с сотрудниками банка любое конкретно выбранное количество времени. На основании этого можно точно утверждать, что признаки независимы.

```
corr(f_df.x, f_df.y)
```

```
Значение выборочного коэффициента корреляции Пирсона: 0.00)
Значение p_value: 0.449867

Значение рангового коэффициента корреляции Спирмена: -0.00)
Значение p_value: 0.560508

Значение рангового коэффициента корреляции Кендалла : -0.00)
Значение p_value: 0.560508
```

Коэффициенты корреляции входят в промежуток [-0.1; 0.1], что говорит об отсутствии зависимости признаков.

Нулевая гиппотеза H0 - корреляция в выборке статистически не значима (значение коэффициента корреляции получено случайно, корреляция отсутствует на генеральной совокупности). Для каждого коэффициента корреляции p_value >> уровня занчимости alpha=0.05, следовательно, гиппотеза H0 о статистической незначимости принимается.

Предположения полностью подтвердились.

Вывод: время разговора клиента с сотрудником банка не зависит от возраста.

Задача 5

```
df = pd.read_csv("Lab_07/Данные_клиентов.txt", sep="\t")
del(df["ID записи"])
```

Удалим колонку неинформативную колонку ID. Найдём колонки, где пропущеных значений > 50%.

```
print(f"количество колонок: {len(df.columns)}")
print()
del_cols = []
for col_name in df.columns:
    null_sum = df[col_name].isnull().sum()
    null_procent = null_sum / len(df[col_name]) * 100
    if null_procent > 50:
        print(f"""Колонка: {col_name}
Пропущено значений (абсолютно): {null_sum}
Пропущено значений (относительно): {null_procent:.02f} %
""")
        del_cols.append(col_name)


df = df.drop(del_cols, axis=1)
```

```
количество колонок: 21

Колонка: Последний контакт, дни
Пропущено значений (абсолютно): 39673
Пропущено значений (относительно): 96.32 %
```

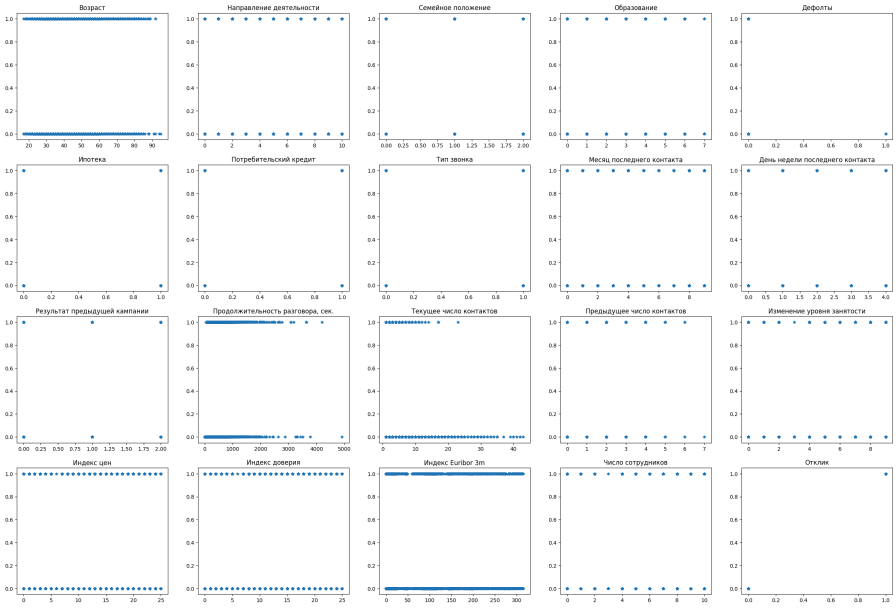
В колонке "Последний контакт, дни" 96% пропущеных значений соответственно. Эти колонка не подойдёт для анализа, такую колонку можно удалить. Построим диаграммы рассеяния целевого признака в зависимости от оставшихся колонок.

