Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)

Рубежный контроль №2
по курсу «Теория машинного обучения»
Вариант 16

Выполнил
студент группы ИУ5-64Б
Сысойкин Е.М.

Москва, 2020

0.1 РК1; ТМО; Сысойкин Егор; Вариант 16; ИУ5-64Б

0.1.1 Задание.

Для заданного набора данных (по Вашему варианту) постройте модели классификации или регрессии (в зависимости от конкретной задачи, рассматриваемой в наборе данных). Для построения моделей используйте методы линейная/логистическая регрессия и градиентный бустинг. Оцените качество моделей на основе подходящих метрик качества (не менее двух метрик). Какие метрики качества Вы использовали и почему? Какие выводы Вы можете сделать о качестве построенных моделей? Для построения моделей необходимо выполнить требуемую предобработку данных: заполнение пропусков, кодирование категориальных признаков, и т.д.

- При решении задач можно выбирать любое подмножество признаков из приведенного набора данных.
- Для сокращения времени построения моделей можно использовать фрагмент набора данных (например, первые 200-500 строк).

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

[1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.linear_model import LinearRegression
from sklearn.linear_model import LogisticRegression
```

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

[2]:
```
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
```

[3]:
```
data.isnull().sum()
```

[3]:
```
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
```

[4]:
```
data.shape
```

[4]:
```
(53973, 23)
```

[5]:
```
data.head()
```

[5]:
```
   business_id            business_name           business_address  \
0       101192              Cochinita #2  2 Marina Blvd Fort Mason
1        97975                BREADBELLY           1408 Clement St
2        92982  Great Gold Restaurant             3161 24th St.
3       101389                    HOMAGE         214 CALIFORNIA ST
4        85986               Pronto Pizza               798 Eddy St
```

```
  business_city business_state business_postal_code 
 ↪business_latitude  \
0  San Francisco             CA                  NaN                
 ↪NaN
1  San Francisco             CA                94118                
 ↪NaN
2  San Francisco             CA                94110                
 ↪NaN
3  San Francisco             CA                94111                
 ↪NaN
4  San Francisco             CA                94109                
 ↪NaN

  business_longitude business_location  business_phone_number  …  \
0                NaN               NaN           1.415043e+10  …
1                NaN               NaN           1.415724e+10  …
2                NaN               NaN                    NaN  …
3                NaN               NaN           1.415488e+10  …
4                NaN               NaN                    NaN  …

        inspection_type           violation_id  \
0         New Ownership                    NaN
1  Routine - Unscheduled  97975_20190725_103124
2         New Ownership                    NaN
3      New Construction                    NaN
4         New Ownership  85986_20161011_103114

                             violation_description  risk_category  \
0                                              NaN            NaN
1  Inadequately cleaned or sanitized food contact…  Moderate Risk
2                                              NaN            NaN
3                                              NaN            NaN
4                     High risk vermin infestation      High Risk

  Neighborhoods (old) Police Districts Supervisor Districts  \
0                 NaN              NaN                  NaN
1                 NaN              NaN                  NaN
2                 NaN              NaN                  NaN
3                 NaN              NaN                  NaN
4                 NaN              NaN                  NaN

  Fire Prevention Districts  Zip Codes  Analysis Neighborhoods
0                       NaN        NaN                     NaN
1                       NaN        NaN                     NaN
2                       NaN        NaN                     NaN
3                       NaN        NaN                     NaN
```

```
4                                NaN         NaN                        NaN

[5 rows x 23 columns]
```

[6]:
```
data2 = data.copy().dropna(axis=0, how='any')
data2.drop_duplicates(keep=False,inplace=True)
```

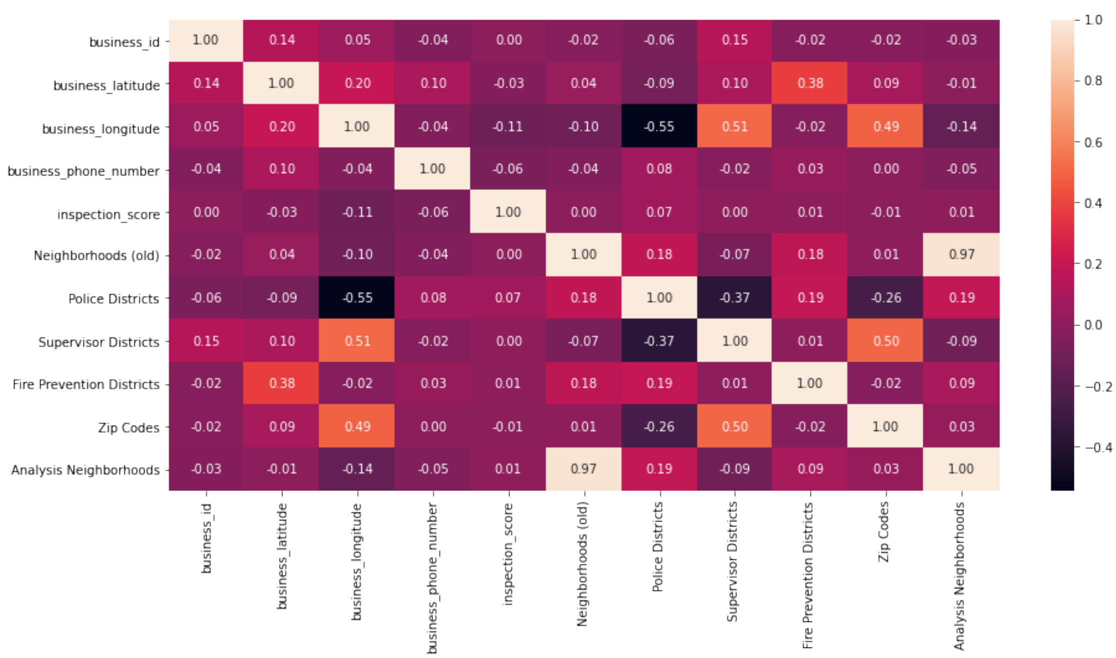
[7]:
```
for col in data2.columns:
    unique_nums = data2[col].unique()
    if unique_nums.size < 10:
        print("{}: {}".format(col, unique_nums))
```

