Московский государственный технический университет им. Н.Э. Баумана
Кафедра «Системы обработки информации и управления»

Лабораторная работа №3
по дисциплине
«Методы машинного обучения»
на тему

«Обработка пропусков в данных, кодирование категориальных признаков, масштабирование данных»

Выполнил:
студент группы ИУ5-21М
Якубов А. Р.

____

Москва — 2019 г.

1. Цель лабораторной работы

Изучить способы предварительной обработки данных для дальнейшего формирования моделей. [1]

2. Задание

Выбрать набор данных (датасет), содержащий категориальные признаки и пропуски в данных. Для выполнения следующих пунктов можно использовать несколько различных наборов данных (один для обработки пропусков, другой для категориальных признаков и т.д.)

Для выбранного датасета (датасетов) на основе материалов лекции решить следующие задачи:

- обработку пропусков в данных;
- кодирование категориальных признаков;
- масштабирование данных.

3. Загрузка и первичный анализ данных

```
In [1]: import numpy as np
        import pandas as pd
        import seaborn as sns
        import matplotlib.pyplot as plt
        %matplotlib inline
        sns.set(style='ticks')
```

```
In [2]: data = pd.read_csv('data/restaurant-scores-lives-standard.csv', sep=",
```

```
In [14]: original_size = data.shape
         print("Исходный размер:")
         print("\t- количество строк: %s" % original_size[0])
         print("\t- количество столбцов: %s" % original_size[1])
         total_count = original_size[0]
```

```
Исходный размер:
        - количество строк: 53686
        - количество столбцов: 17
```

```
In [11]: # Первые 5 строк набора
         data.head()
```

```
Out[11]:    business_id                         business_name   business_address
         0        70961  Our Lady of the Visitacion School  785 Sunnydale Ave
         1        10030         Marshall Elementary School       1575 15th St
         2        69006        Chipotle Mexican Grill #1566   50 California St
         3         5868        LONGFELLOW ELEMENTARY SCHOOL        755 MORSE St
         4         5864     VISITACION VALLEY MIDDLE SCHOOL     450 Raymond Ave

            business_city business_state business_postal_code  business_latitu
```

```
0  San Francisco         CA                 94134            N
1  San Francisco         CA                 94103         37.7668
2  San Francisco         CA                 94111            N
3  San Francisco         CA                 94112         37.7104
4  San Francisco         CA                 94134         37.7144

   business_longitude                                   business_locat
0                 NaN
1         -122.419014  {'latitude': '37.766864', 'human_address': '{"
2                 NaN
3         -122.447713  {'latitude': '37.710459', 'human_address': '{"
4         -122.411433  {'latitude': '37.714428', 'human_address': '{"

   business_phone_number   inspection_id      inspection_date  \
0                    NaN  70961_20160321  2016-03-21T00:00:00
1           1.415525e+10  10030_20160321  2016-03-21T00:00:00
2                    NaN  69006_20160321  2016-03-21T00:00:00
3           1.415546e+10   5868_20160321  2016-03-21T00:00:00
4                    NaN   5864_20160321  2016-03-21T00:00:00

   inspection_score        inspection_type           violation_id  \
0             100.0  Routine - Unscheduled                    NaN
1              96.0  Routine - Unscheduled  10030_20160321_103120
2              96.0  Routine - Unscheduled  69006_20160321_103148
3              87.0  Routine - Unscheduled   5868_20160321_103154
4              94.0  Routine - Unscheduled   5864_20160321_103157

                             violation_description  risk_category
0                                              NaN            NaN
1            Moderate risk food holding temperature  Moderate Risk
2         No thermometers or uncalibrated thermometers       Low Risk
3         Unclean or degraded floors walls or ceilings       Low Risk
4  Food safety certificate or food handler card n…       Low Risk
```

In [12]:
```python
# Типы колонок
data.dtypes
```

Out[12]:
```
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
dtype: object
```

```
In [13]: # Количество пропущенных значений в каждой колонке
         data.isnull().sum()
```

```
Out[13]: business_id                   0
         business_name                 0
         business_address              0
         business_city                 0
         business_state                0
         business_postal_code       1241
         business_latitude         24005
         business_longitude        24005
         business_location         24005
         business_phone_number     36989
         inspection_id                 0
         inspection_date               0
         inspection_score          13947
         inspection_type               0
         violation_id              12946
         violation_description     12946
         risk_category             12946
         dtype: int64
```

4. Обработка пропусков в данных

Рассматриваются колонки категориальных и количественных признаков, содержащих пропуски в данных.