Некоторые колонки имеют строковые значения, для удобства переведём их в числовой тип. Также удалим пропуски.

```
df = df.dropna()

for col_name in df.columns:
    if type(df[col_name][0]) == str:

        variations = pd.unique(df[col_name])
        d = dict([(j, i) for i, j in enumerate(variations)])
        df[col_name] = [d[i] for i in df[col_name]]
```

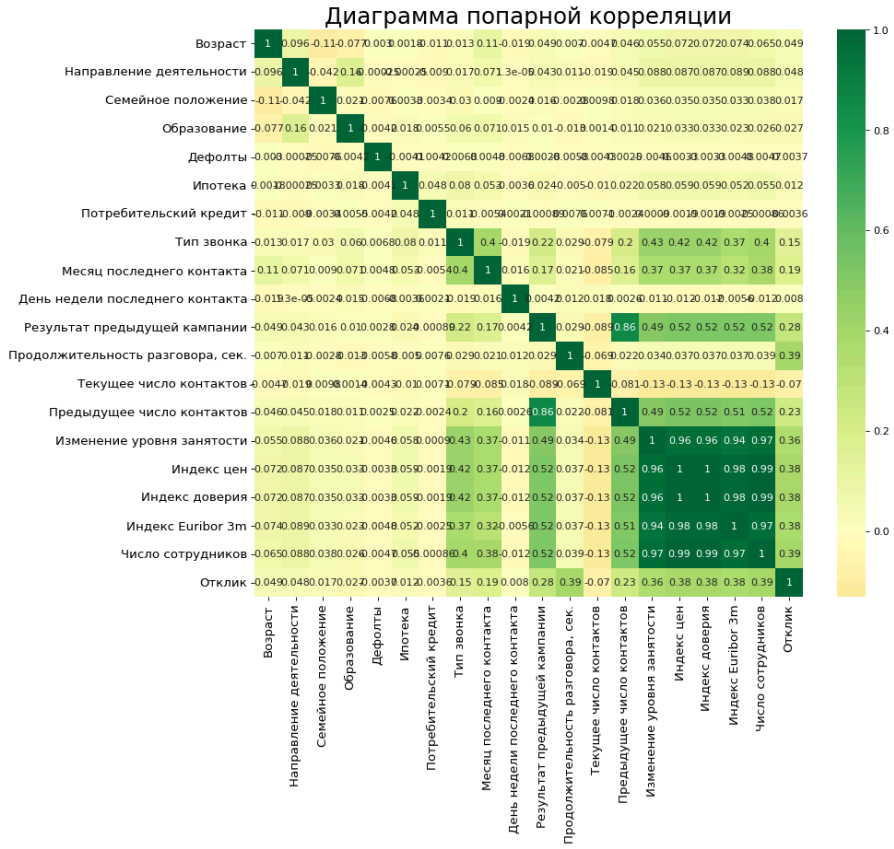
Посмотрим на зависимости целевого признака от всех остальных.

```
fig, ax = plt.subplots(4, 5)
fig.set_size_inches(30, 20)
cols = df.columns
y = df["Отклик"]

for i in range(4):
    for j in range(5):
        ax[i][j].set_title(cols[i*5 + j])
        x = df[cols[i*5 + j]]

        ax[i][j].plot(x, y, "*")
```



Можно заметить, что при текущем количестве контактов больше 20, отклик отрицательный (можно хорошо использовать этот признак в модели случайного леса). Этот признак может иметь высокую кореляцию относительно целевого. Построим диаграмму попарной корреляции, чтобы определить пары признаков с ковариацией.

```
plt.figure(figsize=(12,10), dpi= 80)
sns.heatmap(df.corr(), xticklabels=df.corr().columns, yticklabels=df.corr().columns, cmap='RdYlGn', center=0, annot=True)

plt.title('Диаграмма попарной корреляции', fontsize=22)
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)
plt.show()
```



Сразу группа признаков обладает ковариацией - эти признаки нам не подходят т.к. они будут создавать шум, оставим из них только один - имеющий самую большую корреляцию с целевым: "число сотрудников".

```
del_cols = ["Индекс Euribor 3m", "Индекс доверия", "Индекс цен", "Изменение уровня занятости", "Предыдущее число контактов", "Результ
df = df.drop(del_cols, axis=1)
df.columns
```

```
Index(['Возраст', 'Направление деятельности', 'Семейное положение',
       'Образование', 'Дефолты', 'Ипотека', 'Потребительский кредит',
       'День недели последнего контакта', 'Продолжительность разговора, сек.',
       'Текущее число контактов', 'Число сотрудников', 'Отклик'],
      dtype='object')
```

Теперь можно проверить попарную корреляцию оставшихся признаков с целевым, импользуя коэффициент V Краммера. Выбран коэффициент Краммера, т.к. целевой признак является категориальным.

```
result = []
for col_name in df.columns[:-1]:
    x = df[col_name]
    cross_tab = np.array(pd.crosstab(x, y, margins=False))

    X2 = sts. chi2_contingency (cross_tab, correction= False )[0]
    n = np.sum (cross_tab)
    minDim = min(cross_tab. shape )-1

    V = np.sqrt ((X2/n) / minDim)
```