РК1 Якшин Егор, ИУ5-65Б

# Вариант: 23; Задача: 3; Датасет: 7

Задача 3

Для заданного набора данных произведите масштабирование данных (для одного признака) и преобразование категориальных признаков в количественные двумя способами (label encoding, one hot encoding) для одного признака. Какие методы Вы

использовали для решения задачи и почему?

Dataset: <https://www.kaggle.com/mohansacharya/graduate-admissions>

## Импорт библиотек:

In [1]:

**import** numpy **as** np **import** pandas **as** pd **import** seaborn **as** sns

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

**from** sklearn.impute **import** SimpleImputer **from** sklearn.impute **import** MissingIndicator **import** warnings warnings**.**filterwarnings('ignore') sns**.**set(style**=**"ticks")

**%matplotlib** inline

In [2]:

data **=** pd**.**read\_csv('restaurant-scores-lives-standard.csv')

In [3]:

data**.**head()

Out[3]:

**business\_id business\_name business\_address business\_city business\_state business\_postal\_code business\_latitude business\_longitude busi**

**0** 101192 Cochinita #2 2 Marina Blvd San Francisco CA NaN NaN NaN Fort Mason

**1** 97975 BREADBELLY 1408 Clement St San Francisco

CA

94118

NaN

NaN

**2** 92982 Great Gold Restaurant

3161 24th St. San Francisco CA 94110 NaN NaN

**3** 101389 HOMAGE 214 CALIFORNIA San Francisco

ST

CA

94111

NaN

NaN

**4** 85986 Pronto Pizza 798 Eddy St San Francisco CA 94109 NaN NaN

5 rows × 23 columns

In [4]:

data**.**dtypes

Out[4]: business\_id int64

business\_name object

business\_address object

business\_city object

business\_state object

business\_postal\_code object

business\_latitude float64

business\_longitude float64

business\_location object

business\_phone\_number float64

inspection\_id object

inspection\_date object

inspection\_score float64

inspection\_type object

violation\_id object

violation\_description object

risk\_category object

Neighborhoods (old) float64

Police Districts float64

Supervisor Districts float64 Fire Prevention Districts float64 Zip Codes float64

Analysis Neighborhoods float64 dtype: object

In [5]:

data**.**isnull()**.**sum()

*# проверим есть ли пропущенные значения*

|  |  |
| --- | --- |
| Out[5]: business\_id | 0 |
| business\_name | 0 |
| business\_address | 0 |
| business\_city | 0 |
| business\_state | 0 |
| business\_postal\_code | 1018 |
| business\_latitude | 19556 |
| business\_longitude | 19556 |
| business\_location | 19556 |
| business\_phone\_number | 36938 |
| inspection\_id | 0 |
| inspection\_date | 0 |
| inspection\_score | 13610 |
| inspection\_type | 0 |
| violation\_id | 12870 |
| violation\_description | 12870 |
| risk\_category | 12870 |
| Neighborhoods (old) | 19594 |
| Police Districts | 19594 |
| Supervisor Districts | 19594 |
| Fire Prevention Districts | 19646 |
| Zip Codes | 19576 |
| Analysis Neighborhoods | 19594 |
| dtype: int64 |  |

In [6]:

data**.**info()

<class 'pandas.core.frame.DataFrame'> RangeIndex: 53973 entries, 0 to 53972 Data columns (total 23 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

1. business\_id 53973 non-null int64
2. business\_name 53973 non-null object
3. business\_address 53973 non-null object
4. business\_city 53973 non-null object
5. business\_state 53973 non-null object
6. business\_postal\_code 52955 non-null object
7. business\_latitude 34417 non-null float64
8. business\_longitude 34417 non-null float64
9. business\_location 34417 non-null object
10. business\_phone\_number 17035 non-null float64
11. inspection\_id 53973 non-null object
12. inspection\_date 53973 non-null object
13. inspection\_score 40363 non-null float64
14. inspection\_type 53973 non-null object
15. violation\_id 41103 non-null object
16. violation\_description 41103 non-null object
17. risk\_category 41103 non-null object
18. Neighborhoods (old) 34379 non-null float64
19. Police Districts 34379 non-null float64
20. Supervisor Districts 34379 non-null float64
21. Fire Prevention Districts 34327 non-null float64
22. Zip Codes 34397 non-null float64
23. Analysis Neighborhoods 34379 non-null float64 dtypes: float64(10), int64(1), object(12)

memory usage: 9.5+ MB

In [7]:

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

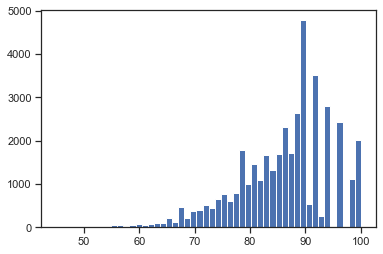
In [8]:

sc1 **=** MinMaxScaler()

sc1\_data **=** sc1**.**fit\_transform(data[['inspection\_score']])

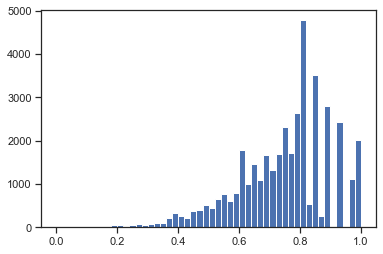
In [9]:

plt**.**hist(data['inspection\_score'], 50) plt**.**show()