Требуется выбрать одну колонку категориального признака и одну колонку количественного признака и произвести обработку пропусков в каждой из них.

```
In [16]: from sklearn.impute import SimpleImputer
         from sklearn.impute import MissingIndicator
```

4.1. Для категориального признака

```
In [15]: # Выберем категориальные колонки с пропущенными значениями
         # Цикл по колонкам датасета
         cat_cols = []
         for col in data.columns:
             # Количество пустых значений
             temp_null_count = data[data[col].isnull()].shape[0]
             dt = str(data[col].dtype)
             if temp_null_count>0 and (dt=='object'):
                 cat_cols.append(col)
                 temp_perc = round((temp_null_count / total_count) * 100.0, 2)
                 print('Колонка {}. Тип данных {}. Количество пустых значений
```

```
Колонка business_postal_code. Тип данных object. Количество пустых значений 12
Колонка business_location. Тип данных object. Количество пустых значений 24005
Колонка violation_id. Тип данных object. Количество пустых значений 12946, 24.
Колонка violation_description. Тип данных object. Количество пустых значений 1
Колонка risk_category. Тип данных object. Количество пустых значений 12946, 24
```

```
In [34]: # Фильтр по колонкам с пропущенными значениями
         data_cat = data[cat_cols]
         data_cat
```

Out[34]:

	business_postal_code	business_
0	94134	
1	94103	{'latitude': '37.766864', 'human_address'
2	94111	
3	94112	{'latitude': '37.710459', 'human_address'
4	94134	{'latitude': '37.714428', 'human_address'
5	94103	{'latitude': '37.766618', 'human_address'
6	94109	{'latitude': '37.794298', 'human_address'
7	94109	{'latitude': '37.792854', 'human_address'
8	94121	{'latitude': '37.772323', 'human_address'
9	94134	{'latitude': '37.714428', 'human_address'
10	94109	{'latitude': '37.791683', 'human_address'
11	94121	{'latitude': '37.782107', 'human_address'
12	94134	
13	94134	{'latitude': '37.714428', 'human_address'
14	94112	{'latitude': '37.710459', 'human_address'
15	94112	{'latitude': '37.709896', 'human_address'
16	94121	{'latitude': '37.782107', 'human_address'
17	94111	
18	94103	{'latitude': '37.766618', 'human_address'
19	94112	{'latitude': '37.709896', 'human_address'
20	94121	{'latitude': '37.782107', 'human_address'
21	94103	{'latitude': '37.766618', 'human_address'
22	94121	{'latitude': '37.782107', 'human_address'
23	94109	{'latitude': '37.790253', 'human_address'
24	94134	{'latitude': '37.729016', 'human_address'
25	94134	
26	94112	{'latitude': '37.710459', 'human_address'
27	94112	{'latitude': '37.709896', 'human_address'
28	94110	{'latitude': '37.754397', 'human_address'
29	94102	{'latitude': '37.788673', 'human_address'
…	…	
53656	94112	
53657	94102	{'latitude': '37.777017', 'human_address'
53658	94110	
53659	94122	
53660	94103	{'latitude': '37.774722', 'human_address'
53661	94111	
53662	94117	
53663	94110	{'latitude': '37.743206', 'human_address'

```
53664                   94110  
53665                   94124  
53666                   94110  
53667                   94110  
53668                   94114  {'latitude': '37.767194', 'human_address'
53669                   94110  
53670                   94103  
53671                   94134  
53672                   94114  
53673                   94109  
53674                   94133  {'latitude': '37.797868', 'human_address'
53675                   94134  {'latitude': '37.715126', 'human_address'
53676                   94103  {'latitude': '37.778547', 'human_address'
53677                   94111  {'latitude': '37.79373', 'human_address':
53678                   94111  
53679                   94103  
53680                   94124  
53681                   94116  {'latitude': '37.742048', 'human_address'
53682                   94102  
53683                   94110  
53684                   94110  
53685                   94111  

                 violation_id  \
0                         NaN   
1       10030_20160321_103120   
2       69006_20160321_103148   
3        5868_20160321_103154   
4        5864_20160321_103157   
5        5998_20160321_103109   
6                         NaN   
7                         NaN   
8                         NaN   
9        5864_20160321_103144   
10                        NaN   
11        551_20160321_103142   
12       5869_20160321_103119   
13       5864_20160321_103154   
14       5868_20160321_103119   
15       5955_20160321_103139   
16        551_20160321_103103   
17      69006_20160321_103141   
18       5998_20160321_103161   
19       5955_20160321_103141   
20        551_20160321_103157   
21       5998_20160321_103133   
22        551_20160321_103124   
23                        NaN   
24       5827_20160321_103120   
25       5869_20160321_103116   
```

```
26      5868_20160321_103109
27      5955_20160321_103149
28      5999_20160322_103124
29      2926_20160322_103154
…                      …
53656  87791_20170221_103154
53657  65066_20160810_103124
53658  92662_20170921_103103
53659  77564_20161107_103119
53660   7407_20170728_103145
53661  90222_20190307_103131
53662  91984_20180206_103125
53663   3870_20190318_103144
53664  89453_20161019_103142
53665  81758_20171103_103124
53666                    NaN
53667  76441_20160517_103147
53668   4479_20180201_103131
53669  94231_20171214_103120
53670                    NaN
53671  76294_20161206_103138
53672  70184_20180313_103154
53673  86545_20160520_103138
53674   2945_20161012_103119
53675    194_20190319_103119
53676  18800_20171213_103120
53677  68826_20170222_103149
53678  86933_20160411_103131
53679  86284_20180820_103103
53680  91245_20170607_103149
53681    985_20181204_103154
53682                    NaN
53683  92662_20170921_103154
53684  97277_20180816_103154
53685  77955_20160819_103154

                                    violation_description  risk_catego
0                                                     NaN            N
1                  Moderate risk food holding temperature  Moderate Ri
2            No thermometers or uncalibrated thermometers       Low Ri
3            Unclean or degraded floors walls or ceilings       Low Ri
4       Food safety certificate or food handler card n…      Low Risk
5             Unclean or unsanitary food contact surfaces      High Ri
6                                                     NaN            N
7                                                     NaN            N
8                                                     NaN            N
9        Unapproved or unmaintained equipment or utensils       Low Ri
10                                                    NaN            N
11                       Unclean nonfood contact surfaces       Low Ri
12      Inadequate and inaccessible handwashing facili…  Moderate Risk
```

```
13           Unclean or degraded floors walls or ceilings       Low Ri
14      Inadequate and inaccessible handwashing facili…  Moderate Risk
15                                Improper food storage       Low Ri
16                   High risk food holding temperature      High Ri
17       Improper food labeling or menu misrepresentation       Low Ri
18                          Low risk vermin infestation       Low Ri
19       Improper food labeling or menu misrepresentation       Low Ri
20      Food safety certificate or food handler card n…       Low Risk
21                Foods not protected from contamination  Moderate Ri
22      Inadequately cleaned or sanitized food contact…  Moderate Risk
23                                                  NaN            N
24               Moderate risk food holding temperature  Moderate Ri
25      Inadequate food safety knowledge or lack of ce…  Moderate Risk
26            Unclean or unsanitary food contact surfaces      High Ri
27      Wiping cloths not clean or properly stored or …       Low Risk
28      Inadequately cleaned or sanitized food contact…  Moderate Risk
29           Unclean or degraded floors walls or ceilings       Low Ri
…                                                     …            …
53656        Unclean or degraded floors walls or ceilings       Low Ri
53657  Inadequately cleaned or sanitized food contact…  Moderate Risk
53658                High risk food holding temperature      High Ri
53659  Inadequate and inaccessible handwashing facili…  Moderate Risk
53660   Improper storage of equipment utensils or linens       Low Ri
53661                  Moderate risk vermin infestation  Moderate Ri
53662     Noncompliance with shell fish tags or display  Moderate Ri
53663   Unapproved or unmaintained equipment or utensils       Low Ri
53664                   Unclean nonfood contact surfaces       Low Ri
53665  Inadequately cleaned or sanitized food contact…  Moderate Risk
53666                                               NaN            N
53667                 Inadequate ventilation or lighting       Low Ri
53668                  Moderate risk vermin infestation  Moderate Ri
53669            Moderate risk food holding temperature  Moderate Ri
53670                                               NaN            N
53671  Improper storage use or identification of toxi…       Low Risk
53672        Unclean or degraded floors walls or ceilings       Low Ri
53673  Improper storage use or identification of toxi…       Low Risk
53674  Inadequate and inaccessible handwashing facili…  Moderate Risk
53675  Inadequate and inaccessible handwashing facili…  Moderate Risk
53676            Moderate risk food holding temperature  Moderate Ri
53677  Wiping cloths not clean or properly stored or …       Low Risk
53678                  Moderate risk vermin infestation  Moderate Ri
53679                High risk food holding temperature      High Ri
53680  Wiping cloths not clean or properly stored or …       Low Risk
53681        Unclean or degraded floors walls or ceilings       Low Ri
53682                                               NaN            N
53683        Unclean or degraded floors walls or ceilings       Low Ri
53684        Unclean or degraded floors walls or ceilings       Low Ri
53685        Unclean or degraded floors walls or ceilings       Low Ri

[53686 rows x 5 columns]
```

