Рубежный контроль №1

Садыков Марк, ИУ5-63Б

Тема: Технологии разведочного анализа и обработки данных.

Задача №2.

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?

Наборы данных: [https://www.kaggle.com/rhuebner/human-resources-data-set (https://www.kaggle.com/rhuebner/human-resources-data-set)](https://www.kaggle.com/rhuebner/human-resources-data-set)

In [5]:

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

In [6]:

```
data = pd.read_csv('HRDataset_v13.csv', sep=",")
```

In [7]:

```
data.head()
```

Out[7]:

	Employee_Name	EmpID	MarriedID	MaritalStatusID	GenderID	EmpStatusID	DeptII
0	Brown, Mia	1.103024e+09	1.0	1.0	0.0	1.0	1.(
1	LaRotonda, William	1.106027e+09	0.0	2.0	1.0	1.0	1.(
2	Steans, Tyrone	1.302053e+09	0.0	0.0	1.0	1.0	1.(
3	Howard, Estelle	1.211051e+09	1.0	1.0	0.0	1.0	1.(
4	Singh, Nan	1.307060e+09	0.0	0.0	0.0	1.0	1.(

5 rows × 35 columns

In [8]:

```
data.shape
```

Out[8]:

```
(401, 35)
```

In [9]:

```
data.dtypes
```

Out[9]:

```
Employee_Name                  object
EmpID                         float64
MarriedID                     float64
MaritalStatusID               float64
GenderID                      float64
EmpStatusID                   float64
DeptID                        float64
PerfScoreID                   float64
FromDiversityJobFairID        float64
PayRate                       float64
Termd                         float64
PositionID                    float64
Position                       object
State                          object
Zip                           float64
DOB                            object
Sex                            object
MaritalDesc                    object
CitizenDesc                    object
HispanicLatino                 object
RaceDesc                       object
DateofHire                     object
DateofTermination              object
TermReason                     object
EmploymentStatus               object
Department                     object
ManagerName                    object
ManagerID                     float64
RecruitmentSource              object
PerformanceScore               object
EngagementSurvey              float64
EmpSatisfaction               float64
SpecialProjectsCount          float64
LastPerformanceReview_Date     object
DaysLateLast30                float64
dtype: object
```

In [10]:

```
data.isnull().sum()
```

Out[10]:

```
Employee_Name                  91
EmpID                          91
MarriedID                      91
MaritalStatusID                91
GenderID                       91
EmpStatusID                    91
DeptID                         91
PerfScoreID                    91
FromDiversityJobFairID         91
PayRate                        91
Termd                          91
PositionID                     91
Position                       91
State                          91
Zip                            91
DOB                            91
Sex                            91
MaritalDesc                    91
CitizenDesc                    91
HispanicLatino                 91
RaceDesc                       91
DateofHire                     91
DateofTermination             298
TermReason                     92
EmploymentStatus               91
Department                     91
ManagerName                    91
ManagerID                      99
RecruitmentSource              91
PerformanceScore               91
EngagementSurvey               91
EmpSatisfaction                91
SpecialProjectsCount           91
LastPerformanceReview_Date    194
DaysLateLast30                194
dtype: int64
```

Заполним пропуски столбца категориального признака "Пол".

In [11]:

```
data['GenderID']
```

Out[11]:

```
0      0.0
1      1.0
2      1.0
3      0.0
4      0.0
5      0.0
6      0.0
7      0.0
8      0.0
9      1.0
10     1.0
11     0.0
12     1.0
13     1.0
14     1.0
15     1.0
16     1.0
17     0.0
18     0.0
19     1.0
20     1.0
21     0.0
22     1.0
23     0.0
24     1.0
25     1.0
26     1.0
27     1.0
28     0.0
29     0.0
      ...
371    NaN
372    NaN
373    NaN
374    NaN
375    NaN
376    NaN
377    NaN
378    NaN
379    NaN
380    NaN
381    NaN
382    NaN
383    NaN
384    NaN
385    NaN
386    NaN
387    NaN
388    NaN
389    NaN
390    NaN
391    NaN
392    NaN
393    NaN
394    NaN
395    NaN
396    NaN
397    NaN
398    NaN
```

```
399    NaN
400    NaN
Name: GenderID, Length: 401, dtype: float64
```

In [12]:

```
#проверка
data['GenderID'].unique()
```

Out[12]:

```
array([ 0.,  1., nan])
```

In [13]:

```
from sklearn.impute import SimpleImputer
```

In [14]:

```
data[data['GenderID'].isnull()].shape[0]
```

Out[14]:

```
91
```

In [15]:

```
imp = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
data['GenderID'] = imp.fit_transform(data[['GenderID']])
```

In [16]:

```
#проверка столбца на пустые значения
data[data['GenderID'].isnull()].shape[0]
```

Out[16]:

```
0
```

In [17]:

```
#проверка
data['GenderID'].unique()
```

Out[17]:

```
array([0., 1.])
```

Заполним пропуски столбца количественного признака "Ставка оплаты".

