ad-task-001

17 июня 2024 г.

[1]:
```
'''
Задание 1
Будем работать с данными о клиентах банка, который интересуется, произойдет ли␣
 ↪просрочка платежа на 90 и более дней при выдаче кредита.
'''
import pandas as pd
```

[2]:
```
'''
Прочтите данные из файла data.csv
'''

filename = "..\\..\\data\\001\\data.csv"
df = pd.read_csv(filename, delimiter=",", decimal=".")
```

[3]:
```
'''
Выведите описание прочтенных данных
'''

df.describe()
```

[3]:

	Id	SeriousDlqin2yrs	RevolvingUtilizationOfUnsecuredLines	\
count	1350.000000	1350.000000	1350.000000	
mean	675.500000	0.060000	3.577895	
std	389.855743	0.237575	84.914699	
min	1.000000	0.000000	0.000000	
25%	338.250000	0.000000	0.031140	
50%	675.500000	0.000000	0.156891	
75%	1012.750000	0.000000	0.543145	
max	1350.000000	1.000000	2340.000000	

	age	NumberOfTime30-59DaysPastDueNotWorse	DebtRatio	\
count	1350.000000	1350.000000	1350.000000	
mean	52.048889	0.257778	356.123363	
std	15.009875	0.751718	1156.603074	
min	22.000000	0.000000	0.000000	
25%	40.000000	0.000000	0.175125	
50%	52.000000	0.000000	0.367049	
75%	63.000000	0.000000	0.807001	

```
max      97.000000                              10.000000  15466.000000

       MonthlyIncome  NumberOfOpenCreditLinesAndLoans  \
count     1094.000000                      1350.000000
mean      6438.473492                         8.434074
std       7849.754675                         5.129287
min          0.000000                         0.000000
25%       3300.000000                         5.000000
50%       5222.500000                         8.000000
75%       8055.250000                        11.000000
max     208333.000000                        31.000000

       NumberOfTimes90DaysLate  NumberRealEstateLoansOrLines  \
count              1350.000000                   1350.000000
mean                  0.080000                      0.986667
std                   0.376634                      1.008401
min                   0.000000                      0.000000
25%                   0.000000                      0.000000
50%                   0.000000                      1.000000
75%                   0.000000                      2.000000
max                   5.000000                      8.000000

       NumberOfTime60-89DaysPastDueNotWorse  NumberOfDependents
count                           1350.000000         1307.000000
mean                               0.062222            0.737567
std                                0.306555            1.086949
min                                0.000000            0.000000
25%                                0.000000            0.000000
50%                                0.000000            0.000000
75%                                0.000000            1.000000
max                                5.000000            8.000000
```

[4]:
```
'''
Отобразите несколько первых и несколько последних записей
'''
df.head(10)
```

[4]:
```
   Id  SeriousDlqin2yrs  RevolvingUtilizationOfUnsecuredLines  age  \
0   1                 1                              0.766127   45
1   2                 0                              0.957151   40
2   3                 0                              0.658180   38
3   4                 0                              0.233810   30
4   5                 0                              0.907239   49
5   6                 0                              0.213179   74
6   7                 0                              0.305682   57
7   8                 0                              0.754464   39
8   9                 0                              0.116951   27
```

```
9  10                  0                                0.189169   57

   NumberOfTime30-59DaysPastDueNotWorse    DebtRatio  MonthlyIncome  \
0                                     2     0.802982         9120.0
1                                     0     0.121876         2600.0
2                                     1     0.085113         3042.0
3                                     0     0.036050         3300.0
4                                     1     0.024926        63588.0
5                                     0     0.375607         3500.0
6                                     0  5710.000000            NaN
7                                     0     0.209940         3500.0
8                                     0    46.000000            NaN
9                                     0     0.606291        23684.0

   NumberOfOpenCreditLinesAndLoans  NumberOfTimes90DaysLate  \
0                               13                        0
1                                4                        0
2                                2                        1
3                                5                        0
4                                7                        0
5                                3                        0
6                                8                        0
7                                8                        0
8                                2                        0
9                                9                        0

   NumberRealEstateLoansOrLines  NumberOfTime60-89DaysPastDueNotWorse  \
0                             6                                     0
1                             0                                     0
2                             0                                     0
3                             0                                     0
4                             1                                     0
5                             1                                     0
6                             3                                     0
7                             0                                     0
8                             0                                     0
9                             4                                     0

   NumberOfDependents
0                 2.0
1                 1.0
2                 0.0
3                 0.0
4                 0.0
5                 1.0
6                 0.0
7                 0.0
```

```
8                  NaN
9                  2.0
```