В качестве рассматриваемого категориального признака, имеющего пропуски, выбран столбец “business_postal_code”.
Обработка пропусков производится по стратегии импьютации постоянным значением.

```
In [31]: cat_temp_data = data[['business_postal_code']]
         cat_temp_data

Out[31]:        business_postal_code
         0                     94134
         1                     94103
         2                     94111
         3                     94112
         4                     94134
         5                     94103
         6                     94109
         7                     94109
         8                     94121
         9                     94134
         10                    94109
         11                    94121
         12                    94134
         13                    94134
         14                    94112
         15                    94112
         16                    94121
         17                    94111
         18                    94103
         19                    94112
         20                    94121
         21                    94103
         22                    94121
         23                    94109
         24                    94134
         25                    94134
         26                    94112
         27                    94112
         28                    94110
         29                    94102
         …                     …
         53656                 94112
         53657                 94102
         53658                 94110
         53659                 94122
         53660                 94103
         53661                 94111
         53662                 94117
         53663                 94110
         53664                 94110
         53665                 94124
         53666                 94110
         53667                 94110
```

```
53668              94114
53669              94110
53670              94103
53671              94134
53672              94114
53673              94109
53674              94133
53675              94134
53676              94103
53677              94111
53678              94111
53679              94103
53680              94124
53681              94116
53682              94102
53683              94110
53684              94110
53685              94111

[53686 rows x 1 columns]
```

```
In [26]: # Количество пропущенных значений в колонке
         cat_temp_data[cat_temp_data['business_postal_code'].isnull()].shape[0
```

```
Out[26]: 1241
```

```
In [19]: cat_temp_data['business_postal_code'].unique()
```

```
Out[19]: array(['94134', '94103', '94111', '94112', '94109', '94121', '94110',
                '94102', '94116', '94117', '94122', '94131', '94115', '94118',
                '94132', '94114', '94127', '94104', '94123', nan, '94108', '94
                '94107', '94124', '94105', '94013', '941033148', '94158', 'Ca'
                '94143', '95105', '94101', '94120', '94130', '941102019', '941
                '92672', 'CA', '94014', '94129', '94080', '00000', '94544', '9
                '94901', '94402', '94188', '95109', '94621', '95133', '64110',
                '95122', '94602', '94102-5917', '94124-1917', '95117', '95132'
```

```
In [43]: # Импьютация константой
         imp3 = SimpleImputer(missing_values=np.nan, strategy='constant', fill
         data_imp3 = imp3.fit_transform(cat_temp_data)
```

```
In [41]: np.unique(data_imp3)
```

```
Out[41]: array(['00000', '64110', '92672', '94013', '94014', '94080', '941',
                '94101', '94102', '94102-5917', '94103', '941033148', '94104',
                '94105', '94107', '94108', '94109', '94110', '941102019', '941
                '94112', '94114', '94115', '94116', '94117', '94118', '94120',
                '94121', '94122', '94123', '94124', '94124-1917', '94127', '94
                '94130', '94131', '94132', '94133', '94134', '94143', '94158',
                '94188', '94301', '94402', '94544', '94602', '94621', '94901',
                '95105', '95109', '95117', '95122', '95132', '95133', 'CA', 'C
                'None'], dtype=object)
```

```
In [25]: # Количество обработанных значений
         data_cat_const[data_cat_const=='None'].size
```

```
Out[25]: 1241
```

