Содержание

Отток клиентов

Из банка стали уходить клиенты. Банковские маркетологи посчитали: сохранять текущих клиентов дешевле, чем привлекать новых.

Нужно спрогнозировать, уйдёт клиент из банка в ближайшее время или нет. Нам предоставлены исторические данные о поведении клиентов и расторжении договоров с банком.

Необходимо построить модель с предельно большим значением *F1*-меры. Для успешного выполнения задачи, нужно довести метрику до 0.59.

Дополнительно измерим *AUC-ROC*, сравним её значение с *F1*-мерой.

Источник данных: https://www.kaggle.com/barelydedicated/bank-customer-churn-modeling

Цель исследования — спрогнозировать, уйдёт клиент из банка в ближайшее время или нет и найти модель с предельно большим значением F1-меры (не меньше 0.59)

Данные о поведении клиентов получим из файла ***Churn.csv***. В нем представлены исторические данные о поведении клиентов и расторжении договоров с банком

Исследование пройдёт в четыре этапа:

- Подготовка данных
- Исследование задачи
- Борьба с дисбалансом
- Тестирование модели

Подготовка данных

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.utils import shuffle

from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import f1_score, roc_auc_score, confusion_matrix, roc_curve


r_state = 12345

data = pd.read_csv('datasets/Churn.csv')
```

Изучим данные

In [2]:
```
data.info()
display(data.sample(5, random_state=r_state))

print('Количество пропусков по столбцам:')
print()

for col in data.columns:
    nmv = data[col].isna().sum()
    pmv = nmv/len(data)
    print('{} - {} шт. - {}%'.format(col, nmv, round(pmv*100, 2)))

print()
print('Количество явных дубликатов:', data.duplicated().sum())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 14 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   RowNumber        10000 non-null  int64  
 1   CustomerId       10000 non-null  int64  
 2   Surname          10000 non-null  object 
 3   CreditScore      10000 non-null  int64  
 4   Geography        10000 non-null  object 
 5   Gender           10000 non-null  object 
 6   Age              10000 non-null  int64  
 7   Tenure           9091 non-null   float64
 8   Balance          10000 non-null  float64
 9   NumOfProducts    10000 non-null  int64  
 10  HasCrCard        10000 non-null  int64  
 11  IsActiveMember   10000 non-null  int64  
 12  EstimatedSalary  10000 non-null  float64
 13  Exited           10000 non-null  int64  
dtypes: float64(3), int64(8), object(3)
memory usage: 1.1+ MB
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	Estimate
7867	7868	15697201	Yocum	640	Spain	Female	46	3.0	0.00	1	1	1	150
1402	1403	15613282	Vorobyova	757	France	Male	29	8.0	130306.49	1	1	0	7
8606	8607	15694581	Rawlings	807	Spain	Male	42	5.0	0.00	2	1	1	7
8885	8886	15815125	Michael	668	Spain	Male	45	4.0	102486.21	2	1	1	158
6494	6495	15752846	Pinto	699	France	Male	28	7.0	0.00	2	1	1	2

Количество пропусков по столбцам:

RowNumber - 0 шт. - 0.0%
CustomerId - 0 шт. - 0.0%
Surname - 0 шт. - 0.0%
CreditScore - 0 шт. - 0.0%
Geography - 0 шт. - 0.0%
Gender - 0 шт. - 0.0%
Age - 0 шт. - 0.0%
Tenure - 909 шт. - 9.09%
Balance - 0 шт. - 0.0%
NumOfProducts - 0 шт. - 0.0%
HasCrCard - 0 шт. - 0.0%
IsActiveMember - 0 шт. - 0.0%
EstimatedSalary - 0 шт. - 0.0%
Exited - 0 шт. - 0.0%

Количество явных дубликатов: 0

Признаки

- RowNumber — индекс строки в данных
- CustomerId — уникальный идентификатор клиента
- Surname — фамилия
- CreditScore — кредитный рейтинг
- Geography — страна проживания
- Gender — пол
- Age — возраст
- Tenure — сколько лет человек является клиентом банка
- Balance — баланс на счёте
- NumOfProducts — количество продуктов банка, используемых клиентом
- HasCrCard — наличие кредитной карты
- IsActiveMember — активность клиента
- EstimatedSalary — предполагаемая зарплата

Целевой признак

- Exited — факт ухода клиента

- Данные состоят из 10000 объектов
- Данные имеют 14 признаков (1 целевой и 13 вспомогательных)
- Явные дубликаты отсутствуют
- Данные в признаке "Tenure" обозначают количество лет, переведем их в формат целого числа (int)
- Пропущенные значения есть в признаке "Tenure" - 909 шт. - 9.09%
  - Заменяя пропущенные значения средним или медианным есть риск повлиять на обучение модели
  - Удалим пропущенные значения
- Признаки "RowNumber", "CustomerId" и "Surname" ни как не влияют на вероятность ухода клиента, но могут усложнить обучение модели
  - Удалим данные признаки

In [3]:
```
data=data.dropna(subset=['Tenure'])
```

```
data['Tenure'] = data['Tenure'].astype(int)
data = data.drop(['RowNumber', 'CustomerId', 'Surname'], axis=1)
```

Изучим подробнее каждый признак и проверим на аномалии

CreditScore — кредитный рейтинг

In [4]:
```
print("Минимальное значение кредитного рейтинга:", data['CreditScore'].min())
print("Максимальное значение кредитного рейтинга:", data['CreditScore'].max())
```

```
Минимальное значение кредитного рейтинга: 350
Максимальное значение кредитного рейтинга: 850
Аномалий не обнаружено
```

Geography — страна проживания

In [5]:
```
data['Geography'].value_counts()
```

Out[5]:
```
France     4550
Germany    2293
Spain      2248
Name: Geography, dtype: int64
Аномалий не обнаружено
```

Gender — пол

In [6]:
```
data['Gender'].value_counts()
```

Out[6]:
```
Male      4974
Female    4117
Name: Gender, dtype: int64
Аномалий не обнаружено
```

Age — возраст

In [7]:
```
print("Минимальный возраст клиента:", data['Age'].min())
print("Максимальный возраст клиента:", data['Age'].max())
```

```
Минимальный возраст клиента: 18
Максимальный возраст клиента: 92
Аномалий не обнаружено
```

Tenure — сколько лет человек является клиентом банка

In [8]:
```
data['Tenure'].value_counts()
```

Out[8]:
```
1     952
2     950
8     933
3     928
5     927
7     925
4     885
9     882
6     881
10    446
0     382
Name: Tenure, dtype: int64
Аномалий не обнаружено
```

Balance — баланс на счёте

In [9]:
```
print("Минимальный баланс на счёте:", data['Balance'].min())
print("Максимальный баланс на счёте:", data['Balance'].max())
print("Количество клиентов с нулевым балансом:", len(data.loc[data['Balance']==0]))
```

```
Минимальный баланс на счёте: 0.0
Максимальный баланс на счёте: 250898.09
Количество клиентов с нулевым балансом: 3283
Аномалий не обнаружено
```

NumOfProducts — количество продуктов банка, используемых клиентом

In [10]:
```
data['NumOfProducts'].value_counts()
```

Out[10]:
```
1    4617
2    4184
3     234
4      56
Name: NumOfProducts, dtype: int64
Аномалий не обнаружено
```

HasCrCard — наличие кредитной карты

In [11]:
```
data['HasCrCard'].value_counts()
```

Out[11]:
```
1    6409
0    2682
Name: HasCrCard, dtype: int64
```
Аномалий не обнаружено

IsActiveMember — активность клиента

In [12]:
```
data['IsActiveMember'].value_counts()
```

Out[12]:
```
1    4687
0    4404
Name: IsActiveMember, dtype: int64
```
Аномалий не обнаружено

EstimatedSalary — предполагаемая зарплата

In [13]:
```
print("Минимальная предполагаемая зарплата:", data['EstimatedSalary'].min())
print("Максимальная предполагаемая зарплата:", data['EstimatedSalary'].max())
```

```
Минимальная предполагаемая зарплата: 11.58
Максимальная предполагаемая зарплата: 199992.48
```
Аномалий не обнаружено

Exited — факт ухода клиента

In [14]:
```
data['Exited'].value_counts()
```

Out[14]:
```
0    7237
1    1854
Name: Exited, dtype: int64
```
Аномалий не обнаружено

Подготовим данные

Преобразуем данные методом OHE, чтобы исключить попадание в дамми-ловушку

In [15]:
```
data = pd.get_dummies(data, drop_first=True)
display(data.sample(5, random_state=r_state))
```

	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited	Geography_Germany	Geography_Spain	G
862	725	41	7	113980.21	1	1	1	116704.25	0	0	0	
9727	530	45	1	0.00	1	0	1	190663.89	1	0	0	
1717	707	35	3	56674.48	1	1	0	17987.40	1	0	1	
8640	730	32	9	127661.69	1	0	0	60905.51	0	0	0	
5288	582	30	2	0.00	2	1	1	132029.95	0	0	0	

Определим целевой и вспомогательные признаки

In [16]:
```
features = data.drop(['Exited'], axis=1)
target = data['Exited']
```

Отделим обучающие данные

In [17]:
```
features_train, features_valid, target_train, target_valid = train_test_split(
    features, target, test_size=0.4, random_state=r_state, stratify=target)
```

Разделим оставшиеся данные на валидационную и тестовую выборки

In [18]:
```
features_valid, features_test, target_valid, target_test = train_test_split(
    features_valid, target_valid, test_size=0.5, random_state=r_state, stratify=target_valid)
```

Проверим правильность разделения выборок

In [19]:
```
features = {'features_train' : features_train, 'features_valid' : features_valid, 'features_test' : features_test}
targets = {'target_train' : target_train, 'target_valid' : target_valid, 'target_test' : target_test}

print('Вспомогательные признаки:')
print()
for i in features:
    sh = features[i].shape
    psh = sh[0]/len(data)
    print('{} - Объектов: {} шт., признаков: {} шт. - {}%'.format(i, sh[0], sh[1], round(psh*100, 2)))

print()

print('Целевые признаки:')
print()
for i in targets:
```

```
    sh = targets[i].shape
    psh = sh[0]/len(data)
    print('{} - Объектов: {} шт. - {}%'.format(i, sh[0], round(psh*100, 2)))
```

Вспомогательные признаки:

features_train - Объектов: 5454 шт., признаков: 11 шт. - 59.99%
features_valid - Объектов: 1818 шт., признаков: 11 шт. - 20.0%
features_test - Объектов: 1819 шт., признаков: 11 шт. - 20.01%

Целевые признаки:

target_train - Объектов: 5454 шт. - 59.99%
target_valid - Объектов: 1818 шт. - 20.0%
target_test - Объектов: 1819 шт. - 20.01%
Стандартизируем численные признаки в выборках, так как в данных присутствуют значения в разных масштабах

In [20]:
```
numeric = ['CreditScore', 'Age', 'Tenure', 'Balance', 'NumOfProducts', 'EstimatedSalary']

scaler = StandardScaler()
scaler.fit(features_train[numeric])
```

Out[20]:StandardScaler()

In [21]:
```
features_train[numeric] = scaler.transform(features_train[numeric])
display(features_train.sample(5, random_state=r_state))
```

	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Geography_Germany	Geography_Spain	Genc
5329	-0.313795	0.202723	-0.705222	-1.222967	0.789359	1	1	0.654508	0	0	
8279	-0.470035	-0.468939	-1.053112	-1.222967	-0.910943	0	0	-1.106494	0	0	
8495	0.477820	0.298675	-0.357331	-1.222967	-0.910943	0	0	0.464100	0	0	
1639	-1.522050	-0.660843	1.034232	1.502997	0.789359	0	1	-1.459704	1	0	
9993	-0.063812	-1.044650	0.686341	1.259187	-0.910943	1	0	-1.224724	0	0	

In [22]:
```
features_valid[numeric] = scaler.transform(features_valid[numeric])
display(features_valid.sample(5, random_state=r_state))
```

	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Geography_Germany	Geography_Spain	Genc
2556	-0.167972	0.202723	-0.357331	-1.222967	0.789359	1	1	1.127936	0	1	
7905	0.529899	-0.372988	1.730013	1.333359	0.789359	1	1	-0.494889	0	1	
6219	-0.147140	-0.181084	-1.401003	0.620111	-0.910943	1	0	-0.819705	0	1	
3110	-1.084578	3.369132	1.034232	0.616133	0.789359	1	1	-0.572072	1	0	
5392	0.915291	0.490579	-1.053112	-1.222967	0.789359	1	0	-0.984205	0	1	

In [23]:
```
features_test[numeric] = scaler.transform(features_test[numeric])
display(features_test.sample(5, random_state=r_state))
```

	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Geography_Germany	Geography_Spain	Genc
7239	-0.730435	0.586530	1.034232	0.437686	-0.910943	1	0	-0.907563	1	0	
5574	-1.532466	-0.277036	1.382122	-1.222967	0.789359	1	1	-1.449003	0	1	
8895	0.509067	0.106771	1.034232	0.730581	-0.910943	1	0	0.042339	0	0	
2299	-0.001316	-1.332505	-1.053112	0.607204	-0.910943	1	1	-1.291712	1	0	
5105	0.217420	1.162241	-0.705222	0.328039	-0.910943	1	0	1.333945	1	0	

Исследование задачи

Проверим баланс классов в целевом признаке

In [24]:
```
p_exited_0 = data['Exited'][data['Exited'] == 0].count()/len(data['Exited'])
p_exited_1 = data['Exited'][data['Exited'] == 1].count()/len(data['Exited'])
print('Соотношение оставшихся и ушедших клиентов:', )
```

```
    print('{}% / {}%'.format(round(p_exited_0*100, 1), round(p_exited_1*100, 1)))
    print()

    data['Exited'].value_counts().plot(kind='pie', title='Количество оставшихся и ушедших клиентов', figsize=(8,8),
                                       legend=False, label='', autopct='%1.1f%%', labels = ['Оставшиеся','Ушедшие']);
```

Соотношение оставшихся и ушедших клиентов:
79.6% / 20.4%

Соотношение "ушедших" к "оставшимся" примерно 1 к 4. В данных имеется дисбаланс

Проверим баланс классов по выборкам

In [25]:
```
for i in targets:
    p_exited_0 = targets[i][targets[i] == 0].count()/len(targets[i])
    p_exited_1 = targets[i][targets[i] == 1].count()/len(targets[i])
    print('Соотношение оставшихся и ушедших клиентов в выборке {}:'.format(i))
    print('{}% / {}%'.format(round(p_exited_0*100, 1), round(p_exited_1*100, 1)))
    print()
```

Соотношение оставшихся и ушедших клиентов в выборке target_train:
79.6% / 20.4%

Соотношение оставшихся и ушедших клиентов в выборке target_valid:
79.6% / 20.4%

Соотношение оставшихся и ушедших клиентов в выборке target_test:
79.6% / 20.4%

Выборки разделены равномерно

Обучим модель без учёта дисбаланса

В нашем исследовании целевой признак - столбец "Exited" (категориальный). Категорий в данном признаке две (ушел или не ушел клиент), соответственно нам предстоит решить задачу бинарной (двоичной) классификации.