[5]:
```
df.tail(10)
```

[5]:

	Id	SeriousDlqin2yrs	RevolvingUtilizationOfUnsecuredLines	age	\
1340	1341	0	0.002428	77	
1341	1342	0	0.173949	68	
1342	1343	0	0.047198	74	
1343	1344	0	0.202775	59	
1344	1345	0	0.087406	32	
1345	1346	0	0.000000	39	
1346	1347	0	0.045694	49	
1347	1348	0	0.022780	53	
1348	1349	0	0.036934	56	
1349	1350	0	0.000000	62	

	NumberOfTime30-59DaysPastDueNotWorse	DebtRatio	MonthlyIncome	\
1340	0	0.238471	3165.0	
1341	0	3015.000000	NaN	
1342	0	0.375672	9305.0	
1343	0	6994.000000	NaN	
1344	0	0.288978	1750.0	
1345	0	0.055916	4166.0	
1346	0	0.300175	4000.0	
1347	0	0.323068	10000.0	
1348	0	0.287935	8362.0	
1349	0	1463.000000	NaN	

	NumberOfOpenCreditLinesAndLoans	NumberOfTimes90DaysLate	\
1340	13	0	
1341	7	0	
1342	8	0	
1343	23	0	
1344	8	0	
1345	5	0	
1346	14	0	
1347	14	0	
1348	8	0	
1349	5	0	

	NumberRealEstateLoansOrLines	NumberOfTime60-89DaysPastDueNotWorse	\
1340	1	0	
1341	2	0	
1342	5	0	
1343	3	0	
1344	0	0	

```
1345                           0                                0
1346                           1                                0
1347                           2                                0
1348                           1                                0
1349                           1                                0

      NumberOfDependents
1340                 0.0
1341                 0.0
1342                 8.0
1343                 2.0
1344                 2.0
1345                 0.0
1346                 1.0
1347                 1.0
1348                 2.0
1349                 0.0
```

[6]:
```
'''
Заметьте, что столбец DebtRatio содержит неправдоподобные данные.
Только значения, соответствующие известному месячному доходу, являются␣
 ↪отношениями.
Остальные - абсолютные значения месячных выплат процентов.
Исправьте данные, сделав все значения столбца DebtRatio абсолютными (умножьте их␣
 ↪на MonthlyIncome).
Чтобы ваша программа быстро работала на полных данных, постарайтесь не␣
 ↪использовать цикл.
'''

df["DebtRatio"] = df["DebtRatio"].mul(df["MonthlyIncome"], fill_value = 1)
```

[7]:
```
'''
Поменяйте имя столбца на Debt.
'''
df.rename(columns= {"DebtRatio" : "Debt"}, inplace=True)

df.loc[:10, ["Debt", "MonthlyIncome"]]
```

[7]:
```
             Debt  MonthlyIncome
0    7323.197016         9120.0
1     316.878123         2600.0
2     258.914887         3042.0
3     118.963951         3300.0
4    1584.975094        63588.0
5    1314.624392         3500.0
6    5710.000000            NaN
7     734.790059         3500.0
```

```
8      46.000000            NaN
9   14359.393699        23684.0
10    773.690525         2500.0
```

