

**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 Рубежному контролю №1

Вариант №15

Выполнил:

студент группы ИУ5-63
Миронова Александра

Подпись и дата:

09.04.22

Проверил:

Юрий Евгеньевич Гапанюк

Подпись и дата:

Москва, 2022 г.

Задача №2.

Для заданного набора данных провести обработку пропусков в данных для одного категориального и одного количественного признака. Указать использованные способы обработки пропусков в данных для категориальных и количественных признаков? Указать, какие признаки лучше использовать для дальнейшего построения моделей машинного обучения и почему?

Дополнительные требования по группам:

Для студентов группы ИУ5-63Б - для произвольной колонки данных построить график "Ящик с усами (boxplot)".

RK1.md

Загрузка и первичный анализ данных

Для выполнения задания был выдан датасет с данными о ресторанах города Сан-Франциско

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

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
```

```
data = pd.read_csv('restaurant-scores-lives-standard.csv', sep=",")
data.shape
data.dtypes
```

```
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
```

```
data.head()
```

```
<style scoped> .dataframe tbody tr th:only-of-type { vertical-align: middle; }
```

```
.dataframe tbody tr th {  
    vertical-align: top;  
}
```

```
.dataframe thead th {  
    text-align: right;  
}
```

```
</style>
```

	business_id	business_name	business_address	business_city	business_s
0	101192	Cochinita #2	2 Marina Blvd Fort Mason	San Francisco	CA
1	97975	BREADBELLY	1408 Clement St	San Francisco	CA
2	92982	Great Gold Restaurant	3161 24th St.	San Francisco	CA
3	101389	HOMAGE	214 CALIFORNIA ST	San Francisco	CA
4	85986	Pronto Pizza	798 Eddy St	San Francisco	CA

5 rows × 23 columns

```
data.shape
```

```
(53973, 23)
```

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

```

Обработка пропусков в данных

1. Обработка категориальных значений

В качестве категориальных данных был выбран столбец `risk_category`. Поскольку данный признак позволяет оценить предпринимателям уровень безопасности сотрудничества с организацией, то имеет смысл попробовать заполнить пропуски с помощью встроенных средства импьютации библиотеки `scikit-learn`, чем просто удалить данный столбец из датасета. Тем более, что число пропущенных данных составляет всего 24% от общего числа строк.

```

risk_cat_data = data[['risk_category']]
risk_cat_data

```

```
<style scoped> .dataframe tbody tr th:only-of-type { vertical-align: middle; }
```

```

.dataframe tbody tr th {
    vertical-align: top;
}

```

```

.dataframe thead th {

```

```
text-align: right;  
}
```

</style>

	risk_category
0	NaN
1	Moderate Risk
2	NaN
3	NaN
4	High Risk
...	...
53968	Moderate Risk
53969	NaN
53970	Moderate Risk
53971	Moderate Risk
53972	Low Risk

53973 rows × 1 columns

```
risk_cat_data['risk_category'].unique()
```

```
array([nan, 'Moderate Risk', 'High Risk', 'Low Risk'], dtype=object)
```

В данном случае, считаю, что наиболее грамотно было бы заполнить пропущенные значения константами "Moderate Risk". Так пользователи обратят внимание на возможность риска.

```
from sklearn.impute import SimpleImputer  
from sklearn.impute import MissingIndicator  
# Импутация константой  
imputer1 = SimpleImputer(missing_values=np.nan, strategy='constant', fill_value='Mode  
full_risk_data = imputer1.fit_transform(risk_cat_data)  
full_risk_data
```

```
array(['Moderate Risk'],
      ['Moderate Risk'],
      ['Moderate Risk'],
      ...,
      ['Moderate Risk'],
      ['Moderate Risk'],
      ['Low Risk']], dtype=object)
```

```
full_risk_data.shape
```

```
(53973, 1)
```

Убедимся, что пустые значения отсутствуют

```
np.unique(full_risk_data)
```

```
array(['High Risk', 'Low Risk', 'Moderate Risk'], dtype=object)
```

Теперь заменим в data столбец risk_category новым столбцом без пропусков. Для этого удалим старый столбец и вставим новый.