Для решения данной задачи используем 3 модели:

Decision Tree (Дерево решений) Random Forest (Случайный лес) Logistic Regression (Логистическая регрессия) Применим все 3 модели, подберем им оптимальные гиперпараметры и найдем лучший результат

Для проверки качества на разных моделях создадим функцию с метриками F1 и AUC-ROC:

In [26]:
```
def test_f1_aucroc(model):
    predictions_valid = model.predict(features_valid)
    probabilities_valid = model.predict_proba(features_valid)
```

```
    probabilities_one_valid = probabilities_valid[:, 1]
    return f1_score(target_valid, predictions_valid), roc_auc_score(target_valid, probabilities_one_valid)
```

Decision Tree (Дерево решений)

In [27]:
```
best_f1_DT = 0
best_aucroc_DT = 0
best_depth_DT = 0


for depth in range(1, 21):
    model_decision_tree = DecisionTreeClassifier(random_state=r_state, max_depth=depth)
    model_decision_tree.fit(features_train, target_train)
    f1, aucroc = test_f1_aucroc(model_decision_tree)
    print('Глубина =', depth, ', F1-мера =', f1, ', AUC-ROC =', aucroc)
    if ((f1 > best_f1_DT) and (aucroc > best_aucroc_DT)):
        best_f1_DT = f1
        best_aucroc_DT = aucroc
        best_depth_DT = depth

print()
print('Лучший результат:')
print('Глубина =', best_depth_DT, ', F1-мера =', best_f1_DT, ', AUC-ROC =', best_aucroc_DT)
```

```
Глубина = 1 , F1-мера = 0.0 , AUC-ROC = 0.6743788151710854
Глубина = 2 , F1-мера = 0.5240464344941957 , AUC-ROC = 0.7304023381398823
Глубина = 3 , F1-мера = 0.5208681135225375 , AUC-ROC = 0.7871290540704161
Глубина = 4 , F1-мера = 0.5127272727272727 , AUC-ROC = 0.8113524216847945
Глубина = 5 , F1-мера = 0.5733788395904437 , AUC-ROC = 0.8450470068195747
Глубина = 6 , F1-мера = 0.5797598627787307 , AUC-ROC = 0.8372802172726544
Глубина = 7 , F1-мера = 0.5815126050420169 , AUC-ROC = 0.8245379882534177
Глубина = 8 , F1-мера = 0.5463576158940396 , AUC-ROC = 0.806618209996703
Глубина = 9 , F1-мера = 0.5382059800664452 , AUC-ROC = 0.7900899155609618
Глубина = 10 , F1-мера = 0.5486443381180223 , AUC-ROC = 0.7564260660125885
Глубина = 11 , F1-мера = 0.5214814814814815 , AUC-ROC = 0.7171525062542261
Глубина = 12 , F1-мера = 0.5007541478129715 , AUC-ROC = 0.6987716942014057
Глубина = 13 , F1-мера = 0.5021770682148041 , AUC-ROC = 0.685568431386063
Глубина = 14 , F1-мера = 0.4921316165951359 , AUC-ROC = 0.6732369415669934
Глубина = 15 , F1-мера = 0.49108367626886146 , AUC-ROC = 0.6761139787309742
Глубина = 16 , F1-мера = 0.4918918918918919 , AUC-ROC = 0.675929565212532
Глубина = 17 , F1-мера = 0.5040431266846361 , AUC-ROC = 0.6809227009315678
Глубина = 18 , F1-мера = 0.4823989569752281 , AUC-ROC = 0.6752803923723588
Глубина = 19 , F1-мера = 0.48339973439575035 , AUC-ROC = 0.6750857336584475
Глубина = 20 , F1-мера = 0.48412698412698413 , AUC-ROC = 0.6749050456656304

Лучший результат:
Глубина = 5 , F1-мера = 0.5733788395904437 , AUC-ROC = 0.8450470068195747
```

Random Forest (Случайный лес)

In [28]:
```
best_f1_RF = 0
best_aucroc_RF = 0
best_est_RF = 0
best_depth_RF = 0

for est in range(10, 51, 10):
    for depth in range(1, 21):
        model_random_forest = RandomForestClassifier(random_state=r_state, n_estimators=est, max_depth=depth)
        model_random_forest.fit(features_train, target_train)
        f1, aucroc = test_f1_aucroc(model_random_forest)
        print('Количество деревьев =', est,', глубина =', depth, ', F1-мера =', f1, ', AUC-ROC =', aucroc)
        if ((f1 > best_f1_RF) and (aucroc > best_aucroc_RF)):
            best_f1_RF = f1
            best_aucroc_RF = aucroc
            best_est_RF = est
            best_depth_RF = depth

print()
print('Лучший результат:')
print('Количество деревьев =', best_est_RF,', глубина =', best_depth_RF, ', F1-мера =', best_f1_RF,
      ', AUC-ROC =', best_aucroc_RF)
```

```
Количество деревьев = 10 , глубина = 1 , F1-мера = 0.0 , AUC-ROC = 0.8003043754435705
Количество деревьев = 10 , глубина = 2 , F1-мера = 0.20192307692307693 , AUC-ROC = 0.8213684973278667
Количество деревьев = 10 , глубина = 3 , F1-мера = 0.23584905660377362 , AUC-ROC = 0.8418588882658982
Количество деревьев = 10 , глубина = 4 , F1-мера = 0.4921875 , AUC-ROC = 0.8527458055238368
Количество деревьев = 10 , глубина = 5 , F1-мера = 0.5265151515151515 , AUC-ROC = 0.8672278550099937
Количество деревьев = 10 , глубина = 6 , F1-мера = 0.5369369369369369 , AUC-ROC = 0.8613443186665599
Количество деревьев = 10 , глубина = 7 , F1-мера = 0.5581395348837209 , AUC-ROC = 0.8678323215426658
Количество деревьев = 10 , глубина = 8 , F1-мера = 0.5748709122203098 , AUC-ROC = 0.8692843451550472
Количество деревьев = 10 , глубина = 9 , F1-мера = 0.5783132530120482 , AUC-ROC = 0.8754798942695828
Количество деревьев = 10 , глубина = 10 , F1-мера = 0.554006968641115 , AUC-ROC = 0.8574241343275519
Количество деревьев = 10 , глубина = 11 , F1-мера = 0.5884297520661157 , AUC-ROC = 0.8609400991362368
Количество деревьев = 10 , глубина = 12 , F1-мера = 0.5878378378378378 , AUC-ROC = 0.854859109934673
```

Количество деревьев = 10 , глубина = 12 , F1-мера = 0.5875785785785785 , AUC-ROC = 0.8548516169934073
Количество деревьев = 10 , глубина = 13 , F1-мера = 0.5884297520661157 , AUC-ROC = 0.8584803208422669
Количество деревьев = 10 , глубина = 14 , F1-мера = 0.5794701986754967 , AUC-ROC = 0.8446893563595652
Количество деревьев = 10 , глубина = 15 , F1-мера = 0.5874799357945426 , AUC-ROC = 0.8426701214707629
Количество деревьев = 10 , глубина = 16 , F1-мера = 0.5557404326123129 , AUC-ROC = 0.8497942578473541
Количество деревьев = 10 , глубина = 17 , F1-мера = 0.546979865771812 , AUC-ROC = 0.8462242729171052
Количество деревьев = 10 , глубина = 18 , F1-мера = 0.5747899159663865 , AUC-ROC = 0.8537237560004247
Количество деревьев = 10 , глубина = 19 , F1-мера = 0.563758389261745 , AUC-ROC = 0.8402978930289826
Количество деревьев = 10 , глубина = 20 , F1-мера = 0.5888157894736843 , AUC-ROC = 0.84857787373076
Количество деревьев = 20 , глубина = 1 , F1-мера = 0.0 , AUC-ROC = 0.8133865586760972
Количество деревьев = 20 , глубина = 2 , F1-мера = 0.14962593516209474 , AUC-ROC = 0.8221340928438241
Количество деревьев = 20 , глубина = 3 , F1-мера = 0.21904761904761902 , AUC-ROC = 0.8452044102772349
Количество деревьев = 20 , глубина = 4 , F1-мера = 0.4543610547667342 , AUC-ROC = 0.858304289756481
Количество деревьев = 20 , глубина = 5 , F1-мера = 0.5431192660550459 , AUC-ROC = 0.8683017377714278
Количество деревьев = 20 , глубина = 6 , F1-мера = 0.5454545454545454 , AUC-ROC = 0.8669931468956127
Количество деревьев = 20 , глубина = 7 , F1-мера = 0.5549738219895287 , AUC-ROC = 0.8718298105383943
Количество деревьев = 20 , глубина = 8 , F1-мера = 0.5884353741496599 , AUC-ROC = 0.8718568205991762
Количество деревьев = 20 , глубина = 9 , F1-мера = 0.5983193277310924 , AUC-ROC = 0.8755190122886461
Количество деревьев = 20 , глубина = 10 , F1-мера = 0.5773195876288659 , AUC-ROC = 0.8709626944491531
Количество деревьев = 20 , глубина = 11 , F1-мера = 0.5902192242833052 , AUC-ROC = 0.8699866067353778
Количество деревьев = 20 , глубина = 12 , F1-мера = 0.5728643216080402 , AUC-ROC = 0.864063020991474
Количество деревьев = 20 , глубина = 13 , F1-мера = 0.5960264900662251 , AUC-ROC = 0.868475906094401
Количество деревьев = 20 , глубина = 14 , F1-мера = 0.6009852216748769 , AUC-ROC = 0.8637482140761534
Количество деревьев = 20 , глубина = 15 , F1-мера = 0.6089743589743591 , AUC-ROC = 0.8639624317995965
Количество деревьев = 20 , глубина = 16 , F1-мера = 0.5809682804674458 , AUC-ROC = 0.8691297358416056
Количество деревьев = 20 , глубина = 17 , F1-мера = 0.5855263157894737 , AUC-ROC = 0.862591438369561
Количество деревьев = 20 , глубина = 18 , F1-мера = 0.5784313725490197 , AUC-ROC = 0.8697919480214666
Количество деревьев = 20 , глубина = 19 , F1-мера = 0.5770491803278689 , AUC-ROC = 0.8611925034973372
Количество деревьев = 20 , глубина = 20 , F1-мера = 0.5891980360065466 , AUC-ROC = 0.8605321540802888
Количество деревьев = 30 , глубина = 1 , F1-мера = 0.0 , AUC-ROC = 0.8186125397467016
Количество деревьев = 30 , глубина = 2 , F1-мера = 0.1309823677581864 , AUC-ROC = 0.829258229220415
Количество деревьев = 30 , глубина = 3 , F1-мера = 0.23167848699763594 , AUC-ROC = 0.843091105866399
Количество деревьев = 30 , глубина = 4 , F1-мера = 0.375 , AUC-ROC = 0.8603617112829407
Количество деревьев = 30 , глубина = 5 , F1-мера = 0.490272373540856 , AUC-ROC = 0.8692284622706705
Количество деревьев = 30 , глубина = 6 , F1-мера = 0.5444646098003629 , AUC-ROC = 0.870619014710238
Количество деревьев = 30 , глубина = 7 , F1-мера = 0.5555555555555555 , AUC-ROC = 0.8739906154009504
Количество деревьев = 30 , глубина = 8 , F1-мера = 0.584192439862543 , AUC-ROC = 0.8739626739587623
Количество деревьев = 30 , глубина = 9 , F1-мера = 0.5918367346938775 , AUC-ROC = 0.8786931601212287
Количество деревьев = 30 , глубина = 10 , F1-мера = 0.5719237435008666 , AUC-ROC = 0.8738816437764162
Количество деревьев = 30 , глубина = 11 , F1-мера = 0.5866209262435678 , AUC-ROC = 0.8749396930539438
Количество деревьев = 30 , глубина = 12 , F1-мера = 0.5810810810810811 , AUC-ROC = 0.8663542192509086
Количество деревьев = 30 , глубина = 13 , F1-мера = 0.6013289036544851 , AUC-ROC = 0.8704103852752326
Количество деревьев = 30 , глубина = 14 , F1-мера = 0.5983471074380166 , AUC-ROC = 0.8674672200314062
Количество деревьев = 30 , глубина = 15 , F1-мера = 0.5964343598055106 , AUC-ROC = 0.8670276080076448
Количество деревьев = 30 , глубина = 16 , F1-мера = 0.5890183028286189 , AUC-ROC = 0.870357296535075
Количество деревьев = 30 , глубина = 17 , F1-мера = 0.6098360655737705 , AUC-ROC = 0.8674402099706243
Количество деревьев = 30 , глубина = 18 , F1-мера = 0.602291325695581 , AUC-ROC = 0.8700089598891283
Количество деревьев = 30 , глубина = 19 , F1-мера = 0.6019736842105263 , AUC-ROC = 0.866179119546529
Количество деревьев = 30 , глубина = 20 , F1-мера = 0.6009852216748769 , AUC-ROC = 0.8664697105452865
Количество деревьев = 40 , глубина = 1 , F1-мера = 0.0 , AUC-ROC = 0.8241002389924688
Количество деревьев = 40 , глубина = 2 , F1-мера = 0.135678391959799 , AUC-ROC = 0.83826375603768
Количество деревьев = 40 , глубина = 3 , F1-мера = 0.22748815165876776 , AUC-ROC = 0.8511913299567653
Количество деревьев = 40 , глубина = 4 , F1-мера = 0.4150943396226415 , AUC-ROC = 0.8630580604541043
Количество деревьев = 40 , глубина = 5 , F1-мера = 0.5151515151515151 , AUC-ROC = 0.8706944566041461
Количество деревьев = 40 , глубина = 6 , F1-мера = 0.5360443622920518 , AUC-ROC = 0.8717916239007371
Количество деревьев = 40 , глубина = 7 , F1-мера = 0.5567375886524824 , AUC-ROC = 0.8751744011683249
Количество деревьев = 40 , глубина = 8 , F1-мера = 0.5793103448275861 , AUC-ROC = 0.8763628438427307
Количество деревьев = 40 , глубина = 9 , F1-мера = 0.5932203389830509 , AUC-ROC = 0.8821485851385057
Количество деревьев = 40 , глубина = 10 , F1-мера = 0.5709342560553633 , AUC-ROC = 0.8772066753968149
Количество деревьев = 40 , глубина = 11 , F1-мера = 0.581196581196581 , AUC-ROC = 0.8756680333136502
Количество деревьев = 40 , глубина = 12 , F1-мера = 0.5935919055649241 , AUC-ROC = 0.869888811687719
Количество деревьев = 40 , глубина = 13 , F1-мера = 0.5956738768718801 , AUC-ROC = 0.8735612485726578
Количество деревьев = 40 , глубина = 14 , F1-мера = 0.5874587458745875 , AUC-ROC = 0.8699419004278766
Количество деревьев = 40 , глубина = 15 , F1-мера = 0.6061588330632092 , AUC-ROC = 0.8684321311683063
Количество деревьев = 40 , глубина = 16 , F1-мера = 0.5990180032733223 , AUC-ROC = 0.870544504197736
Количество деревьев = 40 , глубина = 17 , F1-мера = 0.5866666666666667 , AUC-ROC = 0.8669177050017044
Количество деревьев = 40 , глубина = 18 , F1-мера = 0.5915032679738562 , AUC-ROC = 0.8706488189152387
Количество деревьев = 40 , глубина = 19 , F1-мера = 0.6052631578947368 , AUC-ROC = 0.8654871031616673
Количество деревьев = 40 , глубина = 20 , F1-мера = 0.6078431372549019 , AUC-ROC = 0.8626510467795624
Количество деревьев = 50 , глубина = 1 , F1-мера = 0.0 , AUC-ROC = 0.8234277816171389
Количество деревьев = 50 , глубина = 2 , F1-мера = 0.16748768472906403 , AUC-ROC = 0.8367716830248288
Количество деревьев = 50 , глубина = 3 , F1-мера = 0.23584905660377362 , AUC-ROC = 0.8487334144256078
Количество деревьев = 50 , глубина = 4 , F1-мера = 0.4273858921161826 , AUC-ROC = 0.86274418492019
Количество деревьев = 50 , глубина = 5 , F1-мера = 0.5057471264367815 , AUC-ROC = 0.8695693478653669
Количество деревьев = 50 , глубина = 6 , F1-мера = 0.5306122448979592 , AUC-ROC = 0.8715625040747936
Количество деревьев = 50 , глубина = 7 , F1-мера = 0.5525846702317291 , AUC-ROC = 0.8762073031478828
Количество деревьев = 50 , глубина = 8 , F1-мера = 0.5729166666666666 , AUC-ROC = 0.8782814895396555
Количество деревьев = 50 , глубина = 9 , F1-мера = 0.5956006768189509 , AUC-ROC = 0.8810085742972262
Количество деревьев = 50 , глубина = 10 , F1-мера = 0.5763293310463121 , AUC-ROC = 0.8769477513658709
Количество деревьев = 50 , глубина = 11 , F1-мера = 0.5898305084745763 , AUC-ROC = 0.8754575411158323
Количество деревьев = 50 , глубина = 12 , F1-мера = 0.6076794657762938 , AUC-ROC = 0.8723746686610647
Количество деревьев = 50 , глубина = 13 , F1-мера = 0.5976627712854758 , AUC-ROC = 0.8745298852351829
Количество деревьев = 50 , глубина = 14 , F1-мера = 0.6075533661740558 , AUC-ROC = 0.872247069408405

```
Количество деревьев = 50 , глубина = 15 , F1-мера = 0.6019736842105263 , AUC-ROC = 0.8700788134945988
Количество деревьев = 50 , глубина = 16 , F1-мера = 0.5953947368421053 , AUC-ROC = 0.8710223028591546
Количество деревьев = 50 , глубина = 17 , F1-мера = 0.5894039735099337 , AUC-ROC = 0.8657693117277684
Количество деревьев = 50 , глубина = 18 , F1-мера = 0.5974025974025975 , AUC-ROC = 0.8729344288862355
Количество деревьев = 50 , глубина = 19 , F1-мера = 0.6121112929623569 , AUC-ROC = 0.8656007316932328
Количество деревьев = 50 , глубина = 20 , F1-мера = 0.6223662884927067 , AUC-ROC = 0.8642455717471038

Лучший результат:
Количество деревьев = 20 , глубина = 9 , F1-мера = 0.5983193277310924 , AUC-ROC = 0.8755190122886461
```

