

МГТУ им. Н.Э. Баумана  
Факультет "Информатика и системы управления"  
Кафедра "Системы обработки информации и управления"

ДИСЦИПЛИНА:  
**"Технологии машинного обучения"**



Отчет по рубежному контролю No1

**«Технологии разведочного анализа и обработки данных»**

**Вариант No22**

**Выполнил:**

Студент группы ИУ5-61Б

Сукиасян В.М.

**Преподаватель:**

Гапанюк Ю.Е.

Москва 2020

## Задание:

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?

+ для пары произвольных колонок данных построить график "Диаграмма рассеяния"

**Датасет:** <https://www.kaggle.com/noriuk/us-education-datasets-unification-project> (файл states\_all\_extended.csv)

## Выполнение РК:

```
In [39]: import numpy as np
import pandas as pd
pd.set_option('display.max.columns', 100)
# to draw pictures in jupyter notebook
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
# we don't like warnings
# you can comment the following 2 lines if you'd like to
import warnings
warnings.filterwarnings('ignore')
from sklearn.impute import SimpleImputer
from sklearn.impute import MissingIndicator
```

```
In [40]: data = pd.read_csv('states_all_extended.csv')
data.head()
```

Out[40]:

	PRIMARY_KEY	STATE	YEAR	ENROLL	TOTAL_REVENUE	FEDERAL_REVENUE	STATE_REVENUE	LOCAL_
0	1992_ALABAMA	ALABAMA	1992	NaN	2678885.0	304177.0	1659028.0	715680.0
1	1992_ALASKA	ALASKA	1992	NaN	1049591.0	106780.0	720711.0	222100.0
2	1992_ARIZONA	ARIZONA	1992	NaN	3258079.0	297888.0	1369815.0	1590376.0
3	1992_ARKANSAS	ARKANSAS	1992	NaN	1711959.0	178571.0	958785.0	574603.0
4	1992_CALIFORNIA	CALIFORNIA	1992	NaN	26260025.0	2072470.0	16546514.0	7641041.0

