Московский государственный технический
университет им. Н.Э. Баумана.

Факультет «Информатика и управление»

Кафедра «Системы обработки информации и управления»

Курс «Технологии машинного обучения»

Отчет по рубежному контролю №1

Вариант 15

Выполнила:
студентка группы ИУ5-64Б
Носова Элина
Подпись и дата:

Проверил:
преподаватель каф.ИУ5
Гапанюк Ю. Е.
Подпись и дата:

Москва, 2022 г.

Условие (задача 2):

Для заданного набора данных проведите обработку пропусков в данных для одного категориального и одного количественного признака. Какие способы обработки пропусков в данных для категориальных и количественных признаков Вы использовали? Какие признаки Вы будете использовать для дальнейшего построения моделей машинного обучения и почему?

Набор данных (7):

https://www.kaggle.com/san-francisco/sf-restaurant-scores-lives-standard

Загрузка и проверка данных:

```
data=pd.read_csv(r'C:\Users\Элина\Desktop\Учеба\ТМО\restaurant-scores-lives-standard.csv') # импорт датасета
data.head()
```

state	business_postal_code	business_latitude	business_longitude	business_location	business_phone_number	...	inspection_type	violation_id	viola
CA	NaN	NaN	NaN	NaN	1.415043e+10	...	New Ownership	NaN	
CA	94118	NaN	NaN	NaN	1.415724e+10	...	Routine - Unscheduled	97975_20190725_103124	Inac
CA	94110	NaN	NaN	NaN	NaN	...	New Ownership	NaN	
CA	94111	NaN	NaN	NaN	1.415488e+10	...	New Construction	NaN	
CA	94109	NaN	NaN	NaN	NaN	...	New Ownership	85986_20161011_103114	

Информация о данных:

```
# размер набора данных
data.shape
```

```
(53973, 23)
```

```
# типы колонок
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 53973 entries, 0 to 53972
Data columns (total 23 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   business_id               53973 non-null  int64  
 1   business_name             53973 non-null  object 
 2   business_address          53973 non-null  object 
 3   business_city             53973 non-null  object 
 4   business_state            53973 non-null  object 
 5   business_postal_code      52955 non-null  object 
 6   business_latitude         34417 non-null  float64
 7   business_longitude        34417 non-null  float64
 8   business_location         34417 non-null  object 
 9   business_phone_number     17035 non-null  float64
 10  inspection_id             53973 non-null  object 
 11  inspection_date           53973 non-null  object 
 12  inspection_score          40363 non-null  float64
 13  inspection_type           53973 non-null  object 
 14  violation_id              41103 non-null  object 
 15  violation_description     41103 non-null  object 
 16  risk_category             41103 non-null  object 
 17  Neighborhoods (old)       34379 non-null  float64
 18  Police Districts          34379 non-null  float64
 19  Supervisor Districts      34379 non-null  float64
 20  Fire Prevention Districts  34327 non-null  float64
 21  Zip Codes                 34397 non-null  float64
 22  Analysis Neighborhoods    34379 non-null  float64
dtypes: float64(10), int64(1), object(12)
memory usage: 9.5+ MB
```

```
# проверим есть ли пропущенные значения
data.isnull().sum()
```

```
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
```

Так как в столбце business_phone_number много пропусков удалим его.

```
data = data.drop(['business_phone_number'], axis = 1)
data.head()
```

Определим количество пропусков по количественному признаку

```
num_cols = []
total_count = data.shape[0]
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 {}, {}%.'.format(col, dt, temp_null_count, temp_perc))
```

```
Колонка business_latitude. Тип данных float64. Количество пустых значений 19556, 36.23%.
Колонка business_longitude. Тип данных float64. Количество пустых значений 19556, 36.23%.
Колонка inspection_score. Тип данных float64. Количество пустых значений 13610, 25.22%.
Колонка Neighborhoods (old). Тип данных float64. Количество пустых значений 19594, 36.3%.
Колонка Police Districts. Тип данных float64. Количество пустых значений 19594, 36.3%.
Колонка Supervisor Districts. Тип данных float64. Количество пустых значений 19594, 36.3%.
Колонка Fire Prevention Districts. Тип данных float64. Количество пустых значений 19646, 36.4%.
Колонка Zip Codes. Тип данных float64. Количество пустых значений 19576, 36.27%.
Колонка Analysis Neighborhoods. Тип данных float64. Количество пустых значений 19594, 36.3%.
```

Возьмем столбец inspection_score и заполним пропуски в нем медианным значением

```
data_num_exp = data_num[['inspection_score']]
data_num_exp.head()
```

	inspection_score
0	NaN
1	96.0
2	NaN
3	NaN
4	NaN

```
indicator = MissingIndicator()
mask_missing_values_only = indicator.fit_transform(data_num_exp)
strategies=['mean', 'median', 'most_frequent']
```

```
def test_num_impute(strategy_param):
    imp_num = SimpleImputer(strategy=strategy_param)
    data_num_imp = imp_num.fit_transform(data_num_exp)
    return data_num_imp[mask_missing_values_only]
strategies[1], test_num_impute(strategies[1])
```

```
('median', array([87., 87., 87., ..., 87., 87., 87.]))
```

Далее определим количество пропусков по категориальному признаку

```
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 {}, {}%.'.format(col, dt, temp_null_count, temp_perc))
```

```
Колонка business_postal_code. Тип данных object. Количество пустых значений 1018, 1.89%.
Колонка business_location. Тип данных object. Количество пустых значений 19556, 36.23%.
Колонка violation_id. Тип данных object. Количество пустых значений 12870, 23.85%.
Колонка violation_description. Тип данных object. Количество пустых значений 12870, 23.85%.
Колонка risk_category. Тип данных object. Количество пустых значений 12870, 23.85%.
```

Возьмем столбец business_postal_code и заполним пропуски в нем самым часто встречающимся значением

```
cat_temp_data = data[['business_postal_code']]
cat_temp_data.head()
```

	business_postal_code
0	NaN
1	94118
2	94110
3	94111
4	94109

```
cat_temp_data['business_postal_code'].unique()
```

```
array([nan, '94118', '94110', '94111', '94109', '94107', '94133', '94117',
       '94103', '94121', '94108', '94102', '94132', '94104', '94122',
       '94123', '94112', '94115', '94105', '94188', '94114', '94124',
       '94158', '94116', '94134', '94130', '94127', '94131', '94124-1917',
       'Ca', '94101', '94117-3504', '95122', '64110', '94544', '94143',
       '94080', '95132', '95112', '94102-5917', '94105-2907', '94013',
       '94301', '94120', '94105-1420', '94123-3106', '95105', '94602',
       '00000', '941102019', '94901', '94518', '95133', '95117', '94621',
       '94122-1909', '94129', '941033148', 'CA', '941', '92672', '95109'],
      dtype=object)
```

```
# Импьютация наиболее частыми значениями
imp2 = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
data_imp2 = imp2.fit_transform(cat_temp_data)
data_imp2
```

```
array([['94110'],
       ['94118'],
       ['94110'],
       ...,
       ['94105'],
       ['94112'],
       ['94107']], dtype=object)
```

Для дальнейшего построения моделей машинного обучения можно взять признаки: business_name, inspection_date, inspection_score, inspection_type, violation_description, risk_category, т.к. они имеют наибольшую значимость.