Logistic Regression (Логистическая регрессия)

In [29]:
```
best_f1_LR = 0
best_aucroc_LR = 0
best_solver_LR = ''
solvers = ['lbfgs', 'liblinear', 'newton-cg', 'sag', 'saga']

for solver in solvers:
    model_logistic_regression = LogisticRegression(solver=solver, random_state=r_state)
    model_logistic_regression.fit(features_train, target_train)
    f1, aucroc = test_f1_aucroc(model_logistic_regression)
    print('Алгоритм =', solver, ', F1-мера =', f1, ', AUC-ROC =', aucroc)
    if ((f1 > best_f1_LR) and (aucroc > best_aucroc_LR)):
        best_f1_LR = f1
        best_aucroc_LR = aucroc
        best_solver_LR = solver

print()
print('Лучший результат:')
print('Алгоритм =', best_solver_LR, ', F1-мера =', best_f1_LR, ', AUC-ROC =', best_aucroc_LR)
```

```
Алгоритм = lbfgs , F1-мера = 0.3306772908366534 , AUC-ROC = 0.7893382907660986
Алгоритм = liblinear , F1-мера = 0.3306772908366534 , AUC-ROC = 0.7893010355098474
Алгоритм = newton-cg , F1-мера = 0.3306772908366534 , AUC-ROC = 0.7893401535289111
Алгоритм = sag , F1-мера = 0.3306772908366534 , AUC-ROC = 0.7893476045801613
Алгоритм = saga , F1-мера = 0.3306772908366534 , AUC-ROC = 0.7893382907660984

Лучший результат:
Алгоритм = lbfgs , F1-мера = 0.3306772908366534 , AUC-ROC = 0.7893382907660986
```

Определим лучшую модель

In [30]:
```
max_f1 = best_f1_DT
max_aucroc = best_aucroc_DT
if (max_f1 < best_f1_RF) and (max_aucroc < best_aucroc_RF):
    max_f1 = best_f1_RF
    max_aucroc = best_aucroc_RF
if (max_f1 < best_f1_LR) and (max_aucroc < best_aucroc_LR):
    max_f1 = best_f1_LR
    max_aucroc = best_aucroc_LR

if (max_f1 == best_f1_DT) and (max_aucroc == best_aucroc_DT):
    print('Лучшая модель: Decision Tree (Дерево решений), с гиперпараметром max_depth =', best_depth_DT)
if (max_f1 == best_f1_RF) and (max_aucroc == best_aucroc_RF):
    print('Лучшая модель: Random Forest (Случайный лес), с гиперпараметрами n_estimators =', best_est_RF,
      'max_depth =', best_depth_RF)
if (max_f1 == best_f1_LR) and (max_aucroc == best_aucroc_LR):
    print('Лучшая модель: Logistic Regression (Логистическая регрессия), с гиперпараметром solver =', best_solver_LR)
print('F1-мера =', max_f1, ', AUC-ROC =', max_aucroc)
```

```
Лучшая модель: Random Forest (Случайный лес), с гиперпараметрами n_estimators = 20 max_depth = 9
F1-мера = 0.5983193277310924 , AUC-ROC = 0.8755190122886461
```

Обучив три модели с дисбалансом классов мы определили лучшую модель:

- Random Forest (Случайный лес), с гиперпараметрами:
  - n_estimators = 20
  - max_depth = 9
- F1-мера = 0.5983193277310924
- AUC-ROC = 0.8755190122886461

F1-мера нашей лучшей модели (0.598) примерно равна поставленной задаче (0.59). Улучшим качество модели, учитывая дисбаланс классов

Борьба с дисбалансом

Увеличим объекты редкого класса (техника upsampling)

In [31]:
```
def upsample(features, target, repeat):
    features_zeros = features_train[target_train == 0]
    features_ones = features_train[target_train == 1]
    target_zeros = target_train[target_train == 0]
    target_ones = target_train[target_train == 1]
```

```
    features_upsampled = pd.concat([features_zeros] + [features_ones] * repeat)
    target_upsampled = pd.concat([target_zeros] + [target_ones] * repeat)
    features_upsampled, target_upsampled = shuffle(features_upsampled, target_upsampled, random_state=r_state)

    return features_upsampled, target_upsampled

features_upsampled, target_upsampled = upsample(features_train, target_train, 4)
```

Проверим баланс классов в целевом признаке

In [32]:
```
p_exited_0 = target_upsampled[target_upsampled == 0].count()/len(target_upsampled)
p_exited_1 = target_upsampled[target_upsampled == 1].count()/len(target_upsampled)
print('Соотношение оставшихся и ушедших клиентов:', )
print('{}% / {}%'.format(round(p_exited_0*100, 1), round(p_exited_1*100, 1)))
```

```
Соотношение оставшихся и ушедших клиентов:
49.4% / 50.6%
```

Decision Tree (Дерево решений)

In [33]:
```
best_f1_DT_up = 0
best_aucroc_DT_up = 0
best_depth_DT_up = 0


for depth in range(1, 21):
    model_decision_tree = DecisionTreeClassifier(random_state=r_state, max_depth=depth)
    model_decision_tree.fit(features_upsampled, target_upsampled)
    f1, aucroc = test_f1_aucroc(model_decision_tree)
    print('Глубина =', depth, ', F1-мера =', f1, ', AUC-ROC =', aucroc)
    if ((f1 > best_f1_DT_up) and (aucroc > best_aucroc_DT_up)):
        best_f1_DT_up = f1
        best_aucroc_DT_up = aucroc
        best_depth_DT_up = depth

print()
print('Лучший результат:')
print('Глубина =', best_depth_DT_up, ', F1-мера =', best_f1_DT_up, ', AUC-ROC =', best_aucroc_DT_up)
```

```
Глубина = 1 , F1-мера = 0.5092402464065708 , AUC-ROC = 0.7115642178165812
Глубина = 2 , F1-мера = 0.5376128385155466 , AUC-ROC = 0.7696144267254307
Глубина = 3 , F1-мера = 0.5376128385155466 , AUC-ROC = 0.8123005679563815
Глубина = 4 , F1-мера = 0.5522682445759368 , AUC-ROC = 0.8276497335317796
Глубина = 5 , F1-мера = 0.5711481844946025 , AUC-ROC = 0.8466415317871161
Глубина = 6 , F1-мера = 0.5825049701789264 , AUC-ROC = 0.8347040163029001
Глубина = 7 , F1-мера = 0.5628042843232717 , AUC-ROC = 0.8318847247861082
Глубина = 8 , F1-мера = 0.5813449023861171 , AUC-ROC = 0.8171605161343202
Глубина = 9 , F1-мера = 0.5548245614035088 , AUC-ROC = 0.790960757175828
Глубина = 10 , F1-мера = 0.531284302963776 , AUC-ROC = 0.7452383125604234
Глубина = 11 , F1-мера = 0.5378346915017462 , AUC-ROC = 0.7394795813254302
Глубина = 12 , F1-мера = 0.5087924970691676 , AUC-ROC = 0.7139578680307057
Глубина = 13 , F1-мера = 0.500590318772137 , AUC-ROC = 0.7174170185736081
Глубина = 14 , F1-мера = 0.5115712545676004 , AUC-ROC = 0.7119796139237794
Глубина = 15 , F1-мера = 0.5155666251556662 , AUC-ROC = 0.7060504398914382
Глубина = 16 , F1-мера = 0.5114503816793893 , AUC-ROC = 0.7009827936599006
Глубина = 17 , F1-мера = 0.523936170212766 , AUC-ROC = 0.7103459709371746
Глубина = 18 , F1-мера = 0.49736842105263157 , AUC-ROC = 0.6897345004163274
Глубина = 19 , F1-мера = 0.5190039318479686 , AUC-ROC = 0.7007331834430189
Глубина = 20 , F1-мера = 0.5245033112582781 , AUC-ROC = 0.7025754558646293

Лучший результат:
Глубина = 5 , F1-мера = 0.5711481844946025 , AUC-ROC = 0.8466415317871161
```

Random Forest (Случайный лес)

