Лабораторная работа №2. Обработка признаков

Задание:

Выбрать набор данных (датасет), содержащий категориальные и числовые 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.). Для выбранного датасета (датасетов) на основе материалов лекций решить следующие задачи:

1. устранение пропусков в данных;
2. кодирование категориальных признаков;
3. нормализацию числовых признаков.

In [56]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.impute import KNNImputer
from sklearn.preprocessing import MinMaxScaler
from category_encoders.count import CountEncoder as ce_CountEncoder
```

In [57]:

```
data_loaded = pd.read_csv('world-happiness-report-2021.csv', sep=',')
data = data_loaded.rename(columns={
})
data.head()
```

Out[57]:

	Country name	Regional indicator	Ladder score	Standard error of ladder score	upperwhisker	lowerwhisker	Logged GDP per capita	Social support	He ex
0	Finland	Western Europe	7.842	0.032	7.904	7.780	10.775	0.954	
1	Denmark	Western Europe	NaN	0.035	7.687	7.552	10.933	0.954	
2	Switzerland	Western Europe	7.571	0.036	7.643	7.500	11.117	0.942	
3	Iceland	Western Europe	7.554	0.059	7.670	7.438	10.878	0.983	
4	Netherlands	Western Europe	7.464	0.027	7.518	7.410	10.932	0.942	

In [58]:

```
to_delete = []
data=data.drop(to_delete, axis=1)
data_features = list(zip(
# признаки
[i for i in data.columns],
zip(
 # типы колонок
 [str(i) for i in data.dtypes],
 # проверим есть ли пропущенные значения
 [i for i in data.isnull().sum()]
)))
# Признаки с типом данных и количеством пропусков
data_features
```

Out[58]:

```
[('Country name', ('object', 0)),
 ('Regional indicator', ('object', 0)),
 ('Ladder score', ('float64', 4)),
 ('Standard error of ladder score', ('float64', 0)),
 ('upperwhisker', ('float64', 0)),
 ('lowerwhisker', ('float64', 0)),
 ('Logged GDP per capita', ('float64', 0)),
 ('Social support', ('float64', 0)),
 ('Healthy life expectancy', ('float64', 0)),
 ('Freedom to make life choices', ('float64', 0)),
 ('Generosity', ('float64', 0)),
 ('Perceptions of corruption', ('float64', 0)),
 ('Ladder score in Dystopia', ('float64', 0)),
 ('Explained by: Log GDP per capita', ('float64', 0)),
 ('Explained by: Social support', ('float64', 0)),
 ('Explained by: Healthy life expectancy', ('float64', 0)),
 ('Explained by: Freedom to make life choices', ('float64', 10)),
 ('Explained by: Generosity', ('float64', 0)),
 ('Explained by: Perceptions of corruption', ('float64', 0)),
 ('Dystopia + residual', ('float64', 0))]
```

Устранение пропусков данных

In [59]:

```
cols_with_na = [c for c in data.columns if data[c].isnull().sum() > 0]
[(c, data[c].isnull().sum(), "%.3f" % data[c].isnull().mean()) for c in cols_with_na]
```

Out[59]:

```
[('Ladder score', 4, '0.027'),
 ('Explained by: Freedom to make life choices', 10, '0.067')]
```

In [62]:

```
cols_to_delete = []
data_droped = data.drop(cols_to_delete, axis=1)
data_droped
```

Out[62]:

	Country name	Regional indicator	Ladder score	Standard error of ladder score	upperwhisker	lowerwhisker	Logged GDP per capita	Social support	Healthy life expectancy	Fr t
0	Finland	Western Europe	7.842	0.032	7.904	7.780	10.775	0.954	72.000	
1	Denmark	Western Europe	NaN	0.035	7.687	7.552	10.933	0.954	72.700	
2	Switzerland	Western Europe	7.571	0.036	7.643	7.500	11.117	0.942	74.400	
3	Iceland	Western Europe	7.554	0.059	7.670	7.438	10.878	0.983	73.000	
4	Netherlands	Western	7.464	0.027	7.518	7.410	10.932	0.942	72.400	

In [63]:

```
data_to_impute = data_droped[['Ladder score','Explained by: Freedom to make life choices']]
data_to_impute.isnull().sum()
```

Out[63]:

```
Ladder score                                4
Explained by: Freedom to make life choices    10
dtype: int64
```

In [64]:

```
knnimputer = KNNImputer(
    n_neighbors=5,
    weights='distance',
    metric='nan_euclidean',
    add_indicator=False,
)
array_imputed = knnimputer.fit_transform(data_to_impute)
data_imputed = data_droped.merge(pd.DataFrame(array_imputed, columns=data_to_impute.colum
data_imputed.isnull().sum()
```

Out[64]:

```
Country name                                  0
Regional indicator                            0
Ladder score                                  0
Standard error of ladder score                0
upperwhisker                                  0
lowerwhisker                                  0
Logged GDP per capita                         0
Social support                                0
Healthy life expectancy                       0
Freedom to make life choices                  0
Generosity                                    0
Perceptions of corruption                     0
Ladder score in Dystopia                      0
Explained by: Log GDP per capita              0
Explained by: Social support                  0
Explained by: Healthy life expectancy         0
Explained by: Freedom to make life choices    0
Explained by: Generosity                      0
Explained by: Perceptions of corruption       0
Dystopia + residual                           0
dtype: int64
```

Кодирование категориальных признаков

In [65]:

```
print(data_imputed["Regional indicator"].unique(), '\n')

encoder_test = ce_CountEncoder()
col_encoded_test = encoder_test.fit_transform(data_imputed[['Regional indicator']])
print(col_encoded_test["Regional indicator"].unique(), '\n')

encoder = ce_CountEncoder(normalize=True)
col_encoded = encoder.fit_transform(data_imputed[['Regional indicator']])
print(col_encoded["Regional indicator"].unique())
```

```
['Western Europe' 'North America and ANZ' 'Middle East and North Africa'
 'Latin America and Caribbean' 'Central and Eastern Europe' 'East Asia'
 'Southeast Asia' 'Commonwealth of Independent States'
 'Sub-Saharan Africa' 'South Asia']

[21  4 17 14  5  9 10 34  7]

[0.15217391 0.02898551 0.12318841 0.10144928 0.03623188 0.06521739
 0.07246377 0.24637681 0.05072464]
```

In [66]:

```
data_encoded = data_imputed.copy()
data_encoded['Regional indicator'] = col_encoded['Regional indicator']
data_encoded.head()
```

Out[66]:

	Country name	Regional indicator	Ladder score	Standard error of ladder score	upperwhisker	lowerwhisker	Logged GDP per capita	Social support	He ex
0	Finland	0.152174	7.842	0.032	7.904	7.780	10.775	0.954	
1	Switzerland	0.152174	7.571	0.036	7.643	7.500	11.117	0.942	
2	Iceland	0.152174	7.554	0.059	7.670	7.438	10.878	0.983	
3	Netherlands	0.152174	7.464	0.027	7.518	7.410	10.932	0.942	
4	Norway	0.152174	7.392	0.035	7.462	7.323	11.053	0.954	

Нормализация числовых признаков

In [67]:

```
cols_to_scale = ["Ladder score", "Standard error of ladder score", "upperwhisker", "lower

MMScaler = MinMaxScaler()
data_scaled = data_encoded.copy()
for col in cols_to_scale:
    data_scaled[col] = MMScaler.fit_transform(data_encoded[[col]])
data_scaled.describe()
```

Out[67]:

	Regional indicator	Ladder score	Standard error of ladder score	upperwhisker	lowerwhisker	Logged GDP per capita	Soci suppo
count	138.000000	138.000000	138.000000	138.000000	138.000000	138.000000	138.0000(
mean	0.138731	0.559743	0.221434	0.568775	0.550828	0.558635	0.6724!
std	0.070084	0.205781	0.149712	0.202502	0.209259	0.232779	0.2213'
min	0.028986	0.000000	0.000000	0.000000	0.000000	0.000000	0.0000(
25%	0.079710	0.428652	0.117347	0.446260	0.415166	0.379838	0.5461!
50%	0.123188	0.554334	0.193878	0.556424	0.551679	0.586193	0.7067'
75%	0.152174	0.712916	0.299320	0.719292	0.703902	0.755387	0.8461!
max	0.246377	1.000000	1.000000	1.000000	1.000000	1.000000	1.0000(

Итоговые данные

In [68]:

```
data_scaled
```

Out[68]:

	Country name	Regional indicator	Ladder score	Standard error of ladder score	upperwhisker	lowerwhisker	Logged GDP per capita	Soci suppo
0	Finland	0.152174	1.000000	0.040816	1.000000	1.000000	0.826018	0.9442
1	Switzerland	0.152174	0.949051	0.068027	0.950829	0.947477	0.894254	0.9211
2	Iceland	0.152174	0.945854	0.224490	0.955916	0.935847	0.846568	1.0000
3	Netherlands	0.152174	0.928934	0.006803	0.927280	0.930595	0.857342	0.9211
4	Norway	0.152174	0.915398	0.061224	0.916729	0.914275	0.881484	0.9442
...	...	...	...	...	...	...	...	
133	Lesotho	0.246377	0.185937	0.639456	0.217031	0.155130	0.257582	0.6230
134	Botswana	0.246377	0.177477	0.326531	0.191221	0.163759	0.627893	0.6173
135	Rwanda	0.246377	0.167701	0.285714	0.179352	0.156256	0.207702	0.1711
136	Zimbabwe	0.246377	0.116939	0.217687	0.124906	0.108985	0.260974	0.5519
137	Afghanistan	0.050725	0.000000	0.081633	0.000000	0.000000	0.211492	0.0000

138 rows × 20 columns

In [ ]:

```

```