In [23]:

```
from sklearn.impute import MissingIndicator
```

In [24]:

```
data['PayRate']
```

Out[24]:

```
0      28.50
1      23.00
2      29.00
3      21.50
4      16.56
5      20.50
6      55.00
7      55.00
8      55.00
9      56.00
10     55.50
11     55.00
12     55.50
13     55.00
14     55.00
15     55.00
16     55.00
17     54.00
18     55.00
19     56.00
20     55.00
21     55.00
22     55.00
23     55.00
24     56.00
25     55.00
26     56.00
27     55.00
28     55.00
29     57.00
       ...
371      NaN
372      NaN
373      NaN
374      NaN
375      NaN
376      NaN
377      NaN
378      NaN
379      NaN
380      NaN
381      NaN
382      NaN
383      NaN
384      NaN
385      NaN
386      NaN
387      NaN
388      NaN
389      NaN
390      NaN
391      NaN
392      NaN
393      NaN
394      NaN
395      NaN
396      NaN
397      NaN
398      NaN
```

```
399      NaN
400      NaN
Name: PayRate, Length: 401, dtype: float64
```

In [25]:

```
empty_index = data[data['PayRate'].isnull()].index
```

In [26]:

```
data[data.index.isin(empty_index)]['PayRate']
```

Out[26]:

```
310    NaN
311    NaN
312    NaN
313    NaN
314    NaN
315    NaN
316    NaN
317    NaN
318    NaN
319    NaN
320    NaN
321    NaN
322    NaN
323    NaN
324    NaN
325    NaN
326    NaN
327    NaN
328    NaN
329    NaN
330    NaN
331    NaN
332    NaN
333    NaN
334    NaN
335    NaN
336    NaN
337    NaN
338    NaN
339    NaN
        ..
371    NaN
372    NaN
373    NaN
374    NaN
375    NaN
376    NaN
377    NaN
378    NaN
379    NaN
380    NaN
381    NaN
382    NaN
383    NaN
384    NaN
385    NaN
386    NaN
387    NaN
388    NaN
389    NaN
390    NaN
391    NaN
392    NaN
393    NaN
394    NaN
395    NaN
396    NaN
397    NaN
398    NaN
```

```
399   NaN
400   NaN
Name: PayRate, Length: 91, dtype: float64
```

In [34]:

```
temp_data = data[['PayRate']]
indicator = MissingIndicator()
mask_missing_values_only = indicator.fit_transform(data[['PayRate']])
imp_num = SimpleImputer(strategy = 'median')
data[['PayRate']] = imp_num.fit_transform(data[['PayRate']])
```

In [35]:

```
#проверка столбца на пустые значения
data[data['PayRate'].isnull()].shape[0]
```

Out[35]:

```
0
```

In [37]:

```
data['PayRate']
```

Out[37]:

```
0      28.50
1      23.00
2      29.00
3      21.50
4      16.56
5      20.50
6      55.00
7      55.00
8      55.00
9      56.00
10     55.50
11     55.00
12     55.50
13     55.00
14     55.00
15     55.00
16     55.00
17     54.00
18     55.00
19     56.00
20     55.00
21     55.00
22     55.00
23     55.00
24     56.00
25     55.00
26     56.00
27     55.00
28     55.00
29     57.00
       ...
371    24.00
372    24.00
373    24.00
374    24.00
375    24.00
376    24.00
377    24.00
378    24.00
379    24.00
380    24.00
381    24.00
382    24.00
383    24.00
384    24.00
385    24.00
386    24.00
387    24.00
388    24.00
389    24.00
390    24.00
391    24.00
392    24.00
393    24.00
394    24.00
395    24.00
396    24.00
397    24.00
398    24.00
```

```
399    24.00
400    24.00
Name: PayRate, Length: 401, dtype: float64
```