In [34]:
```
best_f1_RF_up = 0
best_aucroc_RF_up = 0
best_est_RF_up = 0
best_depth_RF_up = 0

for est in range(10, 51, 10):
    for depth in range(1, 21):
        model_random_forest = RandomForestClassifier(random_state=r_state, n_estimators=est, max_depth=depth)
        model_random_forest.fit(features_upsampled, target_upsampled)
        f1, aucroc = test_f1_aucroc(model_random_forest)
        print('Количество деревьев =', est,', глубина =', depth, ', F1-мера =', f1, ', AUC-ROC =', aucroc)
        if ((f1 > best_f1_RF_up) and (aucroc > best_aucroc_RF_up)):
            best_f1_RF_up = f1
            best_aucroc_RF_up = aucroc
            best_est_RF_up = est
            best_depth_RF_up = depth

print()
print('Лучший результат:')
```

```
        print('Количество деревьев =', best_est_RF_up,', глубина =', best_depth_RF_up, ', F1-мера =', best_f1_RF_up,
              ', AUC-ROC =', best_aucroc_RF_up)
```

```
Количество деревьев = 10 , глубина = 1 , F1-мера = 0.4638157894736842 , AUC-ROC = 0.7959436476993949
Количество деревьев = 10 , глубина = 2 , F1-мера = 0.5780104712041885 , AUC-ROC = 0.8458275044380322
Количество деревьев = 10 , глубина = 3 , F1-мера = 0.5516569200779727 , AUC-ROC = 0.845608629807558
Количество деревьев = 10 , глубина = 4 , F1-мера = 0.6077348066298344 , AUC-ROC = 0.861752263722508
Количество деревьев = 10 , глубина = 5 , F1-мера = 0.6051743532058492 , AUC-ROC = 0.8704392580988269
Количество деревьев = 10 , глубина = 6 , F1-мера = 0.6129753914988815 , AUC-ROC = 0.8712001967077531
Количество деревьев = 10 , глубина = 7 , F1-мера = 0.6187845303867403 , AUC-ROC = 0.8743147361303337
Количество деревьев = 10 , глубина = 8 , F1-мера = 0.6285714285714286 , AUC-ROC = 0.8716742698435465
Количество деревьев = 10 , глубина = 9 , F1-мера = 0.6093928980526919 , AUC-ROC = 0.8669605485463932
Количество деревьев = 10 , глубина = 10 , F1-мера = 0.6251497005988025 , AUC-ROC = 0.8643973869163264
Количество деревьев = 10 , глубина = 11 , F1-мера = 0.6140776699029126 , AUC-ROC = 0.8554831354768766
Количество деревьев = 10 , глубина = 12 , F1-мера = 0.5989583333333334 , AUC-ROC = 0.8578860995050639
Количество деревьев = 10 , глубина = 13 , F1-мера = 0.6143617021276596 , AUC-ROC = 0.8432280189331212
Количество деревьев = 10 , глубина = 14 , F1-мера = 0.606060606060606 , AUC-ROC = 0.84588059317819
Количество деревьев = 10 , глубина = 15 , F1-мера = 0.5847457627118643 , AUC-ROC = 0.8388747422401959
Количество деревьев = 10 , глубина = 16 , F1-мера = 0.6005665722379602 , AUC-ROC = 0.8418905552337117
Количество деревьев = 10 , глубина = 17 , F1-мера = 0.5868613138686132 , AUC-ROC = 0.8484707648690386
Количество деревьев = 10 , глубина = 18 , F1-мера = 0.5847076461769115 , AUC-ROC = 0.8231278768043185
Количество деревьев = 10 , глубина = 19 , F1-мера = 0.593245227606461 , AUC-ROC = 0.8344357784578932
Количество деревьев = 10 , глубина = 20 , F1-мера = 0.5730994152046784 , AUC-ROC = 0.8368461935373308
Количество деревьев = 20 , глубина = 1 , F1-мера = 0.5433746425166825 , AUC-ROC = 0.8172210559257279
Количество деревьев = 20 , глубина = 2 , F1-мера = 0.5515210991167812 , AUC-ROC = 0.8370045283763974
Количество деревьев = 20 , глубина = 3 , F1-мера = 0.5737704918032787 , AUC-ROC = 0.854784599422171
Количество деревьев = 20 , глубина = 4 , F1-мера = 0.6025641025641026 , AUC-ROC = 0.8648267537446189
Количество деревьев = 20 , глубина = 5 , F1-мера = 0.6081229418221735 , AUC-ROC = 0.8766180423480497
Количество деревьев = 20 , глубина = 6 , F1-мера = 0.6072607260726073 , AUC-ROC = 0.8770306443110293
Количество деревьев = 20 , глубина = 7 , F1-мера = 0.6228070175438596 , AUC-ROC = 0.8769319178819641
Количество деревьев = 20 , глубина = 8 , F1-мера = 0.6341463414634148 , AUC-ROC = 0.8743613052006476
Количество деревьев = 20 , глубина = 9 , F1-мера = 0.617169373549884 , AUC-ROC = 0.8679636463209504
Количество деревьев = 20 , глубина = 10 , F1-мера = 0.6223277909738717 , AUC-ROC = 0.8719881453774609
Количество деревьев = 20 , глубина = 11 , F1-мера = 0.6287128712871287 , AUC-ROC = 0.865757203769487
Количество деревьев = 20 , глубина = 12 , F1-мера = 0.646074646074646 , AUC-ROC = 0.8668152530470143
Количество деревьев = 20 , глубина = 13 , F1-мера = 0.6216216216216216 , AUC-ROC = 0.8551087201515544
Количество деревьев = 20 , глубина = 14 , F1-мера = 0.6198347107438017 , AUC-ROC = 0.8565039294981529
Количество деревьев = 20 , глубина = 15 , F1-мера = 0.6253521126760563 , AUC-ROC = 0.8520686912414757
Количество деревьев = 20 , глубина = 16 , F1-мера = 0.6326241134751773 , AUC-ROC = 0.8505896575683123
Количество деревьев = 20 , глубина = 17 , F1-мера = 0.5977011494252873 , AUC-ROC = 0.8545498913077898
Количество деревьев = 20 , глубина = 18 , F1-мера = 0.5875370919881306 , AUC-ROC = 0.8497178845720395
Количество деревьев = 20 , глубина = 19 , F1-мера = 0.6142649199417759 , AUC-ROC = 0.8537516974426131
Количество деревьев = 20 , глубина = 20 , F1-мера = 0.6044444444444445 , AUC-ROC = 0.8473568327071346
Количество деревьев = 30 , глубина = 1 , F1-мера = 0.5183823529411765 , AUC-ROC = 0.8140087214554884
Количество деревьев = 30 , глубина = 2 , F1-мера = 0.5582329317269077 , AUC-ROC = 0.8413661875019791
Количество деревьев = 30 , глубина = 3 , F1-мера = 0.562 , AUC-ROC = 0.8576374206695888
Количество деревьев = 30 , глубина = 4 , F1-мера = 0.5918153200419727 , AUC-ROC = 0.865608182744483
Количество деревьев = 30 , глубина = 5 , F1-мера = 0.6019629225736095 , AUC-ROC = 0.8751911660336379
Количество деревьев = 30 , глубина = 6 , F1-мера = 0.6116611661166116 , AUC-ROC = 0.876717700158521
Количество деревьев = 30 , глубина = 7 , F1-мера = 0.6233480176211453 , AUC-ROC = 0.8771498611310323
Количество деревьев = 30 , глубина = 8 , F1-мера = 0.6284403669724771 , AUC-ROC = 0.8756689646950564
Количество деревьев = 30 , глубина = 9 , F1-мера = 0.6252873563218391 , AUC-ROC = 0.8698580761013118
Количество деревьев = 30 , глубина = 10 , F1-мера = 0.6302021403091557 , AUC-ROC = 0.8733460994678087
Количество деревьев = 30 , глубина = 11 , F1-мера = 0.6317103620474406 , AUC-ROC = 0.8679366362601683
Количество деревьев = 30 , глубина = 12 , F1-мера = 0.6356589147286822 , AUC-ROC = 0.8697239571788085
Количество деревьев = 30 , глубина = 13 , F1-мера = 0.6058981233243967 , AUC-ROC = 0.8620121191348584
Количество деревьев = 30 , глубина = 14 , F1-мера = 0.6130374479889041 , AUC-ROC = 0.8598205786858952
Количество деревьев = 30 , глубина = 15 , F1-мера = 0.622905027932961 , AUC-ROC = 0.8578441873417815
Количество деревьев = 30 , глубина = 16 , F1-мера = 0.6225352112676056 , AUC-ROC = 0.8589162073404031
Количество деревьев = 30 , глубина = 17 , F1-мера = 0.611832611832612 , AUC-ROC = 0.8595365073569817
Количество деревьев = 30 , глубина = 18 , F1-мера = 0.6017441860465116 , AUC-ROC = 0.8580276694788177
Количество деревьев = 30 , глубина = 19 , F1-мера = 0.6044444444444445 , AUC-ROC = 0.8554682333743762
Количество деревьев = 30 , глубина = 20 , F1-мера = 0.6026587887740029 , AUC-ROC = 0.8518051103035
Количество деревьев = 40 , глубина = 1 , F1-мера = 0.5352657004830917 , AUC-ROC = 0.8254553989385978
Количество деревьев = 40 , глубина = 2 , F1-мера = 0.5537190082644627 , AUC-ROC = 0.8490444958153034
Количество деревьев = 40 , глубина = 3 , F1-мера = 0.5679012345679012 , AUC-ROC = 0.8593073875310384
Количество деревьев = 40 , глубина = 4 , F1-мера = 0.5995670995670996 , AUC-ROC = 0.8676432511171921
Количество деревьев = 40 , глубина = 5 , F1-мера = 0.601750547045952 , AUC-ROC = 0.8750337625759775
Количество деревьев = 40 , глубина = 6 , F1-мера = 0.6196868008948545 , AUC-ROC = 0.8777217293144848
Количество деревьев = 40 , глубина = 7 , F1-мера = 0.6191536748329621 , AUC-ROC = 0.8778800641535511
Количество деревьев = 40 , глубина = 8 , F1-мера = 0.6321839080459771 , AUC-ROC = 0.8768536818438372
Количество деревьев = 40 , глубина = 9 , F1-мера = 0.6178861788617885 , AUC-ROC = 0.8726708479482599
Количество деревьев = 40 , глубина = 10 , F1-мера = 0.6308243727598567 , AUC-ROC = 0.8739803702054814
Количество деревьев = 40 , глубина = 11 , F1-мера = 0.6281407035175879 , AUC-ROC = 0.8672744240803074
Количество деревьев = 40 , глубина = 12 , F1-мера = 0.6381322957198443 , AUC-ROC = 0.8687385556509705
Количество деревьев = 40 , глубина = 13 , F1-мера = 0.6193029490616622 , AUC-ROC = 0.8629304612014449
Количество деревьев = 40 , глубина = 14 , F1-мера = 0.6181818181818183 , AUC-ROC = 0.8625942325137799
Количество деревьев = 40 , глубина = 15 , F1-мера = 0.6314325452016689 , AUC-ROC = 0.86236418130643
Количество деревьев = 40 , глубина = 16 , F1-мера = 0.6158273381294965 , AUC-ROC = 0.8597917058623008
Количество деревьев = 40 , глубина = 17 , F1-мера = 0.6089466089466089 , AUC-ROC = 0.863152129976138
Количество деревьев = 40 , глубина = 18 , F1-мера = 0.593886462882096 , AUC-ROC = 0.8609363736106117
Количество деревьев = 40 , глубина = 19 , F1-мера = 0.5922619047619048 , AUC-ROC = 0.8576001654133378
Количество деревьев = 40 , глубина = 20 , F1-мера = 0.6099706744868035 , AUC-ROC = 0.8562738782908034
```

```
Количество деревьев = 50 , глубина = 1 , F1-мера = 0.535645472061657 , AUC-ROC = 0.821800658300378
Количество деревьев = 50 , глубина = 2 , F1-мера = 0.5504587155963302 , AUC-ROC = 0.8468762399014972
Количество деревьев = 50 , глубина = 3 , F1-мера = 0.5708376421923474 , AUC-ROC = 0.8589655705549357
Количество деревьев = 50 , глубина = 4 , F1-мера = 0.6034482758620691 , AUC-ROC = 0.8686146819239359
Количество деревьев = 50 , глубина = 5 , F1-мера = 0.6030701754385965 , AUC-ROC = 0.873697230257974
Количество деревьев = 50 , глубина = 6 , F1-мера = 0.6167597765363129 , AUC-ROC = 0.8766571603671134
Количество деревьев = 50 , глубина = 7 , F1-мера = 0.6201550387596898 , AUC-ROC = 0.8766506406972694
Количество деревьев = 50 , глубина = 8 , F1-мера = 0.6360505166475316 , AUC-ROC = 0.8767260825911777
Количество деревьев = 50 , глубина = 9 , F1-мера = 0.6241299303944315 , AUC-ROC = 0.8734643849064053
Количество деревьев = 50 , глубина = 10 , F1-мера = 0.6344993968636913 , AUC-ROC = 0.8748446921505039
Количество деревьев = 50 , глубина = 11 , F1-мера = 0.6331658291457286 , AUC-ROC = 0.868179726807206
Количество деревьев = 50 , глубина = 12 , F1-мера = 0.6300653594771242 , AUC-ROC = 0.8698729782038124
Количество деревьев = 50 , глубина = 13 , F1-мера = 0.625503355704698 , AUC-ROC = 0.8650148927886863
Количество деревьев = 50 , глубина = 14 , F1-мера = 0.6293706293706294 , AUC-ROC = 0.8635544867436484
Количество деревьев = 50 , глубина = 15 , F1-мера = 0.6265734265734265 , AUC-ROC = 0.863492084189428
Количество деревьев = 50 , глубина = 16 , F1-мера = 0.6224783861671469 , AUC-ROC = 0.8622831511240842
Количество деревьев = 50 , глубина = 17 , F1-мера = 0.615606936416185 , AUC-ROC = 0.866787311604826
Количество деревьев = 50 , глубина = 18 , F1-мера = 0.6055312954876273 , AUC-ROC = 0.8621117769453297
Количество деревьев = 50 , глубина = 19 , F1-мера = 0.6119402985074627 , AUC-ROC = 0.8575684984455243
Количество деревьев = 50 , глубина = 20 , F1-мера = 0.6128550074738415 , AUC-ROC = 0.8585678706944565

Лучший результат:
Количество деревьев = 30 , глубина = 7 , F1-мера = 0.6233480176211453 , AUC-ROC = 0.8771498611310323
```

Logistic Regression (Логистическая регрессия)

In [35]:
```
best_f1_LR_up = 0
best_aucroc_LR_up = 0
best_solver_LR_up = ''
solvers = ['lbfgs', 'liblinear', 'newton-cg', 'sag', 'saga']

for solver in solvers:
    model_logistic_regression = LogisticRegression(solver=solver, random_state=r_state)
    model_logistic_regression.fit(features_upsampled, target_upsampled)
    f1, aucroc = test_f1_aucroc(model_logistic_regression)
    print('Алгоритм =', solver, ', F1-мера =', f1, ', AUC-ROC =', aucroc)
    if ((f1 > best_f1_LR_up) and (aucroc > best_aucroc_LR_up)):
        best_f1_LR_up = f1
        best_aucroc_LR_up = aucroc
        best_solver_LR_up = solver

print()
print('Лучший результат:')
print('Алгоритм =', best_solver_LR_up, ', F1-мера =', best_f1_LR_up, ', AUC-ROC =', best_aucroc_LR_up)
```

```
Алгоритм = lbfgs , F1-мера = 0.5050878815911193 , AUC-ROC = 0.7907446766895724
Алгоритм = liblinear , F1-мера = 0.5050878815911193 , AUC-ROC = 0.790767029843323
Алгоритм = newton-cg , F1-мера = 0.5050878815911193 , AUC-ROC = 0.7907372256383223
Алгоритм = sag , F1-мера = 0.5050878815911193 , AUC-ROC = 0.7907409511639474
Алгоритм = saga , F1-мера = 0.5050878815911193 , AUC-ROC = 0.7907390884011348

Лучший результат:
Алгоритм = lbfgs , F1-мера = 0.5050878815911193 , AUC-ROC = 0.7907446766895724
```

