Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс   
«Технологии машинного обучения»  
 Отчет по лабораторной работе №2

|  |  |
| --- | --- |
| Выполнил: | Проверил: |
| студент группы ИУ5-63Б | преподаватель каф. ИУ5 |
| Тарновский Д.Р. | Гапанюк Ю.Е. |

Москва, 2022 г.

In [2]:

**import** numpy **as** np

**import** pandas **as** pd

**import** matplotlib.pyplot **as** plt

**%matplotlib** inline

In [3]:

data **=** pd**.**read\_csv('credit\_train.csv', encoding**=**'cp1251', sep**=**';')

In [5]:

data**.**head() data**.**shape[0]

Out[5]:

In [4]:

170746

# Обработка пропусков в данных

<class 'pandas.core.frame.DataFrame'>

*#проверяем типы данных и заполненность столбцов*

data**.**info()

RangeIndex: 170746 entries, 0 to 170745 Data columns (total 15 columns):

# Column Non-Null Count Dtype

In [5]:

* 1. client\_id 170746 non-null int64
  2. gender 170746 non-null object
  3. age 170743 non-null float64
  4. marital\_status 170743 non-null object
  5. job\_position 170746 non-null object
  6. credit\_sum 170744 non-null object
  7. credit\_month 170746 non-null int64
  8. tariff\_id 170746 non-null float64
  9. score\_shk 170739 non-null object
  10. education 170741 non-null object
  11. living\_region 170554 non-null object
  12. monthly\_income 170741 non-null float64
  13. credit\_count 161516 non-null float64
  14. overdue\_credit\_count 161516 non-null float64
  15. open\_account\_flg 170746 non-null int64 dtypes: float64(5), int64(3), object(7)

memory usage: 19.5+ MB

data**.**drop(['client\_id', 'living\_region'], axis**=**1, inplace**=True**)

In [6]:

*# анализируем столбец marital\_status, смотрим, какое значение в нем явля*

data['marital\_status']**.**describe()

Out[6]:

In [7]:

count 170743

unique 5

top MAR

freq 93954

Name: marital\_status, dtype: object

*# анализируем столбец education, смотрим, какое в нем самое частое значе*

data['education']**.**describe()

Out[7]:

In [8]:

count 170741

unique 5

top SCH

freq 87537

Name: education, dtype: object

*# дозаполняем нечисловые столбцы с пропусками самыми часто встречающимис*

data['marital\_status']**.**fillna('MAR', inplace**=True**) data['education']**.**fillna('SCH', inplace**=True**)

In [9]:

*# дозаполняем числовые столбцы с пропусками медианными значениями*

data['age']**.**fillna(data['age']**.**median(), inplace**=True**)

data['credit\_count']**.**fillna(data['credit\_count']**.**median(), inplace**=True**) data['overdue\_credit\_count']**.**fillna(data['overdue\_credit\_count']**.**median(

In [10]:

*#меняем в столбцах 'credit\_sum', 'score\_shk' запятые на точки и преобр*

**for** i **in** ['credit\_sum', 'score\_shk']:

data[i] **=** data[i]**.**str**.**replace(',', '.')**.**astype('float')

In [11]:

*# дозаполняем ставшие теперь числовыми столбцы 'credit\_sum', 'score\_shk'*

data['score\_shk']**.**fillna(data['score\_shk']**.**median(), inplace**=True**)

data['monthly\_income']**.**fillna(data['monthly\_income']**.**median(), inplace**=T** data['credit\_sum']**.**fillna(data['credit\_sum']**.**median(), inplace**=True**)

In [12]:

*# смотрим, что получилось*

data**.**info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 170746 entries, 0 to 170745 Data columns (total 13 columns):