In [10]:

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



# Масштабирование данных на основе Z-оценки - StandardScaler¶

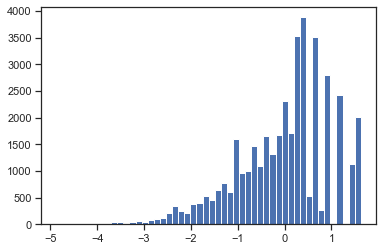
In [11]:

sc2 **=** StandardScaler()

sc2\_data **=** sc2**.**fit\_transform(data[['inspection\_score']])

In [12]:

plt**.**hist(sc2\_data, 50) plt**.**show()



In [13]:

cat\_temp\_data **=** data[['business\_name']] cat\_temp\_data**.**head()

Out[13]:

**business\_name**

**0** Cochinita #2

**1** BREADBELLY

**2** Great Gold Restaurant

**3** HOMAGE

**4** Pronto Pizza

In [14]:

*# Импьютация наиболее частыми значениями*

imp2 **=** SimpleImputer(missing\_values**=**np**.**nan, strategy**=**'most\_frequent') data\_imp2 **=** imp2**.**fit\_transform(cat\_temp\_data)

data\_imp2

Out[14]: array([['Cochinita #2'],

['BREADBELLY'],

['Great Gold Restaurant'],

...,

['Philz Coffee'],

['El Gran Taco Loco'],

['Blue Bottle Coffee']], dtype=object)

In [15]:

cat\_enc **=** pd**.**DataFrame({'business\_name':data\_imp2**.**T[0]}) cat\_enc

Out[15]:

**business\_name**

**0** Cochinita #2

**1** BREADBELLY

**2** Great Gold Restaurant

**3** HOMAGE

**4** Pronto Pizza

**...** ...

**53968** Blue Bottle Coffee

**53969** POKE KANA

**53970** Philz Coffee

**53971** El Gran Taco Loco

**53972** Blue Bottle Coffee

53973 rows × 1 columns

In [16]:

**from** sklearn.preprocessing **import** LabelEncoder, OneHotEncoder

# Кодирование категорий целочисленными значениями - label encoding

In [17]:

le **=** LabelEncoder()

cat\_enc\_le **=** le**.**fit\_transform(cat\_enc['business\_name'])

In [18]:

cat\_enc['business\_name']**.**unique()

Out[18]: array(['Cochinita #2', 'BREADBELLY', 'Great Gold Restaurant', ..., 'LITTLE PARIS', 'Tosca Cafe', 'LITTLE SWEET'], dtype=object)

In [19]:

np**.**unique(cat\_enc\_le)

Out[19]: array([ 0, 1, 2, ..., 5569, 5570, 5571])

In [20]:

le**.**inverse\_transform([0, 1, 2, 3])

Out[20]: array(['#1 VERJUS CAVE, #2 VERJUS', '111 Minna Gallery',

'12 Tribes Kosher Foods', '1428 Haight'], dtype=object)

# Кодирование категорий наборами бинарных значений - one-hot encoding

In [21]:

ohe **=** OneHotEncoder()

cat\_enc\_ohe **=** ohe**.**fit\_transform(cat\_enc[['business\_name']])

In [22]:

cat\_enc**.**shape

Out[22]: (53973, 1)

In [23]:

cat\_enc\_ohe**.**shape

Out[23]: (53973, 5572)

In [24]:

cat\_enc\_ohe

Out[24]: <53973x5572 sparse matrix of type '<class 'numpy.float64'>'

with 53973 stored elements in Compressed Sparse Row format>

In [25]:

cat\_enc\_ohe**.**todense()[0:10]

Out[25]: matrix([[0., 0., 0., ..., 0., 0., 0.],

[0., 0., 0., ..., 0., 0., 0.],

[0., 0., 0., ..., 0., 0., 0.],

...,

[0., 0., 0., ..., 0., 0., 0.],

[0., 0., 0., ..., 0., 0., 0.],

[0., 0., 0., ..., 0., 0., 0.]])

In [26]:

cat\_enc**.**head(10)

|  |  |  |
| --- | --- | --- |
| Out[26]: |  | **business\_name** |
|  | **0** | Cochinita #2 |
|  | **1** | BREADBELLY |
|  | **2** | Great Gold Restaurant |
|  | **3** | HOMAGE |
|  | **4** | Pronto Pizza |
|  | **5** | Brickhouse |

**6** LAI HONG RESTAURANT

**7** Fools Errand

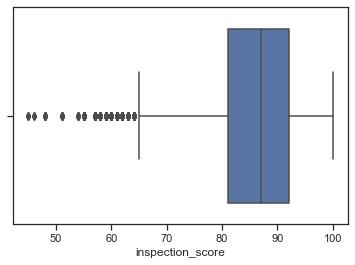
**8** MoBowL

**9** CurveBall

## Реализовываем "ящик с усами"

In [27]:

sns**.**boxplot(data['inspection\_score'])

Out[27]: <AxesSubplot:xlabel='inspection\_score'>