Определим лучшую модель

In [36]:
```
max_f1_up = best_f1_DT_up
max_aucroc_up = best_aucroc_DT_up
if (max_f1_up < best_f1_RF_up) and (max_aucroc_up < best_aucroc_RF_up):
    max_f1_up = best_f1_RF_up
    max_aucroc_up = best_aucroc_RF_up
if (max_f1_up < best_f1_LR_up) and (max_aucroc_up < best_aucroc_LR_up):
    max_f1_up = best_f1_LR_up
    max_aucroc_up = best_aucroc_LR_up

if (max_f1_up == best_f1_DT_up) and (max_aucroc_up == best_aucroc_DT_up):
    print('Лучшая модель: Decision Tree (Дерево решений), с гиперпараметром max_depth =', best_depth_DT_up)
if (max_f1_up == best_f1_RF_up) and (max_aucroc_up == best_aucroc_RF_up):
    print('Лучшая модель: Random Forest (Случайный лес), с гиперпараметрами n_estimators =', best_est_RF_up,
      'max_depth =', best_depth_RF_up)
if (max_f1_up == best_f1_LR_up) and (max_aucroc_up == best_aucroc_LR_up):
    print('Лучшая модель: Logistic Regression (Логистическая регрессия), с гиперпараметром solver =', best_solver_LR_up)
print('F1-мера =', max_f1_up, ', AUC-ROC =', max_aucroc_up)
```

```
Лучшая модель: Random Forest (Случайный лес), с гиперпараметрами n_estimators = 30 max_depth = 7
F1-мера = 0.6233480176211453 , AUC-ROC = 0.8771498611310323
```

Обучив три модели с применением техники "upsampling" мы определили лучшую модель:

- Random Forest (Случайный лес), с гиперпараметрами:
  - n_estimators = 30
  - max_depth = 7
- F1-мера = 0.6233480176211453
- AUC-ROC = 0.8771498611310323

F1-мера нашей лучшей модели (0.623) больше поставленной задачи (0.59)

Уменьшим объекты частого класса (техника downsampling)

In [37]:
```
def downsample(features, target, fraction):
    features_zeros = features[target == 0]
    features_ones = features[target == 1]
    target_zeros = target[target == 0]
    target_ones = target[target == 1]

    features_downsampled = pd.concat(
        [features_zeros.sample(frac=fraction, random_state=r_state)] + [features_ones])
    target_downsampled = pd.concat(
        [target_zeros.sample(frac=fraction, random_state=r_state)] + [target_ones])
    features_downsampled, target_downsampled = shuffle(features_downsampled, target_downsampled,
                                       random_state=r_state)
    return features_downsampled, target_downsampled

features_downsampled, target_downsampled = downsample(features_train, target_train, 0.25)
```

Проверим баланс классов в целевом признаке

In [38]:
```
p_exited_0 = target_downsampled[target_downsampled == 0].count()/len(target_downsampled)
p_exited_1 = target_downsampled[target_downsampled == 1].count()/len(target_downsampled)
print('Соотношение оставшихся и ушедших клиентов:', )
print('{}% / {}%'.format(round(p_exited_0*100, 1), round(p_exited_1*100, 1)))
```

```
Соотношение оставшихся и ушедших клиентов:
49.4% / 50.6%
```

Decision Tree (Дерево решений)

In [39]:
```
best_f1_DT_down = 0
best_aucroc_DT_down = 0
best_depth_DT_down = 0


for depth in range(1, 21):
    model_decision_tree = DecisionTreeClassifier(random_state=r_state, max_depth=depth)
    model_decision_tree.fit(features_downsampled, target_downsampled)
    f1, aucroc = test_f1_aucroc(model_decision_tree)
    print('Глубина =', depth, ', F1-мера =', f1, ', AUC-ROC =', aucroc)
    if ((f1 > best_f1_DT_down) and (aucroc > best_aucroc_DT_down)):
        best_f1_DT_down = f1
        best_aucroc_DT_down = aucroc
        best_depth_DT_down = depth

print()
print('Лучший результат:')
print('Глубина =', best_depth_DT_down, ', F1-мера =', best_f1_DT_down, ', AUC-ROC =', best_aucroc_DT_down)
```

```
Глубина = 1 , F1-мера = 0.5092402464065708 , AUC-ROC = 0.7115642178165812
Глубина = 2 , F1-мера = 0.5376128385155466 , AUC-ROC = 0.7696144267254307
Глубина = 3 , F1-мера = 0.5376128385155466 , AUC-ROC = 0.8137721505782947
Глубина = 4 , F1-мера = 0.5565388397246804 , AUC-ROC = 0.8254330457848471
Глубина = 5 , F1-мера = 0.5667627281460134 , AUC-ROC = 0.8535765977382335
Глубина = 6 , F1-мера = 0.5688442211055276 , AUC-ROC = 0.8456375026311528
Глубина = 7 , F1-мера = 0.5686274509803922 , AUC-ROC = 0.8348921553469675
Глубина = 8 , F1-мера = 0.5505735140771637 , AUC-ROC = 0.8049864297729106
Глубина = 9 , F1-мера = 0.5240532241555783 , AUC-ROC = 0.7687379968221266
Глубина = 10 , F1-мера = 0.5390625 , AUC-ROC = 0.7506328736655634
Глубина = 11 , F1-мера = 0.5138339920948616 , AUC-ROC = 0.7414401391856373
Глубина = 12 , F1-мера = 0.5204957102001907 , AUC-ROC = 0.7319456371300785
Глубина = 13 , F1-мера = 0.5153256704980843 , AUC-ROC = 0.7298323327192424
Глубина = 14 , F1-мера = 0.5041705282669139 , AUC-ROC = 0.7206889614538491
Глубина = 15 , F1-мера = 0.5061147695202258 , AUC-ROC = 0.7164511760553016
Глубина = 16 , F1-мера = 0.49682683590208526 , AUC-ROC = 0.7109541629954715
Глубина = 17 , F1-мера = 0.5018315018315018 , AUC-ROC = 0.714563265944784
Глубина = 18 , F1-мера = 0.5096596136154553 , AUC-ROC = 0.7215346557707462
Глубина = 19 , F1-мера = 0.5004582951420715 , AUC-ROC = 0.7134670300295993
Глубина = 20 , F1-мера = 0.5004582951420715 , AUC-ROC = 0.7134670300295993

Лучший результат:
Глубина = 5 , F1-мера = 0.5667627281460134 , AUC-ROC = 0.8535765977382335
```

Random Forest (Случайный лес)

In [40]:
```
best_f1_RF_down = 0
best_aucroc_RF_down = 0
best_est_RF_down = 0
best_depth_RF_down = 0

for est in range(10, 51, 10):
    for depth in range(1, 21):
        model_random_forest = RandomForestClassifier(random_state=r_state, n_estimators=est, max_depth=depth)
        model_random_forest.fit(features_downsampled, target_downsampled)
        f1, aucroc = test_f1_aucroc(model_random_forest)
        print('Количество деревьев =', est,', глубина =', depth, ', F1-мера =', f1, ', AUC-ROC =', aucroc)
        if ((f1 > best_f1_RF_down) and (aucroc > best_aucroc_RF_down)):
            best_f1_RF_down = f1
            best_aucroc_RF_down = aucroc
            best_est_RF_down = est
            best_depth_RF_down = depth

print()
print('Лучший результат:')
print('Количество деревьев =', best_est_RF_down,', глубина =', best_depth_RF_down, ', F1-мера =', best_f1_RF_down,
      ', AUC-ROC =', best_aucroc_RF_down)
```

```
Количество деревьев = 10 , глубина = 1 , F1-мера = 0.544256120527307 , AUC-ROC = 0.8412963338965086
Количество деревьев = 10 , глубина = 2 , F1-мера = 0.55435847208619 , AUC-ROC = 0.8380746856122061
Количество деревьев = 10 , глубина = 3 , F1-мера = 0.5577841451766954 , AUC-ROC = 0.8497253356232898
Количество деревьев = 10 , глубина = 4 , F1-мера = 0.596153846153846 , AUC-ROC = 0.8595914588599519
Количество деревьев = 10 , глубина = 5 , F1-мера = 0.5953389830508474 , AUC-ROC = 0.8621508949643932
Количество деревьев = 10 , глубина = 6 , F1-мера = 0.5822267620020428 , AUC-ROC = 0.8650642560032188
Количество деревьев = 10 , глубина = 7 , F1-мера = 0.593654042988741 , AUC-ROC = 0.8607258814127938
Количество деревьев = 10 , глубина = 8 , F1-мера = 0.5932028836251287 , AUC-ROC = 0.8648314106516504
Количество деревьев = 10 , глубина = 9 , F1-мера = 0.6038894575230297 , AUC-ROC = 0.862370700976274
Количество деревьев = 10 , глубина = 10 , F1-мера = 0.5877551020408163 , AUC-ROC = 0.8623781520275242
Количество деревьев = 10 , глубина = 11 , F1-мера = 0.5913757700205338 , AUC-ROC = 0.8567777556315976
Количество деревьев = 10 , глубина = 12 , F1-мера = 0.5966303270564917 , AUC-ROC = 0.8626687430262817
Количество деревьев = 10 , глубина = 13 , F1-мера = 0.5926680244399186 , AUC-ROC = 0.8602108274951242
Количество деревьев = 10 , глубина = 14 , F1-мера = 0.5784615384615385 , AUC-ROC = 0.8487455223838893
Количество деревьев = 10 , глубина = 15 , F1-мера = 0.5972660357518402 , AUC-ROC = 0.8540692985021524
Количество деревьев = 10 , глубина = 16 , F1-мера = 0.5972369819341126 , AUC-ROC = 0.8490659175876476
Количество деревьев = 10 , глубина = 17 , F1-мера = 0.5912486659551761 , AUC-ROC = 0.8420572725054346
Количество деревьев = 10 , глубина = 18 , F1-мера = 0.583864118895966 , AUC-ROC = 0.8493388123396859
Количество деревьев = 10 , глубина = 19 , F1-мера = 0.5851063829787234 , AUC-ROC = 0.8461646645071036
Количество деревьев = 10 , глубина = 20 , F1-мера = 0.5786694825765575 , AUC-ROC = 0.8492512624874962
Количество деревьев = 20 , глубина = 1 , F1-мера = 0.5332120109190173 , AUC-ROC = 0.8300489720343418
Количество деревьев = 20 , глубина = 2 , F1-мера = 0.5434173669467787 , AUC-ROC = 0.8332817968955195
Количество деревьев = 20 , глубина = 3 , F1-мера = 0.5620155038759691 , AUC-ROC = 0.8504136264825264
Количество деревьев = 20 , глубина = 4 , F1-мера = 0.5913757700205338 , AUC-ROC = 0.8614747120634381
Количество деревьев = 20 , глубина = 5 , F1-мера = 0.5918153200419727 , AUC-ROC = 0.8714200027196337
Количество деревьев = 20 , глубина = 6 , F1-мера = 0.5922836287799792 , AUC-ROC = 0.8715429450652621
Количество деревьев = 20 , глубина = 7 , F1-мера = 0.6018808777429467 , AUC-ROC = 0.8734196785989043
Количество деревьев = 20 , глубина = 8 , F1-мера = 0.608421052631579 , AUC-ROC = 0.8713725022679136
Количество деревьев = 20 , глубина = 9 , F1-мера = 0.5960945529290853 , AUC-ROC = 0.8697705262491224
Количество деревьев = 20 , глубина = 10 , F1-мера = 0.6051813471502591 , AUC-ROC = 0.8715820630843253
Количество деревьев = 20 , глубина = 11 , F1-мера = 0.6102403343782654 , AUC-ROC = 0.8681555108906428
Количество деревьев = 20 , глубина = 12 , F1-мера = 0.6072874493927125 , AUC-ROC = 0.8689881658678519
Количество деревьев = 20 , глубина = 13 , F1-мера = 0.5973763874873865 , AUC-ROC = 0.8699269983253763
Количество деревьев = 20 , глубина = 14 , F1-мера = 0.5885947046843176 , AUC-ROC = 0.8619655500645447
Количество деревьев = 20 , глубина = 15 , F1-мера = 0.6000000000000001 , AUC-ROC = 0.8640220402095982
Количество деревьев = 20 , глубина = 16 , F1-мера = 0.5915201654601862 , AUC-ROC = 0.8624051620883062
Количество деревьев = 20 , глубина = 17 , F1-мера = 0.6006389776357828 , AUC-ROC = 0.8622216799512701
Количество деревьев = 20 , глубина = 18 , F1-мера = 0.606372045220966 , AUC-ROC = 0.866356082013721
Количество деревьев = 20 , глубина = 19 , F1-мера = 0.5964912280701754 , AUC-ROC = 0.8616730963029746
Количество деревьев = 20 , глубина = 20 , F1-мера = 0.6031413612565445 , AUC-ROC = 0.8669456464438927
Количество деревьев = 30 , глубина = 1 , F1-мера = 0.518918918918919 , AUC-ROC = 0.8210304058773893
Количество деревьев = 30 , глубина = 2 , F1-мера = 0.5503731343283582 , AUC-ROC = 0.8399700467739741
Количество деревьев = 30 , глубина = 3 , F1-мера = 0.5515911282545805 , AUC-ROC = 0.8515731963333377
Количество деревьев = 30 , глубина = 4 , F1-мера = 0.5879959308240081 , AUC-ROC = 0.863189385232389
Количество деревьев = 30 , глубина = 5 , F1-мера = 0.5869120654396728 , AUC-ROC = 0.8684488960336191
Количество деревьев = 30 , глубина = 6 , F1-мера = 0.5931321540062434 , AUC-ROC = 0.8709300960999335
Количество деревьев = 30 , глубина = 7 , F1-мера = 0.6121593291404612 , AUC-ROC = 0.8773426570821311
Количество деревьев = 30 , глубина = 8 , F1-мера = 0.5970772442588727 , AUC-ROC = 0.8728804087646715
Количество деревьев = 30 , глубина = 9 , F1-мера = 0.6088794926004228 , AUC-ROC = 0.8729279092163916
Количество деревьев = 30 , глубина = 10 , F1-мера = 0.5968253968253968 , AUC-ROC = 0.8738006135940705
Количество деревьев = 30 , глубина = 11 , F1-мера = 0.60625 , AUC-ROC = 0.8681946289097064
Количество деревьев = 30 , глубина = 12 , F1-мера = 0.6139630390143737 , AUC-ROC = 0.8745298852351832
Количество деревьев = 30 , глубина = 13 , F1-мера = 0.5873983739837398 , AUC-ROC = 0.8706273971428944
Количество деревьев = 30 , глубина = 14 , F1-мера = 0.5835866261398177 , AUC-ROC = 0.8625392810108097
Количество деревьев = 30 , глубина = 15 , F1-мера = 0.6078028747433265 , AUC-ROC = 0.8666569182079478
Количество деревьев = 30 , глубина = 16 , F1-мера = 0.5872855701311805 , AUC-ROC = 0.8630971784731678
Количество деревьев = 30 , глубина = 17 , F1-мера = 0.6114649681528662 , AUC-ROC = 0.8655318094691685
Количество деревьев = 30 , глубина = 18 , F1-мера = 0.5960539979231568 , AUC-ROC = 0.8663458368182521
```

```
Количество деревьев = 30 , глубина = 19 , F1-мера = 0.5983606557377049 , AUC-ROC = 0.862068002019235
Количество деревьев = 30 , глубина = 20 , F1-мера = 0.6081504702194358 , AUC-ROC = 0.8647736650044614
Количество деревьев = 40 , глубина = 1 , F1-мера = 0.5445829338446788 , AUC-ROC = 0.8328300769134765
Количество деревьев = 40 , глубина = 2 , F1-мера = 0.5552297165200392 , AUC-ROC = 0.8474434511779181
Количество деревьев = 40 , глубина = 3 , F1-мера = 0.5574425574425576 , AUC-ROC = 0.8541037596141847
Количество деревьев = 40 , глубина = 4 , F1-мера = 0.5972660357518402 , AUC-ROC = 0.8649403822761843
Количество деревьев = 40 , глубина = 5 , F1-мера = 0.5973223480947477 , AUC-ROC = 0.8668096647585765
Количество деревьев = 40 , глубина = 6 , F1-мера = 0.593521421107628 , AUC-ROC = 0.8723653548470021
Количество деревьев = 40 , глубина = 7 , F1-мера = 0.6071428571428572 , AUC-ROC = 0.8773855006268196
Количество деревьев = 40 , глубина = 8 , F1-мера = 0.6077003121748179 , AUC-ROC = 0.8744199822292428
Количество деревьев = 40 , глубина = 9 , F1-мера = 0.6106382978723405 , AUC-ROC = 0.8750644981623845
Количество деревьев = 40 , глубина = 10 , F1-мера = 0.6061899679829242 , AUC-ROC = 0.8759269573445942
Количество деревьев = 40 , глубина = 11 , F1-мера = 0.6047966631908238 , AUC-ROC = 0.8701002352669432
Количество деревьев = 40 , глубина = 12 , F1-мера = 0.6074380165289257 , AUC-ROC = 0.8737652211006319
Количество деревьев = 40 , глубина = 13 , F1-мера = 0.5895598771750256 , AUC-ROC = 0.8712160301916598
Количество деревьев = 40 , глубина = 14 , F1-мера = 0.5967078189300411 , AUC-ROC = 0.8654917600686987
Количество деревьев = 40 , глубина = 15 , F1-мера = 0.6078028747433265 , AUC-ROC = 0.8701160687508499
Количество деревьев = 40 , глубина = 16 , F1-мера = 0.5969230769230769 , AUC-ROC = 0.8664827498849745
Количество деревьев = 40 , глубина = 17 , F1-мера = 0.6140904311251315 , AUC-ROC = 0.8702920998366357
Количество деревьев = 40 , глубина = 18 , F1-мера = 0.6020833333333334 , AUC-ROC = 0.87006856829913
Количество деревьев = 40 , глубина = 19 , F1-мера = 0.6061224489795918 , AUC-ROC = 0.8686174760681546
Количество деревьев = 40 , глубина = 20 , F1-мера = 0.6041237113402061 , AUC-ROC = 0.8697789086817784
Количество деревьев = 50 , глубина = 1 , F1-мера = 0.5468451242829828 , AUC-ROC = 0.8312709444393737
Количество деревьев = 50 , глубина = 2 , F1-мера = 0.5474095796676443 , AUC-ROC = 0.846485059710862
Количество деревьев = 50 , глубина = 3 , F1-мера = 0.5546719681908548 , AUC-ROC = 0.8542052801874684
Количество деревьев = 50 , глубина = 4 , F1-мера = 0.5882352941176471 , AUC-ROC = 0.8649310684621216
Количество деревьев = 50 , глубина = 5 , F1-мера = 0.5889698231009366 , AUC-ROC = 0.8672874634199953
Количество деревьев = 50 , глубина = 6 , F1-мера = 0.5951629863301787 , AUC-ROC = 0.8741033125511095
Количество деревьев = 50 , глубина = 7 , F1-мера = 0.6079664570230607 , AUC-ROC = 0.8780132516946487
Количество деревьев = 50 , глубина = 8 , F1-мера = 0.6048472075869337 , AUC-ROC = 0.8751501852517617
Количество деревьев = 50 , глубина = 9 , F1-мера = 0.6096256684491977 , AUC-ROC = 0.8771414786983758
Количество деревьев = 50 , глубина = 10 , F1-мера = 0.6042553191489363 , AUC-ROC = 0.8763479417402302
Количество деревьев = 50 , глубина = 11 , F1-мера = 0.6002055498458376 , AUC-ROC = 0.8708900466994637
Количество деревьев = 50 , глубина = 12 , F1-мера = 0.6008230452674898 , AUC-ROC = 0.8739794388240752
Количество деревьев = 50 , глубина = 13 , F1-мера = 0.6008230452674898 , AUC-ROC = 0.8737223775559434
Количество деревьев = 50 , глубина = 14 , F1-мера = 0.6029106029106029 , AUC-ROC = 0.8682728649478333
Количество деревьев = 50 , глубина = 15 , F1-мера = 0.6084275436793423 , AUC-ROC = 0.8731961470613986
Количество деревьев = 50 , глубина = 16 , F1-мера = 0.5967078189300411 , AUC-ROC = 0.868744143939408
Количество деревьев = 50 , глубина = 17 , F1-мера = 0.6058091286307055 , AUC-ROC = 0.8742020389801746
Количество деревьев = 50 , глубина = 18 , F1-мера = 0.597107438016529 , AUC-ROC = 0.8715019642833858
Количество деревьев = 50 , глубина = 19 , F1-мера = 0.6026557711950971 , AUC-ROC = 0.8719797629448045
Количество деревьев = 50 , глубина = 20 , F1-мера = 0.607621009268795 , AUC-ROC = 0.8729614389470175

Лучший результат:
Количество деревьев = 30 , глубина = 7 , F1-мера = 0.6121593291404612 , AUC-ROC = 0.8773426570821311
```