5 rows × 230 columns

```
In [41]: #смотрим на типы колонок
data.dtypes
```

```
Out[41]: PRIMARY_KEY      object
STATE                    object
YEAR                     int64
ENROLL                   float64
TOTAL_REVENUE            float64
...
PK_WH_M                  float64
AVG_MATH_4_SCORE         float64
AVG_MATH_8_SCORE         float64
AVG_READING_4_SCORE      float64
AVG_READING_8_SCORE      float64
Length: 230, dtype: object
```

```
In [42]: # проверим есть ли пропущенные значения
data.isnull().sum()
```

```
Out[42]: PRIMARY_KEY      0
        STATE            0
        YEAR            0
        ENROLL          694
        TOTAL_REVENUE    643
        ...
        PK_WH_M         1524
        AVG_MATH_4_SCORE 1383
        AVG_MATH_8_SCORE 1387
        AVG_READING_4_SCORE 1386
        AVG_READING_8_SCORE 1421
        Length: 230, dtype: int64
```

```
In [43]: #размер df
        total_count = data.shape[0]
```

```
In [44]: # Выберем числовые колонки с пропущенными значениями
        num_cols = []
        total_count = data.shape[0]
        for col in data.columns:
            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 ENROLL. Тип данных float64. Количество пустых значений 694, 36.18%.

Колонка TOTAL\_REVENUE. Тип данных float64. Количество пустых значений 643, 33.52%.

Колонка FEDERAL\_REVENUE. Тип данных float64. Количество пустых значений 643, 33.52%.

Колонка STATE\_REVENUE. Тип данных float64. Количество пустых значений 643, 33.52%.

Колонка LOCAL\_REVENUE. Тип данных float64. Количество пустых значений 643, 33.52%.

Колонка TOTAL\_EXPENDITURE. Тип данных float64. Количество пустых значений 643, 33.52%.

Колонка INSTRUCTION\_EXPENDITURE. Тип данных float64. Количество пустых значений 643, 33.52%.

Колонка SUPPORT\_SERVICES\_EXPENDITURE. Тип данных float64. Количество пустых значений 643, 33.52%.

Колонка OTHER\_EXPENDITURE. Тип данных float64. Количество пустых значений 694, 36.18%.

Колонка CAPITAL\_OUTLAY\_EXPENDITURE. Тип данных float64. Количество пустых значений 643, 33.52%.

Колонка A\_A\_A. Тип данных float64. Количество пустых значений 286, 14.91%.

Колонка G01\_A\_A. Тип данных float64. Количество пустых значений 286, 14.91%.

Колонка G02\_A\_A. Тип данных float64. Количество пустых значений 286, 14.91%.

Колонка G03\_A\_A. Тип данных float64. Количество пустых значений 286, 14.91%.

Колонка G04\_A\_A. Тип данных float64. Количество пустых значений 286, 14.91%.

Колонка G05\_A\_A. Тип данных float64. Количество пустых значений 286, 14.91%.

Колонка G06\_A\_A. Тип данных float64. Количество пустых значений 286, 14.91%.

Колонка G07\_A\_A. Тип данных float64. Количество пустых значений 286, 14.91%.

Колонка G08\_A\_A. Тип данных float64. Количество пустых значений 286, 14.91%.

Колонка G09\_A\_A. Тип данных float64. Количество пустых значений 286, 14.91%.

Колонка G10\_A\_A. Тип данных float64. Количество пустых значений 286, 14.91%.

Колонка G11\_A\_A. Тип данных float64. Количество пустых значений 286, 14.91%.

Колонка G12\_A\_A. Тип данных float64. Количество пустых значений 286, 14.91%.

Колонка KG\_A\_A. Тип данных float64. Количество пустых значений 286, 14.91%.

Колонка PK\_A\_A. Тип данных float64. Количество пустых значений 376, 19.6%.

Колонка G01-G08\_A\_A. Тип данных float64. Количество пустых значений 898, 46.82%.

...

Колонка PK\_TR\_F. Тип данных float64. Количество пустых значений 1560, 81.33%.

Колонка PK\_TR\_M. Тип данных float64. Количество пустых значений 1560, 81.33%.

Колонка PK\_WH\_F. Тип данных float64. Количество пустых значений 1524, 79.46%.

Колонка PK\_WH\_M. Тип данных float64. Количество пустых значений 1524, 79.46%.

Колонка AVG\_MATH\_4\_SCORE. Тип данных float64. Количество пустых значений 1383, 72.11%.

Колонка AVG\_MATH\_8\_SCORE. Тип данных float64. Количество пустых значений 1387, 72.31%.

Колонка AVG\_READING\_4\_SCORE. Тип данных float64. Количество пустых значений 1386, 72.26%.

Колонка AVG\_READING\_8\_SCORE. Тип данных float64. Количество пустых значений 1421, 74.09%.

```
In [45]: #возьмем колонку Enroll
        # Запоминаем индексы строк с пустыми значениями
        flt_index = data[data['ENROLL'].isnull()].index
        flt_index
```

```
Out[45]: Int64Index([ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9,
        ...,
        1908, 1909, 1910, 1911, 1912, 1913, 1914, 1915, 1916, 1917],
        dtype='int64', length=694)
```

```
In [46]: data_enroll = data[num_cols][['ENROLL']]
        data_enroll.head()
```

```
Out[46]:
```

	ENROLL
0	NaN
1	NaN
2	NaN
3	NaN
4	NaN

```
In [47]: # Фильтр для проверки заполнения пустых значений
indicator = MissingIndicator()
mask_missing_values_only = indicator.fit_transform(data_enroll)
mask_missing_values_only
```

```
Out[47]: array([[ True],
 [ True],
 [ True],
 ...,
 [ True],
 [ True],
 [ True]])
```