# Column Non-Null Count Dtype

* + 1. gender 170746 non-null object
    2. age 170746 non-null float64
    3. marital\_status 170746 non-null object
    4. job\_position 170746 non-null object
    5. credit\_sum 170746 non-null float64
    6. credit\_month 170746 non-null int64
    7. tariff\_id 170746 non-null float64
    8. score\_shk 170746 non-null float64
    9. education 170746 non-null object
    10. monthly\_income 170746 non-null float64
    11. credit\_count 170746 non-null float64
    12. overdue\_credit\_count 170746 non-null float64
    13. open\_account\_flg 170746 non-null int64 dtypes: float64(7), int64(2), object(4)

memory usage: 16.9+ MB

In [13]:

category\_cols **=** ['gender', 'job\_position', 'education', 'marital\_status'

# Кодирование категориальных признаков

In [14]:

print("Количество уникальных значений\n")

**for** col **in** category\_cols:

print(f'{col}: {data[col]**.**unique()**.**size}')

Количество уникальных значений

In [15]:

gender: 2

job\_position: 18

education: 5

marital\_status: 5

*# кодируем нечисловые столбцы методом дамми-кодирования*

data **=** pd**.**concat([data,

pd**.**get\_dummies(data['gender'], prefix**=**"gender"), pd**.**get\_dummies(data['job\_position'], prefix**=**"job\_p

pd**.**get\_dummies(data['education'], prefix**=**"educatio pd**.**get\_dummies(data['marital\_status'], prefix**=**"mar

axis**=**1)

In [16]:

*#удаляем старые нечисловые столбцы, вместо них уже появились новые число*

data**.**drop(['gender','job\_position','education','marital\_status'], axis**=**1

In [17]:

data**.**head()

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Out[17]: | **age** | **credit\_sum** | **credit\_month** | **tariff\_id** | **score\_shk** | **monthly\_income** | **credit\_count over** |
|  | **0** 34.0 | 59998.00 | 10 | 1.6 | 0.461599 | 30000.0 | 1.0 |
|  | **1** 34.0 | 10889.00 | 6 | 1.1 | 0.461599 | 35000.0 | 2.0 |
|  | **2** 32.0 | 10728.00 | 12 | 1.1 | 0.461599 | 35000.0 | 5.0 |
|  | **3** 27.0 | 12009.09 | 12 | 1.1 | 0.461599 | 35000.0 | 2.0 |
|  | **4** 45.0 | 21229.00 | 10 | 1.1 | 0.421385 | 35000.0 | 1.0 |

5 rows × 39 columns

In [18]:

**from** sklearn.preprocessing **import** MinMaxScaler, StandardScaler, Normaliz

# Масштабирование данных

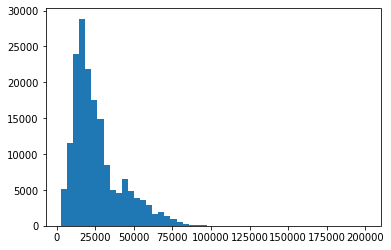
In [19]:

sc1 **=** MinMaxScaler()

sc1\_data **=** sc1**.**fit\_transform(data[['credit\_sum']])

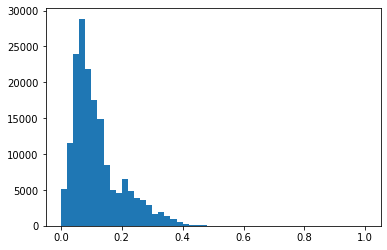
In [28]:

plt**.**hist(data['credit\_sum'], 50) plt**.**show()



In [22]:

plt**.**hist(sc1\_data, 50) plt**.**show()



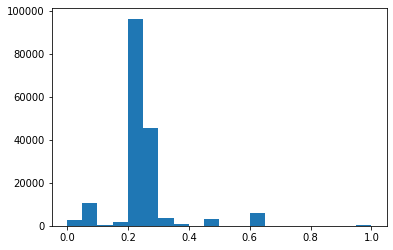
In [23]:

scCredMon **=** MinMaxScaler()

scCredMon **=** scCredMon**.**fit\_transform(data[['credit\_month']])

In [29]:

plt**.**hist(scCredMon, 20) plt**.**show()



In [30]:

scMonInc **=** MinMaxScaler()**.**fit\_transform(data['monthly\_income']**.**values**.**re

In [33]:

plt**.**hist(scMonInc, 30) plt**.**show()