Logistic Regression (Логистическая регрессия)

In [41]:
```
best_f1_LR_down = 0
best_aucroc_LR_down = 0
best_solver_LR_down = ''
solvers = ['lbfgs', 'liblinear', 'newton-cg', 'sag', 'saga']

for solver in solvers:
    model_logistic_regression = LogisticRegression(solver=solver, random_state=r_state)
    model_logistic_regression.fit(features_downsampled, target_downsampled)
    f1, aucroc = test_f1_aucroc(model_logistic_regression)
    print('Алгоритм =', solver, ', F1-мера =', f1, ', AUC-ROC =', aucroc)
    if ((f1 > best_f1_LR_down) and (aucroc > best_aucroc_LR_down)):
        best_f1_LR_down = f1
        best_aucroc_LR_down = aucroc
        best_solver_LR_down = solver

print()
print('Лучший результат:')
print('Алгоритм =', best_solver_LR_down, ', F1-мера =', best_f1_LR_down, ', AUC-ROC =', best_aucroc_LR_down)
```

```
Алгоритм = lbfgs , F1-мера = 0.5078196872125115 , AUC-ROC = 0.7911339941173949
Алгоритм = liblinear , F1-мера = 0.5096596136154553 , AUC-ROC = 0.7911153664892695
Алгоритм = newton-cg , F1-мера = 0.5078196872125115 , AUC-ROC = 0.7911414451686454
Алгоритм = sag , F1-мера = 0.5078196872125115 , AUC-ROC = 0.7911395824058327
Алгоритм = saga , F1-мера = 0.5078196872125115 , AUC-ROC = 0.7911451706942704

Лучший результат:
Алгоритм = lbfgs , F1-мера = 0.5078196872125115 , AUC-ROC = 0.7911339941173949
```

Определим лучшую модель

In [42]:
```
max_f1_down = best_f1_DT_down
max_aucroc_down = best_aucroc_DT_down
if (max_f1_down < best_f1_RF_down) and (max_aucroc_down < best_aucroc_RF_down):
    max_f1_down = best_f1_RF_down
    max_aucroc_down = best_aucroc_RF_down
```

```
    if (max_f1_down < best_f1_LR_down) and (max_aucroc_down < best_aucroc_LR_down):
        max_f1_down = best_f1_LR_down
        max_aucroc_down = best_aucroc_LR_down

    if (max_f1_down == best_f1_DT_down) and (max_aucroc_down == best_aucroc_DT_down):
        print('Лучшая модель: Decision Tree (Дерево решений), с гиперпараметром max_depth =', best_depth_DT_down)
    if (max_f1_down == best_f1_RF_down) and (max_aucroc_down == best_aucroc_RF_down):
        print('Лучшая модель: Random Forest (Случайный лес), с гиперпараметрами n_estimators =', best_est_RF_down,
          'max_depth =', best_depth_RF_down)
    if (max_f1_down == best_f1_LR_down) and (max_aucroc_down == best_aucroc_LR_down):
        print('Лучшая модель: Logistic Regression (Логистическая регрессия), с гиперпараметром solver =', best_solver_LR_down)
    print('F1-мера =', max_f1_down, ', AUC-ROC =', max_aucroc_down)
```

Лучшая модель: Random Forest (Случайный лес), с гиперпараметрами n_estimators = 30 max_depth = 7
F1-мера = 0.6121593291404612 , AUC-ROC = 0.8773426570821311
Обучив три модели с применением техники "downsampling" мы определили лучшую модель:

- Random Forest (Случайный лес), с гиперпараметрами:
  - n_estimators = 30
  - max_depth = 7
- F1-мера = 0.6121593291404612
- AUC-ROC = 0.8773426570821311

F1-мера нашей лучшей модели (0.612) больше поставленной задачи (0.59)

Аргумент class_weight = 'balanced'

Decision Tree (Дерево решений)

In [43]:
```
best_f1_DT_bal = 0
best_aucroc_DT_bal = 0
best_depth_DT_bal = 0


for depth in range(1, 21):
    model_decision_tree = DecisionTreeClassifier(random_state=r_state, max_depth=depth, class_weight='balanced')
    model_decision_tree.fit(features_train, target_train)
    f1, aucroc = test_f1_aucroc(model_decision_tree)
    print('Глубина =', depth, ', F1-мера =', f1, ', AUC-ROC =', aucroc)
    if ((f1 > best_f1_DT_bal) and (aucroc > best_aucroc_DT_bal)):
        best_f1_DT_bal = f1
        best_aucroc_DT_bal = aucroc
        best_depth_DT_bal = depth

print()
print('Лучший результат:')
print('Глубина =', best_depth_DT_bal, ', F1-мера =', best_f1_DT_bal, ', AUC-ROC =', best_aucroc_DT_bal)
```

```
Глубина = 1 , F1-мера = 0.5092402464065708 , AUC-ROC = 0.7115642178165812
Глубина = 2 , F1-мера = 0.5376128385155466 , AUC-ROC = 0.7696144267254307
Глубина = 3 , F1-мера = 0.5376128385155466 , AUC-ROC = 0.8123005679563815
Глубина = 4 , F1-мера = 0.5522682445759368 , AUC-ROC = 0.8276497335317796
Глубина = 5 , F1-мера = 0.5711481844946025 , AUC-ROC = 0.8466415317871161
Глубина = 6 , F1-мера = 0.583084577114428 , AUC-ROC = 0.8353643657199484
Глубина = 7 , F1-мера = 0.5622568093385214 , AUC-ROC = 0.8312374147087478
Глубина = 8 , F1-мера = 0.5813449023861171 , AUC-ROC = 0.815457950923651
Глубина = 9 , F1-мера = 0.5514950166112956 , AUC-ROC = 0.786207917859611
Глубина = 10 , F1-мера = 0.5290889132821075 , AUC-ROC = 0.7434500602603771
Глубина = 11 , F1-мера = 0.5345838218053927 , AUC-ROC = 0.735072284510941
Глубина = 12 , F1-мера = 0.5152941176470588 , AUC-ROC = 0.7174244696248582
Глубина = 13 , F1-мера = 0.49526066350710907 , AUC-ROC = 0.715544941946997
Глубина = 14 , F1-мера = 0.5173267326732675 , AUC-ROC = 0.7135089421928815
Глубина = 15 , F1-мера = 0.5063291139240506 , AUC-ROC = 0.699595035364552
Глубина = 16 , F1-мера = 0.503209242618742 , AUC-ROC = 0.6947239106097381
Глубина = 17 , F1-мера = 0.5076923076923077 , AUC-ROC = 0.7025968776369736
Глубина = 18 , F1-мера = 0.5151915455746366 , AUC-ROC = 0.7010954908100597
Глубина = 19 , F1-мера = 0.5292553191489361 , AUC-ROC = 0.7056946521942414
Глубина = 20 , F1-мера = 0.5161290322580645 , AUC-ROC = 0.6962169150039956

Лучший результат:
Глубина = 5 , F1-мера = 0.5711481844946025 , AUC-ROC = 0.8466415317871161
```

Random Forest (Случайный лес)