4.2. Для количественного признака

```
In [27]: # Выберем числовые колонки с пропущенными значениями
         # Цикл по колонкам датасета
         num_cols = []
         for col in data.columns:
             # Количество пустых значений
             temp_null_count = data[data[col].isnull()].shape[0]
             dt = str(data[col].dtype)
             if temp_null_count>0 and (dt=='float64' or dt=='int64'):
                 num_cols.append(col)
                 temp_perc = round((temp_null_count / total_count) * 100.0, 2)
                 print('Колонка {}. Тип данных {}. Количество пустых значений
```

```
Колонка business_latitude. Тип данных float64. Количество пустых значений 2400
Колонка business_longitude. Тип данных float64. Количество пустых значений 240
Колонка business_phone_number. Тип данных float64. Количество пустых значений
Колонка inspection_score. Тип данных float64. Количество пустых значений 13947
```

```
In [28]: # Фильтр по колонкам с пропущенными значениями
         data_num = data[num_cols]
         data_num
```

Out[28]:

	business_latitude	business_longitude	business_phone_number
0	NaN	NaN	NaN
1	37.766864	-122.419014	1.415525e+10
2	NaN	NaN	NaN
3	37.710459	-122.447713	1.415546e+10
4	37.714428	-122.411433	NaN
5	37.766618	-122.421263	1.415587e+10
6	37.794298	-122.421387	NaN
7	37.792854	-122.416114	NaN
8	37.772323	-122.509946	NaN
9	37.714428	-122.411433	NaN
10	37.791683	-122.420944	NaN
11	37.782107	-122.483631	NaN
12	NaN	NaN	1.415546e+10
13	37.714428	-122.411433	NaN
14	37.710459	-122.447713	1.415546e+10
15	37.709896	-122.448082	1.415558e+10
16	37.782107	-122.483631	NaN
17	NaN	NaN	NaN
18	37.766618	-122.421263	1.415587e+10
19	37.709896	-122.448082	1.415558e+10
20	37.782107	-122.483631	NaN

21	37.766618	-122.421263	1.415587e+10
22	37.782107	-122.483631	NaN
23	37.790253	-122.415357	NaN
24	37.729016	-122.419253	1.415546e+10
25	NaN	NaN	1.415546e+10
26	37.710459	-122.447713	1.415546e+10
27	37.709896	-122.448082	1.415558e+10
28	37.754397	-122.420915	1.415564e+10
29	37.788673	-122.408524	NaN
…	…	…	…
53656	NaN	NaN	NaN
53657	37.777017	-122.421430	NaN
53658	NaN	NaN	NaN
53659	NaN	NaN	1.415582e+10
53660	37.774722	-122.406761	NaN
53661	NaN	NaN	NaN
53662	NaN	NaN	NaN
53663	37.743206	-122.421546	1.415565e+10
53664	NaN	NaN	NaN
53665	NaN	NaN	NaN
53666	NaN	NaN	NaN
53667	NaN	NaN	NaN
53668	37.767194	-122.435576	1.415562e+10
53669	NaN	NaN	NaN
53670	NaN	NaN	NaN
53671	NaN	NaN	NaN
53672	NaN	NaN	1.415594e+10
53673	NaN	NaN	1.415545e+10
53674	37.797868	-122.407194	NaN
53675	37.715126	-122.398901	1.415546e+10
53676	37.778547	-122.410130	1.415586e+10
53677	37.793730	-122.403974	NaN
53678	NaN	NaN	NaN
53679	NaN	NaN	1.415526e+10
53680	NaN	NaN	NaN
53681	37.742048	-122.499002	NaN
53682	NaN	NaN	NaN
53683	NaN	NaN	NaN
53684	NaN	NaN	NaN
53685	NaN	NaN	NaN

	inspection_score
0	100.0
1	96.0
2	96.0
3	87.0
4	94.0
5	87.0
6	NaN
7	NaN

```
8                       NaN
9                      94.0
10                      NaN
11                     85.0
12                     92.0
13                     94.0
14                     87.0
15                     94.0
16                     85.0
17                     96.0
18                     87.0
19                     94.0
20                     85.0
21                     87.0
22                     85.0
23                      NaN
24                     96.0
25                     92.0
26                     87.0
27                     94.0
28                     92.0
29                     88.0
…                    …
53656                   NaN
53657                  72.0
53658                  89.0
53659                  86.0
53660                  94.0
53661                   NaN
53662                   NaN
53663                  77.0
53664                  90.0
53665                  75.0
53666                   NaN
53667                  77.0
53668                  82.0
53669                  85.0
53670                   NaN
53671                  88.0
53672                   NaN
53673                  76.0
53674                  80.0
53675                  86.0
53676                  92.0
53677                  79.0
53678                   NaN
53679                   NaN
53680                   NaN
53681                  72.0
53682                   NaN
```

```
53683                89.0
53684                85.0
53685                 NaN

[53686 rows x 4 columns]
```