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

```
<style scoped> .dataframe tbody tr th:only-of-type { vertical-align: middle; }
```

```
.dataframe tbody tr th {
    vertical-align: top;
}
```

```
.dataframe thead th {
    text-align: right;
}
```

```
</style>
```

	business_id	business_name	business_address	business_city	business_s
0	101192	Cochinita #2	2 Marina Blvd Fort Mason	San Francisco	CA

	business_id	business_name	business_address	business_city	business_s
1	97975	BREADBELLY	1408 Clement St	San Francisco	CA
2	92982	Great Gold Restaurant	3161 24th St.	San Francisco	CA
3	101389	HOMAGE	214 CALIFORNIA ST	San Francisco	CA
4	85986	Pronto Pizza	798 Eddy St	San Francisco	CA

5 rows × 23 columns

```
data['risk_category'] = full_risk_data.reshape(-1)
data.head()
```

<style scoped> .dataframe tbody tr th:only-of-type { vertical-align: middle; }

```
.dataframe tbody tr th {
    vertical-align: top;
}
```

```
.dataframe thead th {
    text-align: right;
}
```

</style>

	business_id	business_name	business_address	business_city	business_s
0	101192	Cochinita #2	2 Marina Blvd Fort Mason	San Francisco	CA
1	97975	BREADBELLY	1408 Clement St	San Francisco	CA
2	92982	Great Gold Restaurant	3161 24th St.	San Francisco	CA

	business_id	business_name	business_address	business_city	business_s
3	101389	HOMAGE	214 CALIFORNIA ST	San Francisco	CA
4	85986	Pronto Pizza	798 Eddy St	San Francisco	CA

5 rows × 23 columns

2. Обработка количественных значений

В качестве количественных данных был выбран столбец inspection_score. Этот столбец не стоит удалять из датасета, поскольку он информирует пользователей о том, на сколько соответствует ресторан требованиям проведенной проверки. Пропущенные данные в этом столбце буду заполнять значениями медианы, тк неизвестное оценочное числовое значение можно взять в середине диапазона известных значений.

```
inspection_score_data = data[['inspection_score']]
inspection_score_data
```

```
<style scoped> .dataframe tbody tr th:only-of-type { vertical-align: middle; }
```

```
.dataframe tbody tr th {
    vertical-align: top;
}
```

```
.dataframe thead th {
    text-align: right;
}
```

```
</style>
```

	inspection_score
0	NaN
1	96.0
2	NaN
3	NaN
4	NaN

	inspection_score
...	...
53968	80.0
53969	NaN
53970	92.0
53971	76.0
53972	80.0

53973 rows × 1 columns

```
np.unique(inspection_score_data)
```

```
array([ 45.,  46.,  48.,  51.,  54.,  55.,  57.,  58.,  59.,  60.,  61.,
        62.,  63.,  64.,  65.,  66.,  67.,  68.,  69.,  70.,  71.,  72.,
        73.,  74.,  75.,  76.,  77.,  78.,  79.,  80.,  81.,  82.,  83.,
        84.,  85.,  86.,  87.,  88.,  89.,  90.,  91.,  92.,  93.,  94.,
        96.,  98., 100., nan])
```

```
# Импутация медианой
```

```
imputer2 = SimpleImputer(missing_values=np.nan, strategy='median')
full_inspection_score = imputer2.fit_transform(inspection_score_data)
full_inspection_score
```

```
array([[87.],
       [96.],
       [87.],
       ...,
       [92.],
       [76.],
       [80.]])
```

Убедимся, что пустые значения отсутствуют

```
np.unique(full_inspection_score)
```

```
array([ 45.,  46.,  48.,  51.,  54.,  55.,  57.,  58.,  59.,  60.,  61.,
        62.,  63.,  64.,  65.,  66.,  67.,  68.,  69.,  70.,  71.,  72.,
        73.,  74.,  75.,  76.,  77.,  78.,  79.,  80.,  81.,  82.,  83.,
```

84., 85., 86., 87., 88., 89., 90., 91., 92., 93., 94.,
96., 98., 100.]])

```
full_inspection_score.shape
```

```
(53973, 1)
```

Теперь заменим в data столбец inspection_score новым столбцом без пропусков. Для этого удалим старый столбец и вставим новый.

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

```
<style scoped> .dataframe tbody tr th:only-of-type { vertical-align: middle; }
```

```
.dataframe tbody tr th {  
    vertical-align: top;  
}
```

```
.dataframe thead th {  
    text-align: right;  
}
```

```
</style>
```

	business_id	business_name	business_address	business_city	business_s
0	101192	Cochinita #2	2 Marina Blvd Fort Mason	San Francisco	CA
1	97975	BREADBELLY	1408 Clement St	San Francisco	CA
2	92982	Great Gold Restaurant	3161 24th St.	San Francisco	CA
3	101389	HOMAGE	214 CALIFORNIA ST	San Francisco	CA
4	85986	Pronto Pizza	798 Eddy St	San Francisco	CA

5 rows × 23 columns

```
data['inspection_score'] = full_inspection_score.reshape(-1)
data['inspection_score'].head()
```

```
0    87.0
1    96.0
2    87.0
3    87.0
4    87.0
```

```
Name: inspection_score, dtype: float64
```

Дополнительное задание

Построю график "Ящик с усами (boxplot)" для оценок проверки по категориям риска.

```
sns.boxplot( x=data["inspection_score"], y=data["risk_category"])
```

```
<AxesSubplot:xlabel='inspection_score', ylabel='risk_category'>
```