[8]:
```
'''
Вычислите средний ежемесячный доход.
'''

avg_monthly_income = df["MonthlyIncome"].mean()
avg_monthly_income
```

[8]:
```
6438.4734917733085
```

[9]:
```
'''
Присвойте всем клиентам с неизвестным доходом полученное число.
'''

df["MonthlyIncome"] = df["MonthlyIncome"].fillna(avg_monthly_income)

df.loc[:10, ["Debt", "MonthlyIncome"]]
```

[9]:
```
             Debt  MonthlyIncome
0     7323.197016    9120.000000
1      316.878123    2600.000000
2      258.914887    3042.000000
3      118.963951    3300.000000
4     1584.975094   63588.000000
5     1314.624392    3500.000000
6     5710.000000    6438.473492
7      734.790059    3500.000000
8       46.000000    6438.473492
9    14359.393699   23684.000000
10     773.690525    2500.000000
```

[10]:
```
'''
Используя метод groupby, оцените вероятности невозврата кредита␣
↪(SeriousDlqin2yrs=1) для различных значений количества иждивенцев␣
↪(NumberOfDependents).
'''

# SeriousDlqin2yrs_Count - общее количество записей в группе, исключая NaN
# SeriousDlqin2yrs_Sum - т.к. значения столбца - 0 и 1, то в этой колонке␣
↪отобразится количество записей в группе, содержащих 1, т.е. с просрочкой␣
↪платежа
tuples = [("SeriousDlqin2yrs_Count", "count"), ("SeriousDlqin2yrs_Sum", "sum")]

grouped1 = df.groupby("NumberOfDependents")
```

```
data = grouped1["SeriousDlqin2yrs"].agg(tuples)
df1 = pd.DataFrame(data = data)
# Вероятность невозврата кредита находим как отношение количества просрочек к␣
 ↪общему количеству записей в группе
df1["ProbabilityOfDefault"] = df1["SeriousDlqin2yrs_Sum"] /␣
 ↪df1["SeriousDlqin2yrs_Count"]

# df1 - вероятность невозврата кредита в зависимости от количества иждивенцев␣
 ↪(NumberOfDependents)
df1.loc[:, "ProbabilityOfDefault"]
```

[10]:
```
NumberOfDependents
0.0    0.041397
1.0    0.089844
2.0    0.110465
3.0    0.057143
4.0    0.033333
5.0    0.000000
6.0    0.000000
8.0    0.000000
Name: ProbabilityOfDefault, dtype: float64
```

[11]:
```
# df[df["NumberOfDependents"] == 5].loc[:, ["SeriousDlqin2yrs",␣
 ↪"NumberOfDependents"]]
```

[12]:
```
'''
Проделайте аналогичную процедуру для различных значений столбца␣
 ↪NumberRealEstateLoansOrLines
'''

grouped2 = df.groupby("NumberRealEstateLoansOrLines")
data = grouped2["SeriousDlqin2yrs"].agg(tuples)
df2 = pd.DataFrame(data = data)

# Вероятность невозврата кредита находим как отношение количества просрочек к␣
 ↪общему количеству записей в группе
df2["ProbabilityOfDefault"] = df2["SeriousDlqin2yrs_Sum"] /␣
 ↪df2["SeriousDlqin2yrs_Count"]

# df2 - вероятность невозврата кредита в зависимости от количества ипотек␣
 ↪(NumberRealEstateLoansOrLines)

df2.loc[:, "ProbabilityOfDefault"]
```

[12]:
```
NumberRealEstateLoansOrLines
0    0.056863
1    0.048729
```

```
2    0.063158
3    0.145455
4    0.105263
5    0.000000
6    1.000000
8    0.000000
Name: ProbabilityOfDefault, dtype: float64
```

[13]:
```
#df[df["NumberRealEstateLoansOrLines"] == 5].loc[:, ["SeriousDlqin2yrs",
↪"NumberRealEstateLoansOrLines"]]
```

[13]:
```

```