```
business_city: ['San Francisco']
business_state: ['CA']
inspection_type: ['Routine - Unscheduled']
risk_category: ['Low Risk' 'High Risk' 'Moderate Risk']
```

business_city: [‘San Francisco’], business_state: [‘CA’], inspection_type: [‘Routine - Unscheduled’] - имеют 1 уникальное значение. Можно убрать.

[8]:
```
fig, ax = plt.subplots(figsize=(15,7))
sns.heatmap(data2.corr(method='pearson'), ax=ax, annot=True, fmt='.
  ↪2f')
```

[8]: <AxesSubplot:>



[9]:
```
data2["risk_category"] = data2["risk_category"].astype('category')
data2["risk_category_cat"] = data2["risk_category"].cat.codes

data2.drop(["business_city", "business_state", "business_location",
 ↪"business_name",
            "business_address", "violation_description",
 ↪"risk_category", "Neighborhoods (old)", "inspection_type",
            "inspection_id", "violation_id", "inspection_date"
           ],
           axis=1, inplace=True)
```

[10]:
```
data2["business_postal_code"].unique()
```

[10]:
```
array(['94107', '94131', '94112', '94121', '94110', '94109', '94115',
       '94111', '94118', '94103', '94134', '94117', '94114', '94123',
       '94124', '94104', '94122', '94108', '94133', '94132',
 ↪'941102019',
       '94127', '94102', '92672', '94105', '94116', '94158'],
 ↪dtype=object)
```

[11]:
```
data2["business_postal_code"] = data2["business_postal_code"].
 ↪astype(int)
```

[12]:
```
data2["Police Districts"] = data2["Police Districts"].astype(int)
data2["inspection_score"] = data2["inspection_score"].astype(int)
data2["Supervisor Districts"] = data2["Supervisor Districts"].
 ↪astype(int)
data2["Fire Prevention Districts"] = data2["Fire Prevention
 ↪Districts"].astype(int)
data2["Zip Codes"] = data2["Zip Codes"].astype(int)
data2["Analysis Neighborhoods"] = data2["Analysis Neighborhoods"].
 ↪astype(int)
```

[13]:
```
data2.isnull().sum()
```

[13]:
```
business_id                  0
business_postal_code         0
business_latitude            0
business_longitude           0
business_phone_number        0
inspection_score             0
Police Districts             0
Supervisor Districts         0
Fire Prevention Districts    0
Zip Codes                    0
Analysis Neighborhoods       0
```

```
risk_category_cat           0
dtype: int64
```

[14]:
```python
data2.head()
```

[14]:
```
     business_id  business_postal_code  business_latitude  
↪business_longitude  \
11          4794                 94107          37.778634          
↪-122.393089
372         2684                 94131          37.746759          
↪-122.426995
464         3256                 94112          37.709737          
↪-122.450070
484         3951                 94121          37.779962          
↪-122.485087
496         4864                 94110          37.759174          
↪-122.419066

     business_phone_number  inspection_score  Police Districts  \
11            1.415561e+10                71                 2
372           1.415528e+10                87                 7
464           1.415534e+10                94                 7
484           1.415539e+10                77                 6
496           1.415583e+10                78                 4

     Supervisor Districts  Fire Prevention Districts  Zip Codes  \
11                      9                          6      28856
372                     5                          2         63
464                     6                          9      28861
484                     2                         11         55
496                     7                          2      28859

     Analysis Neighborhoods  risk_category_cat
11                       34                  1
372                      22                  0
464                      28                  1
484                      29                  1
496                      20                  1
```

[15]:
```python
data2.dtypes
```

[15]:
```
business_id                    int64
business_postal_code           int64
business_latitude            float64
business_longitude           float64
business_phone_number        float64
```