In [44]:
```
best_f1_RF_bal = 0
best_aucroc_RF_bal = 0
best_est_RF_bal = 0
best_depth_RF_bal = 0

for est in range(10, 51, 10):
    for depth in range(1, 21):
        model_random_forest = RandomForestClassifier(random_state=r_state, n_estimators=est, max_depth=depth,
```

```
                                        class_weight='balanced')
            model_random_forest.fit(features_train, target_train)
            f1, aucroc = test_f1_aucroc(model_random_forest)
            print('Количество деревьев =', est,', глубина =', depth, ', F1-мера =', f1, ', AUC-ROC =', aucroc)
            if ((f1 > best_f1_RF_bal) and (aucroc > best_aucroc_RF_bal)):
                best_f1_RF_bal = f1
                best_aucroc_RF_bal = aucroc
                best_est_RF_bal = est
                best_depth_RF_bal = depth

        print()
        print('Лучший результат:')
        print('Количество деревьев =', best_est_RF_bal,', глубина =', best_depth_RF_bal, ', F1-мера =', best_f1_RF_bal,
            ', AUC-ROC =', best_aucroc_RF_bal)
```

```
Количество деревьев = 10 , глубина = 1 , F1-мера = 0.5471923536439666 , AUC-ROC = 0.8064831596927933
Количество деревьев = 10 , глубина = 2 , F1-мера = 0.5647840531561461 , AUC-ROC = 0.8441249392273633
Количество деревьев = 10 , глубина = 3 , F1-мера = 0.5668016194331983 , AUC-ROC = 0.854865629604517
Количество деревьев = 10 , глубина = 4 , F1-мера = 0.6022988505747127 , AUC-ROC = 0.860454849423568
Количество деревьев = 10 , глубина = 5 , F1-мера = 0.6 , AUC-ROC = 0.8662228944726239
Количество деревьев = 10 , глубина = 6 , F1-мера = 0.6132075471698113 , AUC-ROC = 0.8655727902510445
Количество деревьев = 10 , глубина = 7 , F1-мера = 0.6210153482880756 , AUC-ROC = 0.8615976544090664
Количество деревьев = 10 , глубина = 8 , F1-мера = 0.6219512195121951 , AUC-ROC = 0.8639978242930351
Количество деревьев = 10 , глубина = 9 , F1-мера = 0.621454993834772 , AUC-ROC = 0.8615156928453144
Количество деревьев = 10 , глубина = 10 , F1-мера = 0.6131578947368421 , AUC-ROC = 0.8546961181885749
Количество деревьев = 10 , глубина = 11 , F1-мера = 0.5983379501385042 , AUC-ROC = 0.8509379942142588
Количество деревьев = 10 , глубина = 12 , F1-мера = 0.6107091172214183 , AUC-ROC = 0.849678766552976
Количество деревьев = 10 , глубина = 13 , F1-мера = 0.5869894099848714 , AUC-ROC = 0.8532599280601002
Количество деревьев = 10 , глубина = 14 , F1-мера = 0.5810593900481541 , AUC-ROC = 0.8340259706391326
Количество деревьев = 10 , глубина = 15 , F1-мера = 0.5736925515055468 , AUC-ROC = 0.8292181798199453
Количество деревьев = 10 , глубина = 16 , F1-мера = 0.5805369127516777 , AUC-ROC = 0.8419082514804306
Количество деревьев = 10 , глубина = 17 , F1-мера = 0.541455160744501 , AUC-ROC = 0.8343966604388297
Количество деревьев = 10 , глубина = 18 , F1-мера = 0.54421768707483 , AUC-ROC = 0.8258111866357944
Количество деревьев = 10 , глубина = 19 , F1-мера = 0.5616438356164384 , AUC-ROC = 0.8370334011999918
Количество деревьев = 10 , глубина = 20 , F1-мера = 0.5617597292724197 , AUC-ROC = 0.8377021330496967
Количество деревьев = 20 , глубина = 1 , F1-мера = 0.5604838709677419 , AUC-ROC = 0.8310408932320239
Количество деревьев = 20 , глубина = 2 , F1-мера = 0.5511961722488039 , AUC-ROC = 0.8363898166482564
Количество деревьев = 20 , глубина = 3 , F1-мера = 0.5656970912738215 , AUC-ROC = 0.8575778122595872
Количество деревьев = 20 , глубина = 4 , F1-мера = 0.5961123110151189 , AUC-ROC = 0.862418201427994
Количество деревьев = 20 , глубина = 5 , F1-мера = 0.6064814814814814 , AUC-ROC = 0.8707102900880528
Количество деревьев = 20 , глубина = 6 , F1-мера = 0.6095017381228274 , AUC-ROC = 0.8674932987107817
Количество деревьев = 20 , глубина = 7 , F1-мера = 0.6153846153846154 , AUC-ROC = 0.8682775218548647
Количество деревьев = 20 , глубина = 8 , F1-мера = 0.6324786324786325 , AUC-ROC = 0.8693905226353623
Количество деревьев = 20 , глубина = 9 , F1-мера = 0.6362484157160964 , AUC-ROC = 0.8666336336727909
Количество деревьев = 20 , глубина = 10 , F1-мера = 0.6236842105263157 , AUC-ROC = 0.8629323239642575
Количество деревьев = 20 , глубина = 11 , F1-мера = 0.6255259467040674 , AUC-ROC = 0.8610546590492086
Количество деревьев = 20 , глубина = 12 , F1-мера = 0.6169590643274854 , AUC-ROC = 0.860760342524826
Количество деревьев = 20 , глубина = 13 , F1-мера = 0.6170542635658914 , AUC-ROC = 0.8676069272423473
Количество деревьев = 20 , глубина = 14 , F1-мера = 0.5911949685534591 , AUC-ROC = 0.8539016498490231
Количество деревьев = 20 , глубина = 15 , F1-мера = 0.5889967637540453 , AUC-ROC = 0.847666982715424
Количество деревьев = 20 , глубина = 16 , F1-мера = 0.5963756177924218 , AUC-ROC = 0.8573114371773928
Количество деревьев = 20 , глубина = 17 , F1-мера = 0.5767284991568297 , AUC-ROC = 0.8632359543027026
Количество деревьев = 20 , глубина = 18 , F1-мера = 0.5823627287853579 , AUC-ROC = 0.8491134180393676
Количество деревьев = 20 , глубина = 19 , F1-мера = 0.5733558178752107 , AUC-ROC = 0.854784599422171
Количество деревьев = 20 , глубина = 20 , F1-мера = 0.5544217687074829 , AUC-ROC = 0.8625886442253421
Количество деревьев = 30 , глубина = 1 , F1-мера = 0.5195797516714422 , AUC-ROC = 0.818400184786071
Количество деревьев = 30 , глубина = 2 , F1-мера = 0.5580448065173116 , AUC-ROC = 0.8409470658691559
Количество деревьев = 30 , глубина = 3 , F1-мера = 0.5548780487804877 , AUC-ROC = 0.8532506142460373
Количество деревьев = 30 , глубина = 4 , F1-мера = 0.5937834941050375 , AUC-ROC = 0.8601586701363728
Количество деревьев = 30 , глубина = 5 , F1-мера = 0.6152046783625731 , AUC-ROC = 0.8714432872547906
Количество деревьев = 30 , глубина = 6 , F1-мера = 0.608 , AUC-ROC = 0.8716481911641709
Количество деревьев = 30 , глубина = 7 , F1-мера = 0.6108490566037736 , AUC-ROC = 0.8707186725207091
Количество деревьев = 30 , глубина = 8 , F1-мера = 0.6322263222632226 , AUC-ROC = 0.8695134649809905
Количество деревьев = 30 , глубина = 9 , F1-мера = 0.6245161290322581 , AUC-ROC = 0.8703815124516381
Количество деревьев = 30 , глубина = 10 , F1-мера = 0.6258322237017311 , AUC-ROC = 0.8651322468458769
Количество деревьев = 30 , глубина = 11 , F1-мера = 0.6402266288951841 , AUC-ROC = 0.8661250994249651
Количество деревьев = 30 , глубина = 12 , F1-мера = 0.6194690265486725 , AUC-ROC = 0.863807822486155
Количество деревьев = 30 , глубина = 13 , F1-мера = 0.6149068322981367 , AUC-ROC = 0.8732948734904634
Количество деревьев = 30 , глубина = 14 , F1-мера = 0.5930599369085174 , AUC-ROC = 0.8595132228218249
Количество деревьев = 30 , глубина = 15 , F1-мера = 0.586709886547812 , AUC-ROC = 0.8501686731726762
Количество деревьев = 30 , глубина = 16 , F1-мера = 0.5719063545150501 , AUC-ROC = 0.8622738373100214
Количество деревьев = 30 , глубина = 17 , F1-мера = 0.5795644891122278 , AUC-ROC = 0.8631223257711373
Количество деревьев = 30 , глубина = 18 , F1-мера = 0.5767284991568297 , AUC-ROC = 0.855141318500774
Количество деревьев = 30 , глубина = 19 , F1-мера = 0.5733788395904437 , AUC-ROC = 0.861253043288745
Количество деревьев = 30 , глубина = 20 , F1-мера = 0.5511265164644714 , AUC-ROC = 0.8619133927057934
Количество деревьев = 40 , глубина = 1 , F1-мера = 0.5388397246804326 , AUC-ROC = 0.8304280442666955
Количество деревьев = 40 , глубина = 2 , F1-мера = 0.5705263157894737 , AUC-ROC = 0.8511298587839511
Количество деревьев = 40 , глубина = 3 , F1-мера = 0.556935817805383 , AUC-ROC = 0.8580397774370991
Количество деревьев = 40 , глубина = 4 , F1-мера = 0.5919477693144722 , AUC-ROC = 0.8632173266745773
Количество деревьев = 40 , глубина = 5 , F1-мера = 0.6187717265353418 , AUC-ROC = 0.8720859404251198
Количество деревьев = 40 , глубина = 6 , F1-мера = 0.6091954022988506 , AUC-ROC = 0.8728841342902967
Количество деревьев = 40 , глубина = 7 , F1-мера = 0.6223776223776224 , AUC-ROC = 0.8717860356122995
Количество деревьев = 40 , глубина = 8 , F1-мера = 0.635118306351183 , AUC-ROC = 0.8714246596266652
```

```
Количество деревьев = 40 , глубина = 9 , F1-мера = 0.6428571428571429 , AUC-ROC = 0.872344864456064
Количество деревьев = 40 , глубина = 10 , F1-мера = 0.6283422459893049 , AUC-ROC = 0.8672856006571827
Количество деревьев = 40 , глубина = 11 , F1-мера = 0.628169014084507 , AUC-ROC = 0.8685876718631541
Количество деревьев = 40 , глубина = 12 , F1-мера = 0.6094674556213018 , AUC-ROC = 0.8667658898324817
Количество деревьев = 40 , глубина = 13 , F1-мера = 0.595679012345679 , AUC-ROC = 0.8714488755432283
Количество деревьев = 40 , глубина = 14 , F1-мера = 0.599049128367670З , AUC-ROC = 0.8624582508284638
Количество деревьев = 40 , глубина = 15 , F1-мера = 0.6 , AUC-ROC = 0.8580574736838184
Количество деревьев = 40 , глубина = 16 , F1-мера = 0.5771812080536912 , AUC-ROC = 0.8642446403656977
Количество деревьев = 40 , глубина = 17 , F1-мера = 0.5714285714285714 , AUC-ROC = 0.8639410100272523
Количество деревьев = 40 , глубина = 18 , F1-мера = 0.566610455311973 , AUC-ROC = 0.8597125384427676
Количество деревьев = 40 , глубина = 19 , F1-мера = 0.5782312925170068 , AUC-ROC = 0.8596315082604217
Количество деревьев = 40 , глубина = 20 , F1-мера = 0.555366269165247 , AUC-ROC = 0.8628373230608173
Количество деревьев = 50 , глубина = 1 , F1-мера = 0.5344295991778006 , AUC-ROC = 0.8286286153897738
Количество деревьев = 50 , глубина = 2 , F1-мера = 0.5578512396694215 , AUC-ROC = 0.8504695093669027
Количество деревьев = 50 , глубина = 3 , F1-мера = 0.5641025641025641 , AUC-ROC = 0.8598690105190216
Количество деревьев = 50 , глубина = 4 , F1-мера = 0.5937161430119177 , AUC-ROC = 0.8648994014943083
Количество деревьев = 50 , глубина = 5 , F1-мера = 0.6143344709897611 , AUC-ROC = 0.8721325094954334
Количество деревьев = 50 , глубина = 6 , F1-мера = 0.6171428571428572 , AUC-ROC = 0.8735724251495331
Количество деревьев = 50 , глубина = 7 , F1-мера = 0.6274970622796711 , AUC-ROC = 0.8715923082797944
Количество деревьев = 50 , глубина = 8 , F1-мера = 0.6394052044609666 , AUC-ROC = 0.8721399605466835
Количество деревьев = 50 , глубина = 9 , F1-мера = 0.6375321336760926 , AUC-ROC = 0.872171627514497
Количество деревьев = 50 , глубина = 10 , F1-мера = 0.6265060240963856 , AUC-ROC = 0.8702008244588209
Количество деревьев = 50 , глубина = 11 , F1-мера = 0.6348314606741572 , AUC-ROC = 0.868200217198144
Количество деревьев = 50 , глубина = 12 , F1-мера = 0.6149341142020497 , AUC-ROC = 0.8684777688572136
Количество деревьев = 50 , глубина = 13 , F1-мера = 0.5990783410138248 , AUC-ROC = 0.8721539312677777
Количество деревьев = 50 , глубина = 14 , F1-мера = 0.6078740157480315 , AUC-ROC = 0.8646879779150841
Количество деревьев = 50 , глубина = 15 , F1-мера = 0.5924713584288053 , AUC-ROC = 0.8638413522167809
Количество деревьев = 50 , глубина = 16 , F1-мера = 0.5876460767946577 , AUC-ROC = 0.8656351928052648
Количество деревьев = 50 , глубина = 17 , F1-мера = 0.5820642978003384 , AUC-ROC = 0.8653976905466649
Количество деревьев = 50 , глубина = 18 , F1-мера = 0.5704697986577181 , AUC-ROC = 0.8620810413589227
Количество деревьев = 50 , глубина = 19 , F1-мера = 0.5806451612903225 , AUC-ROC = 0.8617233908989134
Количество деревьев = 50 , глубина = 20 , F1-мера = 0.5675675675675677 , AUC-ROC = 0.864619987072426

Лучший результат:
Количество деревьев = 40 , глубина = 9 , F1-мера = 0.6428571428571429 , AUC-ROC = 0.872344864456064
```

Logistic Regression (Логистическая регрессия)

In [45]:
```
best_f1_LR_bal = 0
best_aucroc_LR_bal = 0
best_solver_LR_bal = ''
solvers = ['lbfgs', 'liblinear', 'newton-cg', 'sag', 'saga']

for solver in solvers:
    model_logistic_regression = LogisticRegression(solver=solver, random_state=r_state, class_weight='balanced')
    model_logistic_regression.fit(features_train, target_train)
    f1, aucroc = test_f1_aucroc(model_logistic_regression)
    print('Алгоритм =', solver, ', F1-мера =', f1, ', AUC-ROC =', aucroc)
    if ((f1 > best_f1_LR_bal) and (aucroc > best_aucroc_LR_bal)):
        best_f1_LR_bal = f1
        best_aucroc_LR_bal = aucroc
        best_solver_LR_bal = solver

print()
print('Лучший результат:')
print('Алгоритм =', best_solver_LR_bal, ', F1-мера =', best_f1_LR_bal, ', AUC-ROC =', best_aucroc_LR_bal)
```

```
Алгоритм = lbfgs , F1-мера = 0.5079365079365079 , AUC-ROC = 0.7907428139267598
Алгоритм = liblinear , F1-мера = 0.5065420560747663 , AUC-ROC = 0.790763304317698
Алгоритм = newton-cg , F1-мера = 0.5079365079365079 , AUC-ROC = 0.7907390884011347
Алгоритм = sag , F1-мера = 0.5079365079365079 , AUC-ROC = 0.7907428139267598
Алгоритм = saga , F1-мера = 0.5079365079365079 , AUC-ROC = 0.7907390884011349

Лучший результат:
Алгоритм = lbfgs , F1-мера = 0.5079365079365079 , AUC-ROC = 0.7907428139267598
```