```
In [48]: strategy='mean'
```

```
In [49]: def test_num_impute(strategy_param):
imp_num = SimpleImputer(strategy=strategy_param)
data_num_imp = imp_num.fit_transform(data_enroll)
return data_num_imp[mask_missing_values_only]
```

```
In [53]: new_enroll = pd.DataFrame({'id': flt_index,
'ENROLL':test_num_impute('mean')})
new_enroll
```

```
Out[53]:
```

	id	ENROLL
0	0	917541.566176
1	1	917541.566176
2	2	917541.566176
3	3	917541.566176
4	4	917541.566176
...	...	...
689	1913	917541.566176
690	1914	917541.566176
691	1915	917541.566176
692	1916	917541.566176
693	1917	917541.566176

694 rows x 2 columns

```
In [55]: for index, row in new_enroll.iterrows():
data.loc[row['id'], 'ENROLL'] = row['ENROLL']
data
#очистили данные для колонки ENROLL
```

```
Out[55]:
```

	PRIMARY_KEY	STATE	YEAR	ENROLL	TOTAL_REVENUE	FEDERAL_REVENUE	STA1
0	1992_ALABAMA	ALABAMA	1992	917541.566176	2678885.0	304177.0	1659
1	1992_ALASKA	ALASKA	1992	917541.566176	1049591.0	106780.0	7207
2	1992_ARIZONA	ARIZONA	1992	917541.566176	3258079.0	297888.0	1369
3	1992_ARKANSAS	ARKANSAS	1992	917541.566176	1711959.0	178571.0	9587
4	1992_CALIFORNIA	CALIFORNIA	1992	917541.566176	26260025.0	2072470.0	1654
...	...	...	...	...	...	...	...
1913	2017_NORTH_DAKOTA	NORTH_DAKOTA	2017	917541.566176	NaN	NaN	NaN
1914	2017_RHODE_ISLAND	RHODE_ISLAND	2017	917541.566176	NaN	NaN	NaN
1915	2017_SOUTH_CAROLINA	SOUTH_CAROLINA	2017	917541.566176	NaN	NaN	NaN
1916	2017_SOUTH_DAKOTA	SOUTH_DAKOTA	2017	917541.566176	NaN	NaN	NaN
1917	2017_WEST_VIRGINIA	WEST_VIRGINIA	2017	917541.566176	NaN	NaN	NaN

1918 rows x 230 columns

```
In [59]: # Выберем категориальные колонки с пропущенными значениями
cat_cols = []
for col in data.columns:
    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))
```

В данном датасете нет категориальных признаков с пропущенными данными, поэтому возьмем другой датасет и очистим категориальный признак там

```
In [76]: data1 = pd.read_csv('COVID19_line_list_data.csv')
data1.head()
```

```
Out[76]:
```

	id	case_in_country	reporting date	Unnamed: 3	summary	location	country	gender	age	symptom_onset	If_onset_app
0	1	NaN	1/20/2020	NaN	First confirmed imported COVID-19 pneumonia pa...	Shenzhen, Guangdong	China	male	66.0	01/03/20	0.0
1	2	NaN	1/20/2020	NaN	First confirmed imported COVID-19 pneumonia pa...	Shanghai	China	female	56.0	1/15/2020	0.0
2	3	NaN	1/21/2020	NaN	First confirmed imported cases in Zhejiang: pa...	Zhejiang	China	male	46.0	01/04/20	0.0
3	4	NaN	1/21/2020	NaN	new confirmed imported COVID-19 pneumonia in T...	Tianjin	China	female	60.0	NaN	NaN
4	5	NaN	1/21/2020	NaN	new confirmed imported COVID-19 pneumonia in T...	Tianjin	China	male	58.0	NaN	NaN