В качестве рассматриваемого количественного признака, имеющего пропуски, выбран столбец “inspection_score”.

Обработка пропусков производится по стратегии импьютации наиболее частыми значениями.

```
In [37]: num_temp_data = data[['inspection_score']]
         num_temp_data
```

```
Out[37]:        inspection_score
         0                 100.0
         1                  96.0
         2                  96.0
         3                  87.0
         4                  94.0
         5                  87.0
         6                   NaN
         7                   NaN
         8                   NaN
         9                  94.0
         10                  NaN
         11                 85.0
         12                 92.0
         13                 94.0
         14                 87.0
         15                 94.0
         16                 85.0
         17                 96.0
         18                 87.0
         19                 94.0
         20                 85.0
         21                 87.0
         22                 85.0
         23                  NaN
         24                 96.0
         25                 92.0
         26                 87.0
         27                 94.0
         28                 92.0
         29                 88.0
         …                  …
         53656               NaN
         53657              72.0
         53658              89.0
         53659              86.0
         53660              94.0
```

```
53661                 NaN
53662                 NaN
53663                77.0
53664                90.0
53665                75.0
53666                 NaN
53667                77.0
53668                82.0
53669                85.0
53670                 NaN
53671                88.0
53672                 NaN
53673                76.0
53674                80.0
53675                86.0
53676                92.0
53677                79.0
53678                 NaN
53679                 NaN
53680                 NaN
53681                72.0
53682                 NaN
53683                89.0
53684                85.0
53685                 NaN

[53686 rows x 1 columns]
```

In [38]:
```
# Количество пропущенных значений в колонке
num_temp_data[num_temp_data['inspection_score'].isnull()].shape[0]
```

Out[38]:
```
13947
```

In [39]:
```
num_temp_data['inspection_score'].unique()
```

Out[39]:
```
array([ 100.,   96.,   87.,   94.,   nan,   85.,   92.,   88.,   90.,
         98.,   83.,   76.,   80.,   93.,   57.,   91.,   68.,   86.,
         77.,   84.,   89.,   81.,   82.,   73.,   74.,   75.,   71.,
         79.,   78.,   69.,   72.,   70.,   63.,   67.,   61.,   66.,
         65.,   55.,   56.,   64.,   59.,   62.,   53.,   60.,   48.,
         58.,   45.,   51.,   54.])
```

In [56]:
```
# Функция для импьютации
def test_num_impute_col(dataset, column, strategy_param):
    temp_data = dataset[[column]]

    indicator = MissingIndicator()
    mask_missing_values_only = indicator.fit_transform(temp_data)

    imp_num = SimpleImputer(strategy=strategy_param)
    data_num_imp = imp_num.fit_transform(temp_data)
```

```
            filled_data = data_num_imp[mask_missing_values_only]

        #    return column, strategy_param, filled_data.size, filled_data[0],
            return data_num_imp, column, strategy_param, filled_data.size, fi
```

```
In [57]: data[['inspection_score']].describe()
```

```
Out[57]:        inspection_score
         count       39739.000000
         mean           85.984071
         std             8.647772
         min            45.000000
         25%            81.000000
         50%            87.000000
         75%            92.000000
         max           100.000000
```