Определим лучшую модель

In [46]:
```
max_f1_bal = best_f1_DT_bal
max_aucroc_bal = best_aucroc_DT_bal
if (max_f1_bal < best_f1_RF_bal) and (max_aucroc_bal < best_aucroc_RF_bal):
    max_f1_bal = best_f1_RF_bal
    max_aucroc_bal = best_aucroc_RF_bal
if (max_f1_bal < best_f1_LR_bal) and (max_aucroc_bal < best_aucroc_LR_bal):
    max_f1_bal = best_f1_LR_bal
    max_aucroc_bal = best_aucroc_LR_bal

if (max_f1_bal == best_f1_DT_bal) and (max_aucroc_bal == best_aucroc_DT_bal):
    print('Лучшая модель: Decision Tree (Дерево решений), с гиперпараметром max_depth =', best_depth_DT_bal)
if (max_f1_bal == best_f1_RF_bal) and (max_aucroc_bal == best_aucroc_RF_bal):
    print('Лучшая модель: Random Forest (Случайный лес), с гиперпараметрами n_estimators =', best_est_RF_bal,
      'max_depth =', best_depth_RF_bal)
if (max_f1_bal == best_f1_LR_bal) and (max_aucroc_bal == best_aucroc_LR_bal):
```

```
        print('Лучшая модель: Logistic Regression (Логистическая регрессия), с гиперпараметром solver =', best_solver_LR_bal)
    print('F1-мера =', max_f1_bal, ', AUC-ROC =', max_aucroc_bal)
```

Лучшая модель: Random Forest (Случайный лес), с гиперпараметрами n_estimators = 40 max_depth = 9
F1-мера = 0.6428571428571429 , AUC-ROC = 0.872344864456064
Обучив три модели с аргументом class_weight = 'balanced' мы определили лучшую модель:

- Random Forest (Случайный лес), с гиперпараметрами:
  - n_estimators = 50
  - max_depth = 9
- F1-мера = 0.6463878326996199
- AUC-ROC = 0.8749341047655059

F1-мера нашей лучшей модели (0.646) больше поставленной задачи (0.59)

Определим лучший метод балансировки

Сравнивать методы балансировки будем по метрике "F1-мера"

In [47]:
```
best_f1 = max_f1_up
if best_f1 < max_f1_down:
    best_f1 = max_f1_down
if best_f1 < max_f1_bal:
    best_f1 = max_f1_bal

if best_f1 == max_f1_up:
    print('Лучший метод балансировки: "upsampling"')
    print('F1-мера =', max_f1_up, ', AUC-ROC =', max_aucroc_up)
if best_f1 == max_f1_down:
    print('Лучший метод балансировки: "downsampling"')
    print('F1-мера =', max_f1_down, ', AUC-ROC =', max_aucroc_down)
if best_f1 == max_f1_bal:
    print('Лучший метод балансировки: Аргумент class_weight = "balanced"')
    print('F1-мера =', max_f1_bal, ', AUC-ROC =', max_aucroc_bal)
```

Лучший метод балансировки: Аргумент class_weight = "balanced"
F1-мера = 0.6428571428571429 , AUC-ROC = 0.872344864456064
По итогу исследования мы определили лучший метод балансировки (аргумент class_weight = 'balanced') и лучшую модель:

- Random Forest (Случайный лес), с гиперпараметрами:
  - n_estimators = 50
  - max_depth = 9
- F1-мера = 0.6463878326996199
- AUC-ROC = 0.8749341047655059

Тестирование модели

Объединим обучающую и валидационную выборки для обучения итоговой модели на большем количестве данных

In [48]:
```
features = pd.concat([features_train, features_valid])
target = pd.concat([target_train, target_valid])
```

Обучим итоговую модель и проверим результат на тестовой выборке

In [49]:
```
model = RandomForestClassifier(random_state=r_state, n_estimators=50, max_depth=9, class_weight='balanced')
model.fit(features, target)
predictions_test = model.predict(features_test)
probabilities_test = model.predict_proba(features_test)
probabilities_one_test = probabilities_test[:, 1]
f1 = f1_score(target_test, predictions_test)
aucroc = roc_auc_score(target_test, probabilities_one_test)


print('Результат на тестовой выборке:')
print('F1-мера =', f1,', AUC-ROC =', aucroc)
```

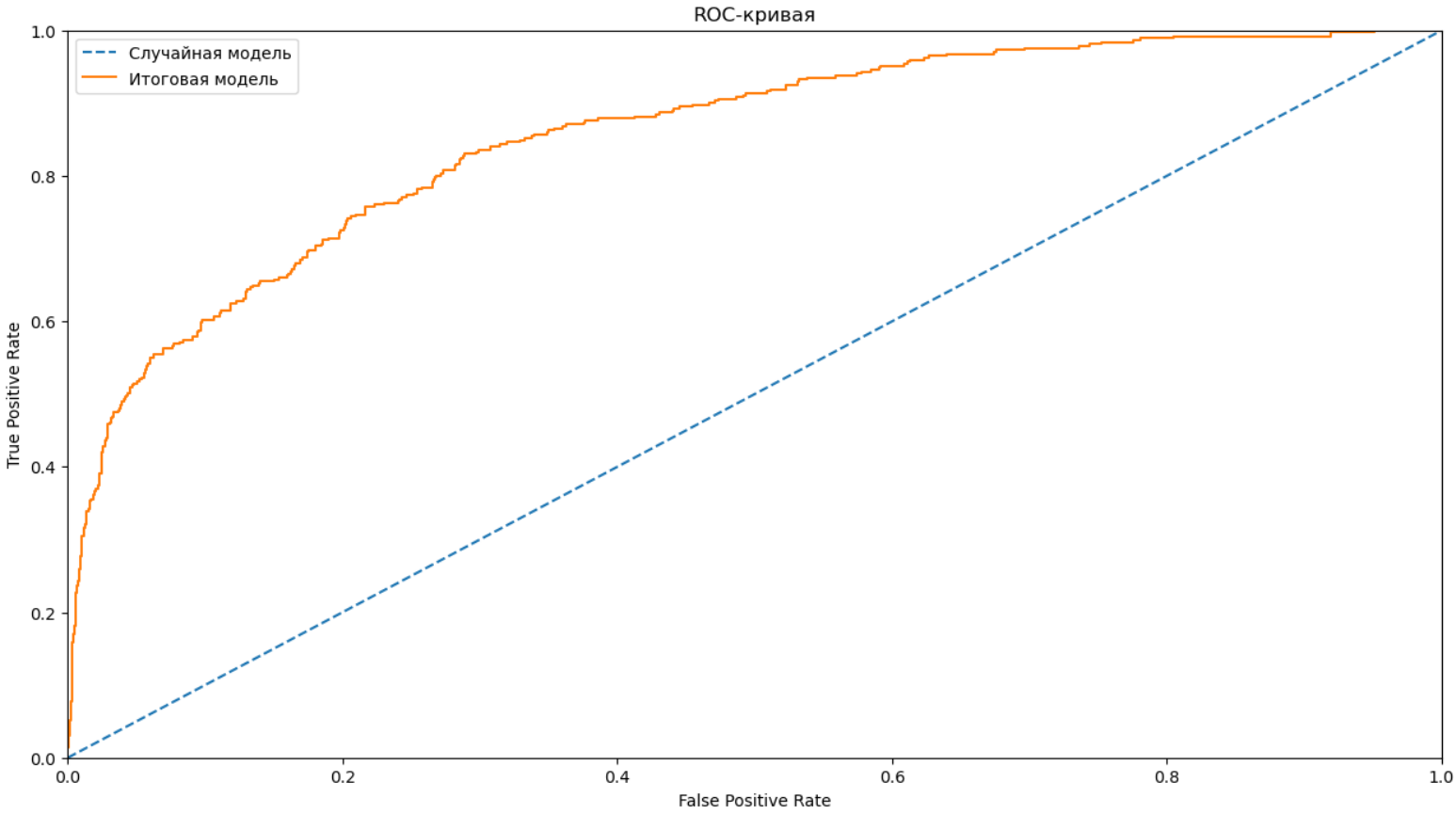
Результат на тестовой выборке:
F1-мера = 0.5987096774193549 , AUC-ROC = 0.8534515494929338
F1-мера нашей модели на тестовой выборке (0.593) чуть больше поставленной задачи (0.59)

Сравним AUC-ROC нашей модели с AUC-ROC случайной модели

In [50]:
```
const_target_predict = pd.Series([0]*len(target_test))
const_aucroc = roc_auc_score(target_test, const_target_predict)
print('AUC-ROC случайной модели =', const_aucroc)
print('AUC-ROC нашей модели =', round(aucroc, 2))
```

AUC-ROC случайной модели = 0.5
AUC-ROC нашей модели = 0.85
In [51]:
```
fpr_const, tpr_const, thresholds = roc_curve(target_test, const_target_predict)
fpr_test, tpr_test, thresholds = roc_curve(target_test, probabilities_one_test)
```

```
plt.figure(figsize=(15,8))
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.0])
plt.plot(fpr_const,tpr_const, linestyle='--')
plt.plot(fpr_test,tpr_test)
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC-кривая')
plt.legend(('Случайная модель','Итоговая модель'), loc= 'upper left')
plt.show()
```



Мы получили достаточно большое значение AUC-ROC на нашей итоговой модели. Итоговая модель является адекватной

Общий вывод

Проведено исследование для поиска модели с предельно большим значением F1-меры (не меньше 0.59)

Обучение модели проводилось с целью спрогнозировать, уйдёт клиент из банка в ближайшее время или нет

Данные о поведении клиентов были взяты из файла ***Churn.csv***. В нем представлены исторические данные о поведении клиентов и расторжении договоров с банком

Исследование проходило в четыре этапа:

- **Подготовка данных:**
  - Данные состоят из 10000 объектов
  - Данные имеют 14 признаков (1 целевой и 13 вспомогательных)
  - Явные дубликаты отсутствуют
  - Пропущенные значения есть в признаке "Tenure" - 909 шт. - 9.09% от общего
  - Мы удалили пропущенные значения, так как заменяя пропущенные значения средним или медианным есть риск повлиять на обучение модели
  - Мы удалили признаки "RowNumber", "CustomerId" и "Surname", так как они ни как не влияют на вероятность ухода клиента, но могут усложнить обучение модели
  - Аномалий в данных не обнаружено
  - Преобразовали данные методом OHE, чтобы исключить попадание в дамми-ловушку
  - Разделили данные на выборки для обучения:
    - Вспомогательные признаки:
      - features_train - Объектов: 5454 шт., признаков: 11 шт. - 59.99%
      - features_valid - Объектов: 1818 шт., признаков: 11 шт. - 20.0%
      - features_test - Объектов: 1819 шт., признаков: 11 шт. - 20.01%
    - Целевые признаки:
      - target_train - Объектов: 5454 шт. - 59.99%
      - target_valid - Объектов: 1818 шт. - 20.0%
      - target_test - Объектов: 1819 шт. - 20.01%
  - Стандартизировали численные признаки в выборках, так как в данных присутствуют значения в разных масштабах

- **Исследование задачи**

- Проверили баланс классов в целевом признаке:
  - Соотношение оставшихся и ушедших клиентов: 79.6% / 20.4%
  - Соотношение оставшихся и ушедших клиентов в выборке target_train: 79.6% / 20.4%
  - Соотношение оставшихся и ушедших клиентов в выборке target_valid: 79.6% / 20.4%
  - Соотношение оставшихся и ушедших клиентов в выборке target_test: 79.6% / 20.4%
- Обучили модель без учёта дисбаланса
  - Лучший результат Decision Tree (Дерево решений):
    - Глубина = 5
    - F1-мера = 0.5733788395904437
    - AUC-ROC = 0.8450470068195747
  - Лучший результат Random Forest (Случайный лес):
    - Количество деревьев = 20
    - Глубина = 9
    - F1-мера = 0.5983193277310924
    - AUC-ROC = 0.8755190122886461
  - Лучший результат Logistic Regression (Логистическая регрессия):
    - Алгоритм = lbfgs
    - F1-мера = 0.3306772908366534
    - AUC-ROC = 0.7893382907660986
  - Лучшая модель - Random Forest (Случайный лес)
  - F1-мера нашей лучшей модели (0.598) примерно равна поставленной задаче (0.59)

- **Борьба с дисбалансом**

  - Метод 1: Увеличение объектов редкого класса (техника upsampling)
    - Соотношение оставшихся и ушедших клиентов: 49.4% / 50.6%
    - Обучив три модели с применением техники "upsampling" мы определили лучшую модель - Random Forest (Случайный лес), с гиперпараметрами:
      - n_estimators = 30
      - max_depth = 7
      - F1-мера = 0.6233480176211453
      - AUC-ROC = 0.8771498611310323
    - F1-мера нашей лучшей модели (0.623) больше поставленной задачи (0.59)
  - Метод 2: Уменьшение объектов частого класса (техника downsampling)
    - Соотношение оставшихся и ушедших клиентов: 49.4% / 50.6%
    - Обучив три модели с применением техники "downsampling" мы определили лучшую модель - Random Forest (Случайный лес), с гиперпараметрами:
      - n_estimators = 30
      - max_depth = 7
      - F1-мера = 0.6121593291404612
      - AUC-ROC = 0.8773426570821311
    - F1-мера нашей лучшей модели (0.612) больше поставленной задачи (0.59)
  - Метод 3: Аргумент class_weight = 'balanced'
    - Обучив три модели с аргументом class_weight = 'balanced' мы определили лучшую модель - Random Forest (Случайный лес), с гиперпараметрами:
      - n_estimators = 50
      - max_depth = 9
      - F1-мера = 0.6463878326996199
      - AUC-ROC = 0.8749341047655059
    - F1-мера нашей лучшей модели (0.646) больше поставленной задачи (0.59)
  - По итогу исследования мы определили лучший метод балансировки и лучшую модель:
    - Аргумент class_weight = 'balanced'
    - Random Forest (Случайный лес), с гиперпараметрами:
      - n_estimators = 50
      - max_depth = 9
      - F1-мера = 0.6463878326996199
      - AUC-ROC = 0.8749341047655059

- **Тестирование модели**

  - Объединили обучающую и валидационную выборки для обучения итоговой модели на большем количестве данных
  - Обучили итоговую модель и проверили результат на тестовой выборке:
    - F1-мера = 0.5930680359435173
    - AUC-ROC = 0.8529266131554258
  - F1-мера нашей модели на тестовой выборке (0.593) чуть больше поставленной задачи (0.59)
  - Сравнили AUC-ROC нашей модели с AUC-ROC случайной модели:
    - AUC-ROC случайной модели = 0.5
    - AUC-ROC нашей модели = 0.85
    - Мы получили достаточно большое значение AUC-ROC на нашей итоговой модели
    - Итоговую модель можно считать адекватной