In [39]:

```
data.corr()
```

Out[39]:

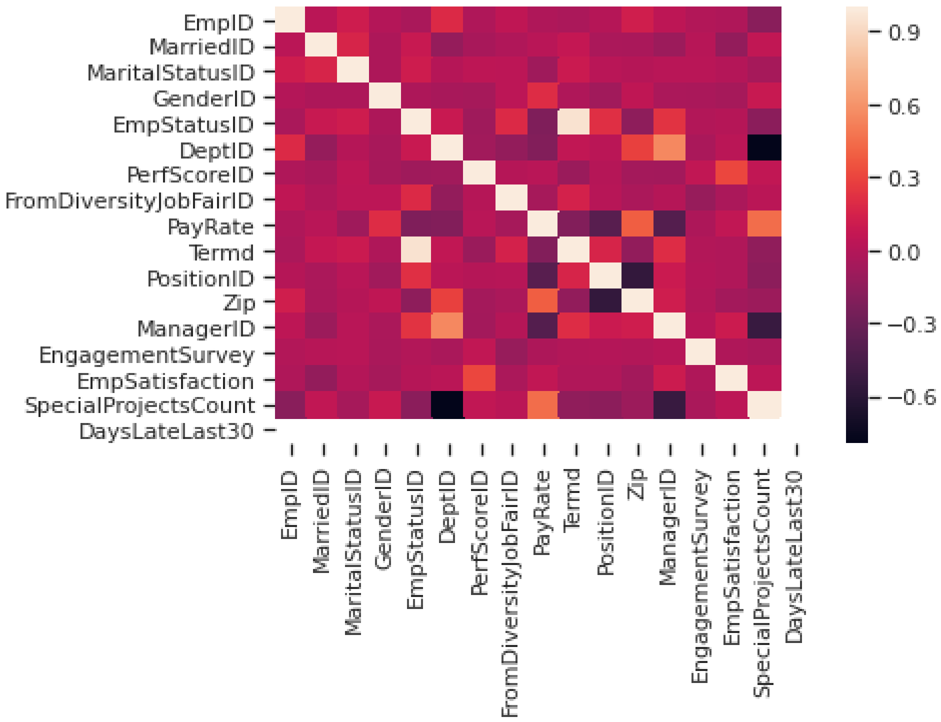
	EmpID	MarriedID	MaritalStatusID	GenderID	EmpStatusID	Dep
EmpID	1.000000	0.034146	0.112300	0.000119	-0.038664	0.192
MarriedID	0.034146	1.000000	0.163655	-0.023593	0.089000	-0.125
MaritalStatusID	0.112300	0.163655	1.000000	-0.025479	0.115255	0.011
GenderID	0.000119	-0.023593	-0.025479	1.000000	-0.024618	-0.046
EmpStatusID	-0.038664	0.089000	0.115255	-0.024618	1.000000	0.092
DeptID	0.192228	-0.125659	0.011966	-0.046189	0.092266	1.000
PerfScoreID	-0.019210	-0.045959	0.047773	-0.054915	-0.081250	-0.072
FromDiversityJobFairID	0.049055	-0.011468	0.041335	0.034872	0.188436	-0.129
PayRate	-0.020310	0.026342	-0.082459	0.206868	-0.214835	-0.202
Termd	-0.035483	0.071844	0.098774	-0.016471	0.955596	0.060
PositionID	0.007435	-0.028783	0.021703	-0.075992	0.222350	0.028
Zip	0.130735	-0.040212	0.010792	0.051408	-0.151348	0.291
ManagerID	0.045432	-0.092960	0.023278	-0.031737	0.233673	0.553
EngagementSurvey	-0.005720	0.019149	0.021298	-0.037021	-0.002734	-0.036
EmpSatisfaction	-0.017726	-0.126980	0.001990	-0.053138	0.010866	0.031
SpecialProjectsCount	-0.171329	0.056748	-0.051893	0.089131	-0.163831	-0.791
DaysLateLast30	NaN	NaN	NaN	NaN	NaN	N

In [38]:

```
sns.heatmap(data.corr())
```

Out[38]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc304747810>
```



In [40]:

```
data['FromDiversityJobFairID'].unique()
```

Out[40]:

```
array([ 1.,  0., nan])
```

В качестве целевого признака подходит FromDiversityJobFairID. Все признаки слабо коррелируют с целевым признаком, поэтому они не подходят для дальнейшего построения моделей машинного обучения.