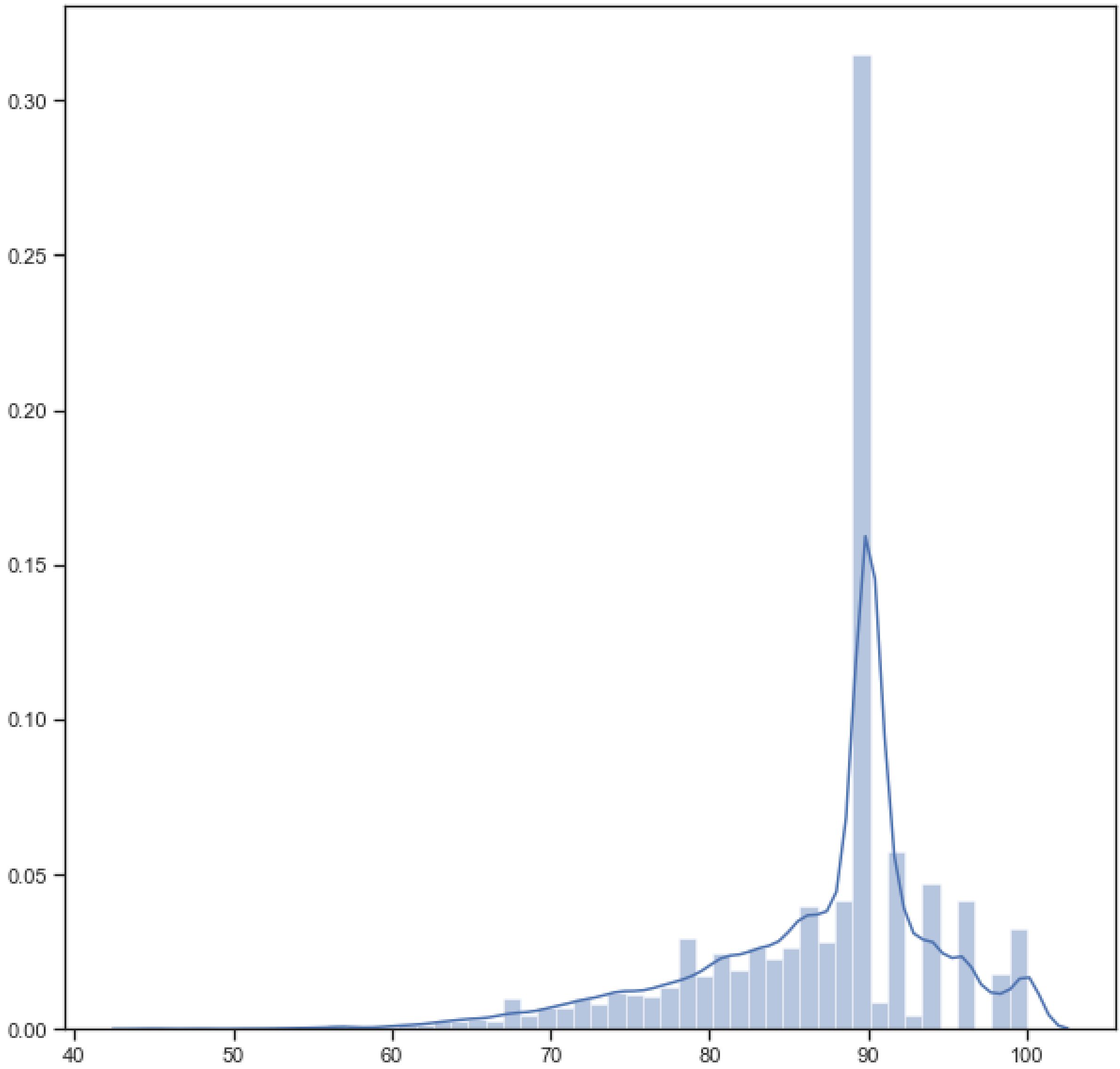
```
In [58]: data_num_imp = test_num_impute_col(data, 'inspection_score', 'most_fr
         data_num_imp
```

```
Out[58]: (array([[ 100.],
                 [  96.],
                 [  96.],
                 …,
                 [  89.],
                 [  85.],
                 [  90.]]), 'inspection_score', 'most_frequent', 13947, 90.0,
```

4.2.1. Визуализация

```
In [60]: fig, ax = plt.subplots(figsize=(10,10))
         sns.distplot(data_num_imp[0])
```

```
Out[60]: <matplotlib.axes._subplots.AxesSubplot at 0x21c41d51e10>
```



Список литературы

[1] Гапанюк Ю. Е. Лабораторная работа «Обработка пропусков в данных, кодирование категориальных признаков, масштабирование данных» [Электронный ресурс] // GitHub. — 2019. — Режим доступа: `https://github.com/ugapanyuk/ml_course/wiki/LAB_MISSING` (дата обращения: 05.04.2019).