```
inspection_score                int64
Police Districts                int64
Supervisor Districts            int64
Fire Prevention Districts       int64
Zip Codes                       int64
Analysis Neighborhoods          int64
risk_category_cat                int8
dtype: object
```

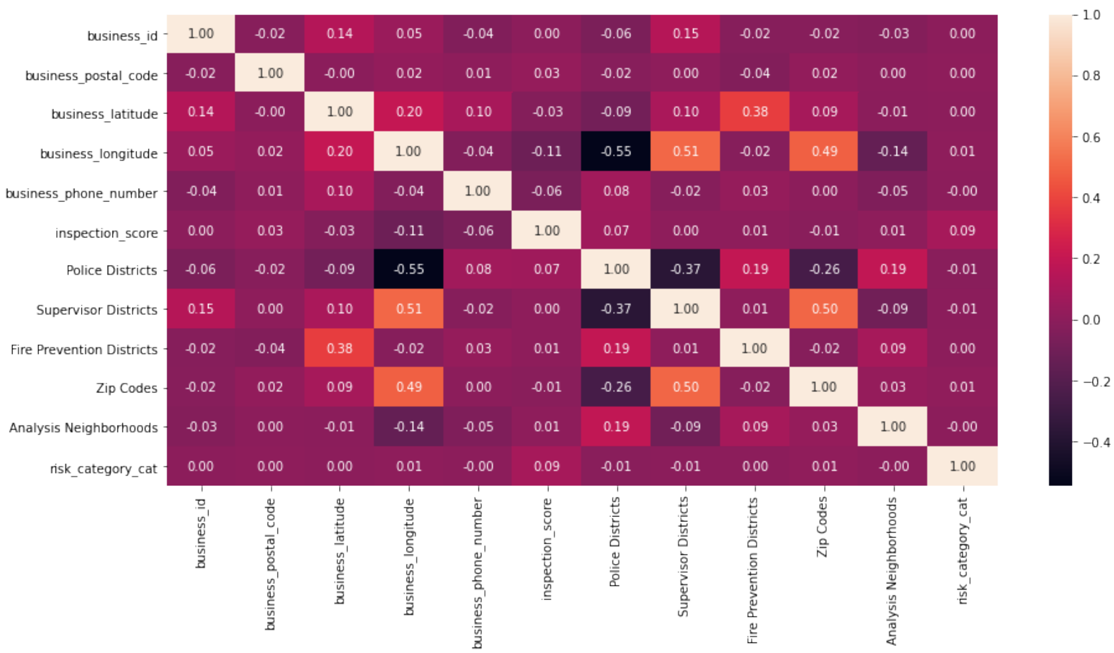
[16]:
```python
target = "Supervisor Districts"
```

[17]:
```python
# Масштабирование
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
for col in data2.columns:
  if col != target:
    data2[col] = scaler.fit_transform(data2[[col]])
```

[18]:
```python
fig, ax = plt.subplots(figsize=(15,7))
sns.heatmap(data2.corr(method='pearson'), ax=ax, annot=True, fmt='.
↪2f')
```

[18]: <AxesSubplot:>



[19]:
```
from sklearn.model_selection import train_test_split

feature_cols = ["business_latitude", "business_longitude", "Zip
 ↪Codes"]

X_train, X_test, y_train, y_test = train_test_split(
  data2[feature_cols],
  data2[target],
  test_size=0.3,
  random_state=1,
)
```

Линейная регреессия

[20]:
```
from sklearn.linear_model import LinearRegression

linreg = LinearRegression().fit(X_train, y_train)
```

[21]:
```
from sklearn.metrics import r2_score, mean_absolute_error

linreg_predict = linreg.predict(X_test)
r2_score(y_test, linreg_predict), \
  mean_absolute_error(y_test, linreg_predict)
```

[21]:
```
(0.32634156097620515, 1.8639363772620643)
```

Градиентный бустинг

[22]:
```
from sklearn.ensemble import GradientBoostingRegressor

gboostreg = GradientBoostingRegressor(random_state=10).fit(X_train,
 ↪y_train)
```

[23]:
```
gboostreg_predict = gboostreg.predict(X_test)
r2_score(y_test, gboostreg_predict), \
  mean_absolute_error(y_test, gboostreg_predict)
```

[23]:
```
(0.9155392208894986, 0.4158270828303186)
```

0.1.2 Вывод

Как видно по тепловой карте, данные плохо коррелируют друг с другом. Поэтому для построения модели был выбрал целевой признак “Supervisor Districts”, а в качестве ключевых признаков - [“business_latitude”, “business_longitude”, “Zip Codes”]. Как видно по оценкам, модель линейной регрессии недообучается, а модель градиентного бустинга хорошо обучается. Вторая модель имеет высокую оценку r2(близкую к 1) и низкую абсолютную ошибку(<1, что для целочисленного признака дает хороший результат).