МГТУ им. Н. Э. Баумана, кафедра ИУ5

курс “Технология машинного обучения”

Лабораторная работа №2

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

ВЫПОЛНИЛ:

Нагдимаев И. И.

Группа: ИУ5-65Б

ПРОВЕРИЛ:

Гапанюк Ю.Е.

Москва 2020

Цель лабораторной работы: изучение способов предварительной обработки данных для дальнейшего формирования моделей.

Задание:

1. 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.)
2. Для выбранного датасета (датасетов) на основе материалов лекции решить следующие задачи:

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

Выполненная работа

In [1]:
```
import pandas as pd
import numpy as np
```

In [2]:
```
import os
import tarfile
from six.moves import urllib

GT_PATH = os.path.join("dataset", "gt")
```

In [3]:
```
def load_gt_data(gt_path=GT_PATH):
    csv_path = os.path.join(gt_path, "GlobalLandTemperaturesByCountry.csv")
    return pd.read_csv(csv_path)
```

In [4]:
```
gt = load_gt_data()
gt.head()
```

Out[4]:

	dt	AverageTemperature	AverageTemperatureUncertainty	Country
0	1743-11-01	4.384	2.294	Åland
1	1743-12-01	NaN	NaN	Åland
2	1744-01-01	NaN	NaN	Åland
3	1744-02-01	NaN	NaN	Åland
4	1744-03-01	NaN	NaN	Åland

Кодирование категориальных признаков

Категориальные признаки

In [5]:
```
gt['dt'].value_counts()
```

Out[5]:
```
2000-10-01    243
1994-10-01    243
1995-05-01    243
2005-06-01    243
1993-11-01    243
             ...
1750-10-01     50
1752-08-01     50
1749-07-01     50
1748-11-01     50
1744-12-01     50
Name: dt, Length: 3239, dtype: int64
```

Кодирование с помощью LabelEncoder

In [6]:
```
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
cat_enc_le = le.fit_transform(gt['dt'])
cat_enc_le
```

Out[6]:
```
array([   0,    1,    2, ..., 3236, 3237, 3238])
```

In [7]:
```
np.unique(cat_enc_le)
```

Out[7]:
```
array([   0,    1,    2, ..., 3236, 3237, 3238])
```

In [8]:
```
le.inverse_transform([   0,    1,    2, 3236, 3237, 3238])
```

Out[8]:
```
array(['1743-11-01', '1743-12-01', '1744-01-01', '2013-07-01',
       '2013-08-01', '2013-09-01'], dtype=object)
```

In [9]:
```
gt['dt'] = cat_enc_le
gt
```

Out[9]:

	dt	AverageTemperature	AverageTemperatureUncertainty	Country
0	0	4.384	2.294	Åland
1	1	NaN	NaN	Åland
2	2	NaN	NaN	Åland
3	3	NaN	NaN	Åland
4	4	NaN	NaN	Åland
...	...	...	...	...
577457	3234	19.059	1.022	Zimbabwe
577458	3235	17.613	0.473	Zimbabwe
577459	3236	17.000	0.453	Zimbabwe
577460	3237	19.759	0.717	Zimbabwe
577461	3238	NaN	NaN	Zimbabwe

577462 rows × 4 columns

Кодирование с помощью OneHotEncoder

In [10]:
```
from sklearn.preprocessing import OneHotEncoder

ohe = OneHotEncoder()
cat_ohe = ohe.fit_transform(gt[['Country']])
```