```
In [77]: #проверим наличие категориальных признаков в данно датасете
total_count1 = data1.shape[0]
cat_cols1 = []
for col in data1.columns:
    temp_null_count = data1[data1[col].isnull()].shape[0]
    dt = str(data1[col].dtype)
    if temp_null_count>0 and (dt=='object'):
        cat_cols1.append(col)
        temp_perc = round((temp_null_count / total_count1) * 100.0, 2)
        print('Колонка {}. Тип данных {}. Количество пустых значений {}, {}%'.format(col, dt, temp_null_count, temp_perc))
```

Колонка reporting date. Тип данных object. Количество пустых значений 1, 0.09%.  
 Колонка summary. Тип данных object. Количество пустых значений 5, 0.46%.  
 Колонка gender. Тип данных object. Количество пустых значений 183, 16.87%.  
 Колонка symptom\_onset. Тип данных object. Количество пустых значений 522, 48.11%.  
 Колонка hosp\_visit\_date. Тип данных object. Количество пустых значений 578, 53.27%.  
 Колонка exposure\_start. Тип данных object. Количество пустых значений 957, 88.2%.  
 Колонка exposure\_end. Тип данных object. Количество пустых значений 744, 68.57%.  
 Колонка symptom. Тип данных object. Количество пустых значений 815, 75.12%.

```
In [78]: #для очистки возьмем столбец gender
cat_temp_data = data1[['gender']]
cat_temp_data.head()
```

```
Out[78]:
```

	gender
0	male
1	female
2	male
3	female
4	male

```
In [79]: cat_temp_data['gender'].unique()
```

```
Out[79]: array(['male', 'female', nan], dtype=object)
```

```
In [80]: imp = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
data_imp = imp.fit_transform(cat_temp_data)
data_imp
```

```
Out[80]: array([[ 'male'],
                [ 'female'],
                [ 'male'],
                ...,
                [ 'male'],
                [ 'male'],
                [ 'male']], dtype=object)
```

```
In [81]: # Пустые значения отсутствуют
np.unique(data_imp)
```

```
Out[81]: array(['female', 'male'], dtype=object)
```

```
In [82]: cat_enc = pd.DataFrame({'gender':data_imp.T[0]})
cat_enc
```

```
Out[82]:
```

	gender
0	male
1	female
2	male
3	female
4	male
...	...
1080	male
1081	male
1082	male
1083	male

1084	male
------	------

1085 rows x 1 columns

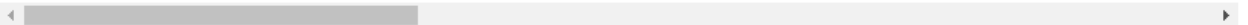
```
In [84]: for index, row in cat_enc.iterrows():
data1.loc[index, 'gender'] = row['gender']
data1
```

```
Out[84]:
```

	id	case_in_country	reporting date	Unnamed: 3	summary	location	country	gender	age	symptom_onset
0	1	NaN	1/20/2020	NaN	First confirmed imported COVID-19 pneumonia pa...	Shenzhen, Guangdong	China	male	66.0	01/03/20

1	2	NaN	1/20/2020	NaN	First confirmed imported COVID-19 pneumonia pa...	Shanghai	China	female	56.0	1/15/2020
2	3	NaN	1/21/2020	NaN	First confirmed imported cases in Zhejiang: pa...	Zhejiang	China	male	46.0	01/04/20
3	4	NaN	1/21/2020	NaN	new confirmed imported COVID-19 pneumonia in T...	Tianjin	China	female	60.0	NaN
4	5	NaN	1/21/2020	NaN	new confirmed imported COVID-19 pneumonia in T...	Tianjin	China	male	58.0	NaN
...	...	...	...	...	...	...	...	...	...	...
1080	1081	2.0	2/25/2020	NaN	new COVID-19 patient confirmed in Austria: 24,...	Innsbruck	Austria	male	24.0	NaN
1081	1082	1.0	2/24/2020	NaN	new COVID-19 patient confirmed in Afghanistan:...	Afghanistan	Afghanistan	male	35.0	NaN
1082	1083	1.0	2/26/2020	NaN	new COVID-19 patient confirmed in Algeria: mal...	Algeria	Algeria	male	NaN	NaN
1083	1084	1.0	2/25/2020	NaN	new COVID-19 patient confirmed in Croatia: mal...	Croatia	Croatia	male	NaN	NaN
1084	1085	1.0	2/25/2020	NaN	new COVID-19 patient confirmed in	Bern	Switzerland	male	70.0	2/17/2020

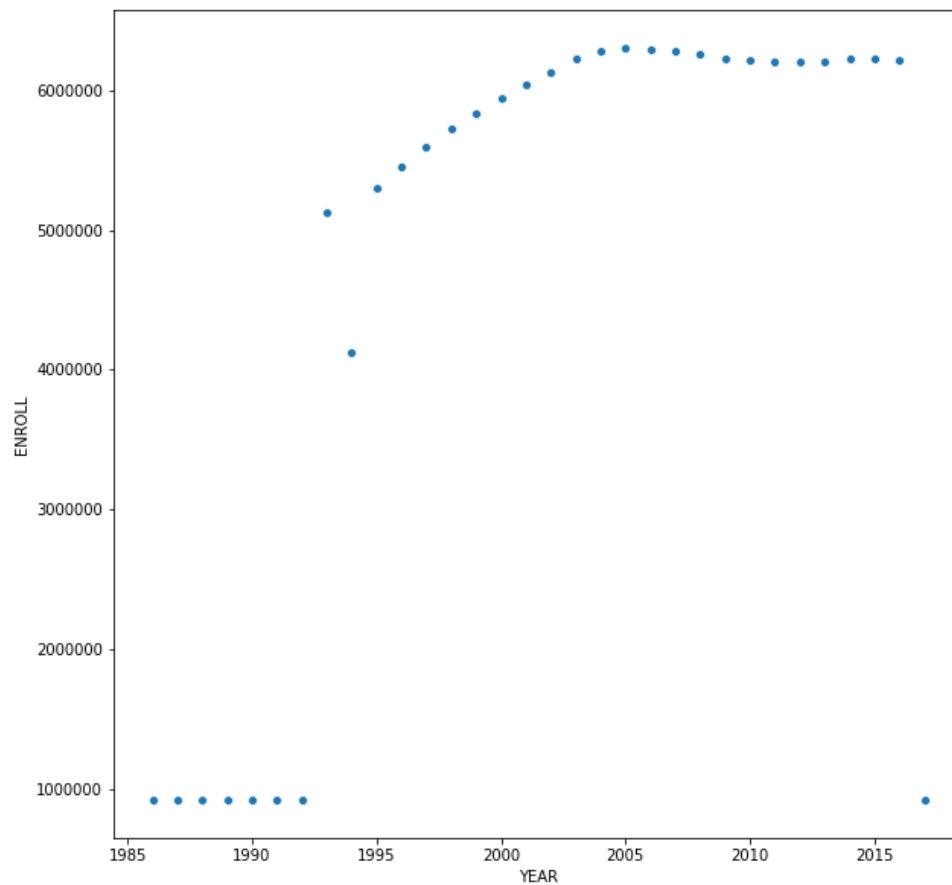
1085 rows x 27 columns



In [87]: *# диаграмма рассеяния для первого датасета для штата КАЛИФОРНИЯ*  
*# зависимость года поступления и числа поступающих*

```
calif_df = data[data['STATE'] == 'CALIFORNIA']
fig, ax = plt.subplots(figsize=(10,10))
sns.scatterplot(ax=ax, x='YEAR', y='ENROLL', data=calif_df)
```

Out[87]: <matplotlib.axes.\_subplots.AxesSubplot at 0xc2d2b88>



Таким образом для обработки пропусков в данных для количественного признака использовался метод импьютации средними значениями, а для категориальных признаков - импьютации наиболее частыми значениями. Для дальнейшего построения моделей можно в принципе использовать все столбцы, обработав предварительно пропуски, в зависимости от нужд исследований.