In [11]:
```
ohe.categories_
```

Out[11]:
```
[array(['Afghanistan', 'Africa', 'Albania', 'Algeria', 'American Samoa',
        'Andorra', 'Angola', 'Anguilla', 'Antarctica',
        'Antigua And Barbuda', 'Argentina', 'Armenia', 'Aruba', 'Asia',
        'Australia', 'Austria', 'Azerbaijan', 'Bahamas', 'Bahrain',
        'Baker Island', 'Bangladesh', 'Barbados', 'Belarus', 'Belgium',
        'Belize', 'Benin', 'Bhutan', 'Bolivia',
        'Bonaire, Saint Eustatius And Saba', 'Bosnia And Herzegovina',
        'Botswana', 'Brazil', 'British Virgin Islands', 'Bulgaria',
        'Burkina Faso', 'Burma', 'Burundi', 'Cambodia', 'Cameroon',
        'Canada', 'Cape Verde', 'Cayman Islands',
        'Central African Republic', 'Chad', 'Chile', 'China',
        'Christmas Island', 'Colombia', 'Comoros', 'Congo',
        'Congo (Democratic Republic Of The)', 'Costa Rica', 'Croatia',
        'Cuba', 'Curaçao', 'Cyprus', 'Czech Republic', "Côte D'Ivoire",
        'Denmark', 'Denmark (Europe)', 'Djibouti', 'Dominica',
        'Dominican Republic', 'Ecuador', 'Egypt', 'El Salvador',
        'Equatorial Guinea', 'Eritrea', 'Estonia', 'Ethiopia', 'Europe',
        'Falkland Islands (Islas Malvinas)', 'Faroe Islands',
        'Federated States Of Micronesia', 'Fiji', 'Finland', 'France',
        'France (Europe)', 'French Guiana', 'French Polynesia',
        'French Southern And Antarctic Lands', 'Gabon', 'Gambia',
        'Gaza Strip', 'Georgia', 'Germany', 'Ghana', 'Greece', 'Greenland',
        'Grenada', 'Guadeloupe', 'Guam', 'Guatemala', 'Guernsey', 'Guinea',
        'Guinea Bissau', 'Guyana', 'Haiti',
        'Heard Island And Mcdonald Islands', 'Honduras', 'Hong Kong',
        'Hungary', 'Iceland', 'India', 'Indonesia', 'Iran', 'Iraq',
        'Ireland', 'Isle Of Man', 'Israel', 'Italy', 'Jamaica', 'Japan',
        'Jersey', 'Jordan', 'Kazakhstan', 'Kenya', 'Kingman Reef',
        'Kiribati', 'Kuwait', 'Kyrgyzstan', 'Laos', 'Latvia', 'Lebanon',
        'Lesotho', 'Liberia', 'Libya', 'Liechtenstein', 'Lithuania',
        'Luxembourg', 'Macau', 'Macedonia', 'Madagascar', 'Malawi',
        'Malaysia', 'Mali', 'Malta', 'Martinique', 'Mauritania',
        'Mauritius', 'Mayotte', 'Mexico', 'Moldova', 'Monaco', 'Mongolia',
        'Montenegro', 'Montserrat', 'Morocco', 'Mozambique', 'Namibia',
        'Nepal', 'Netherlands', 'Netherlands (Europe)', 'New Caledonia',
        'New Zealand', 'Nicaragua', 'Niger', 'Nigeria', 'Niue',
        'North America', 'North Korea', 'Northern Mariana Islands',
        'Norway', 'Oceania', 'Oman', 'Pakistan', 'Palau', 'Palestina',
        'Palmyra Atoll', 'Panama', 'Papua New Guinea', 'Paraguay', 'Peru',
        'Philippines', 'Poland', 'Portugal', 'Puerto Rico', 'Qatar',
        'Reunion', 'Romania', 'Russia', 'Rwanda', 'Saint Barthélemy',
        'Saint Kitts And Nevis', 'Saint Lucia', 'Saint Martin',
        'Saint Pierre And Miquelon', 'Saint Vincent And The Grenadines',
        'Samoa', 'San Marino', 'Sao Tome And Principe', 'Saudi Arabia',
        'Senegal', 'Serbia', 'Seychelles', 'Sierra Leone', 'Singapore',
        'Sint Maarten', 'Slovakia', 'Slovenia', 'Solomon Islands',
        'Somalia', 'South Africa', 'South America',
        'South Georgia And The South Sandwich Isla', 'South Korea',
        'Spain', 'Sri Lanka', 'Sudan', 'Suriname',
        'Svalbard And Jan Mayen', 'Swaziland', 'Sweden', 'Switzerland',
        'Syria', 'Taiwan', 'Tajikistan', 'Tanzania', 'Thailand',
        'Timor Leste', 'Togo', 'Tonga', 'Trinidad And Tobago', 'Tunisia',
        'Turkey', 'Turkmenistan', 'Turks And Caicos Islands', 'Uganda',
        'Ukraine', 'United Arab Emirates', 'United Kingdom',
        'United Kingdom (Europe)', 'United States', 'Uruguay',
        'Uzbekistan', 'Venezuela', 'Vietnam', 'Virgin Islands',
        'Western Sahara', 'Yemen', 'Zambia', 'Zimbabwe', 'Åland'],
       dtype=object)]
```


In [12]:
```
cat_ohe.toarray()
#cat_ohe.A
```

Out[12]:
```
array([[0., 0., 0., ..., 0., 0., 1.],
       [0., 0., 0., ..., 0., 0., 1.],
       [0., 0., 0., ..., 0., 0., 1.],
       ...,
       [0., 0., 0., ..., 0., 1., 0.],
       [0., 0., 0., ..., 0., 1., 0.],
       [0., 0., 0., ..., 0., 1., 0.]])
```

In [13]:
```
pd.get_dummies(gt['Country'])
```

Out[13]:

	Afghanistan	Africa	Albania	Algeria	American Samoa	Andorra	Angola	Anguilla	Antarctica	Antigua And Barbuda	...	Uruguay	Uzbekista
0	0	0	0	0	0	0	0	0	0	0	...	0	
1	0	0	0	0	0	0	0	0	0	0	...	0	
2	0	0	0	0	0	0	0	0	0	0	...	0	
3	0	0	0	0	0	0	0	0	0	0	...	0	
4	0	0	0	0	0	0	0	0	0	0	...	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	
577457	0	0	0	0	0	0	0	0	0	0	...	0	
577458	0	0	0	0	0	0	0	0	0	0	...	0	
577459	0	0	0	0	0	0	0	0	0	0	...	0	
577460	0	0	0	0	0	0	0	0	0	0	...	0	
577461	0	0	0	0	0	0	0	0	0	0	...	0	

577462 rows × 243 columns

Обработка пропусков в данных

Первый способ определить признаки с нулевыми значениями

total_bedrooms имеет 20433 ненулевых объекта из 20640

In [14]:
```
gt.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 577462 entries, 0 to 577461
Data columns (total 4 columns):
 #   Column                         Non-Null Count   Dtype  
---  ------                         --------------   -----  
 0   dt                             577462 non-null  int32  
 1   AverageTemperature             544811 non-null  float64
 2   AverageTemperatureUncertainty  545550 non-null  float64
 3   Country                        577462 non-null  object 
dtypes: float64(2), int32(1), object(1)
memory usage: 15.4+ MB
```

Второй способ

In [15]:
```
gt.isnull().sum()
```

Out[15]:
```
dt                                 0
AverageTemperature             32651
AverageTemperatureUncertainty  31912
Country                            0
dtype: int64
```

In [16]:
```
sample_incomplete_rows = gt[gt.isnull().any(axis=1)].head()
sample_incomplete_rows
```

Out[16]:

	dt	AverageTemperature	AverageTemperatureUncertainty	Country
1	1	NaN	NaN	Åland
2	2	NaN	NaN	Åland
3	3	NaN	NaN	Åland
4	4	NaN	NaN	Åland
9	9	NaN	NaN	Åland

Первый способ решить эту проблему

Удалить строки с нулевыми значениями

In [17]:
```
sample_incomplete_rows.dropna(subset=['AverageTemperatureUncertainty'])
```

Out[17]:

	dt	AverageTemperature	AverageTemperatureUncertainty	Country

Второй способ

Удалить столбцы, у которых есть нулевые значения(пропуски)

In [18]:
```
sample_incomplete_rows.drop("AverageTemperatureUncertainty", axis=1)
```

Out[18]:

	dt	AverageTemperature	Country
1	1	NaN	Åland
2	2	NaN	Åland
3	3	NaN	Åland
4	4	NaN	Åland
9	9	NaN	Åland

Третий способ

Заменить нулевые (пустые) значения средним/медианой/самой частой величиной

In [19]:
```
mean_ = gt['AverageTemperatureUncertainty'].mean()
sample_incomplete_rows['AverageTemperatureUncertainty'].fillna(mean_, inplace=True)
sample_incomplete_rows
```

Out[19]:

	dt	AverageTemperature	AverageTemperatureUncertainty	Country
1	1	NaN	1.019057	Åland
2	2	NaN	1.019057	Åland
3	3	NaN	1.019057	Åland
4	4	NaN	1.019057	Åland
9	9	NaN	1.019057	Åland

In [20]:
```
median = gt['AverageTemperatureUncertainty'].median()
sample_incomplete_rows['AverageTemperatureUncertainty'].fillna(median, inplace=True)
sample_incomplete_rows
```

Out[20]:

	dt	AverageTemperature	AverageTemperatureUncertainty	Country
1	1	NaN	1.019057	Åland
2	2	NaN	1.019057	Åland
3	3	NaN	1.019057	Åland
4	4	NaN	1.019057	Åland
9	9	NaN	1.019057	Åland

Масштабирование данных

In [25]:
```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
gt_1 = gt.copy()
gt_1.drop(['Country'], axis=1, inplace=True)
gt_1 = scaler.fit_transform(gt_1)
df = pd.DataFrame(gt_1)
df
```

Out[25]:

	0	1	2
0	-2.509901	-1.169382	1.060747
1	-2.508637	NaN	NaN
2	-2.507373	NaN	NaN
3	-2.506108	NaN	NaN
4	-2.504844	NaN	NaN
...	...	...	...
577457	1.578606	0.170317	0.002449
577458	1.579870	0.038310	-0.454317
577459	1.581134	-0.017652	-0.470957
577460	1.582398	0.234221	-0.251310
577461	1.583663	NaN	NaN

577462 rows × 3 columns

Гит-репозиторий: https://github.com/Ilyagu/TMO