Рубежный контроль №1

Савельев Алексей ИУ5-64б Вариант 2-6

Используемый набор данных:

Контекст

Данные по управлению персоналом могут быть труднодоступными, а специалисты по персоналу, как правило, отстают в отношении компетенции в области аналитики и визуализации данных. Таким образом, доктор Мы с Карлой Паталано поставили перед собой целью создать свой собственный набор данных, связанный с персоналом, который используется на одном из наших курсов MSHRM под названием HR Metrics and Analytics, в Колледже бизнеса Новой Англии. Мы сами создали этот набор данных. Мы используем набор данных, чтобы научить студентов отдела кадров использовать и анализировать данные в Tableau Desktop - инструменте визуализации данных, который легко освоить.

Обратите внимание, что этот набор данных не идеален. По замыслу есть некоторые проблемы. Он в первую очередь разработан как обучающий набор данных - научить кадровых специалистов работе с данными и аналитикой.

Содержание

Мы сократили сложность набора данных до одного файла данных (v14). CSV вращается вокруг вымышленной компании, и основной набор данных содержит имена, DOB, возраст, пол, семейное положение, дату найма, причины увольнения, отдел, независимо от того, активны они или уволены, должность, ставку заработной платы, имя менеджера и оценку работы.

In [69]:
```
# Импортируем библиотеки
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from random import randint as ri
%matplotlib inline
sns.set(style="ticks")
```

In [70]:
```
# читаем данные набора
data = pd.read_csv('JupyterNotebooks/data/RK1_dataset.csv', sep=",")
```

In [71]:
```
data.shape
```

Out[71]:
```
(311, 36)
```

In [72]:
```
# суммы пропусков по столбцам
data.isnull().sum()
```

Out[72]:
```
Employee_Name                  0
EmpID                          0
```

```
MarriedID                     0
MaritalStatusID               0
GenderID                      0
EmpStatusID                   0
DeptID                        0
PerfScoreID                   0
FromDiversityJobFairID        0
Salary                        0
Termd                         0
PositionID                    0
Position                      0
State                         0
Zip                           0
DOB                           0
Sex                           0
MaritalDesc                   0
CitizenDesc                   0
HispanicLatino                0
RaceDesc                      0
DateofHire                    0
DateofTermination           207
TermReason                    0
EmploymentStatus              0
Department                    0
ManagerName                   0
ManagerID                     8
RecruitmentSource             0
PerformanceScore              0
EngagementSurvey              0
EmpSatisfaction               0
SpecialProjectsCount          0
LastPerformanceReview_Date    0
DaysLateLast30                0
Absences                      0
dtype: int64
```

In [73]:
```
# типы данных столбцов
data.dtypes
```

Out[73]:
```
Employee_Name                object
EmpID                         int64
MarriedID                     int64
MaritalStatusID               int64
GenderID                      int64
EmpStatusID                   int64
DeptID                        int64
PerfScoreID                   int64
FromDiversityJobFairID        int64
Salary                        int64
Termd                         int64
PositionID                    int64
Position                     object
State                        object
Zip                           int64
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                int64
SpecialProjectsCount           int64
LastPerformanceReview_Date    object
DaysLateLast30                 int64
Absences                       int64
dtype: object
```

В данном наборе данных есть пропуски, много категориальных в столбце DateofTermination, и не много количественных в столбце ManagerID. Получим эти колонки в списке.

In [74]:
```
# Выберем числовые колонки с пропущенными значениями
# Цикл по колонкам датасета
num_cols = []
cat_cols = []
temp_perc = 0
total_count = data.shape[0]
for col in data.columns:
    # Количество пустых значений
    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)

    if temp_null_count>0 and dt=='object':
        cat_cols.append(col)
        temp_perc = round((temp_null_count / total_count) * 100.0, 2)

    if temp_null_count>0 and (dt=='float64' or dt=='int64' or dt=='objec
        print('Колонка {}. Тип данных {}. Количество пустых значений {},
```

```
Колонка DateofTermination. Тип данных object. Количество пустых значений
207, 66.56%.
Колонка ManagerID. Тип данных float64. Количество пустых значений 8, 2.5
7%.
```

Количественные пропуски

В столбце ManagerID прописанные и пногократно повторяются ID менеджеров в диапазоне от 1 до 39, поэтому пропущенные значения можно записать одним из ID из используемого диапазона, так как это ни на что не повлияет.

In [75]:
```
sett = sorted(data[num_cols[0]].unique())
data[num_cols[0]].fillna(sett[random.randint(0, len(sett))], inplace=Tru
```

In [76]:
```
data.isnull().sum()
```

Out[76]:
```
Employee_Name                0
EmpID                        0
MarriedID                    0
MaritalStatusID              0
GenderID                     0
EmpStatusID                  0
DeptID                       0
PerfScoreID                  0
FromDiversityJobFairID       0
Salary                       0
Termd                        0
PositionID                   0
Position                     0
```

```
State                          0
Zip                            0
DOB                            0
Sex                            0
MaritalDesc                    0
CitizenDesc                    0
HispanicLatino                 0
RaceDesc                       0
DateofHire                     0
DateofTermination            207
TermReason                     0
EmploymentStatus               0
Department                     0
ManagerName                    0
ManagerID                      0
RecruitmentSource              0
PerformanceScore               0
EngagementSurvey               0
EmpSatisfaction                0
SpecialProjectsCount           0
LastPerformanceReview_Date     0
DaysLateLast30                 0
Absences                       0
dtype: int64
```

Таким образом мы исправили пропуски количественных данных

Качественные пропуски

In [77]:
```
data['DateofTermination']
```

Out[77]:
```
0            NaN
1       6/16/2016
2       9/24/2012
3            NaN
4        9/6/2016
           ...
306          NaN
307     9/29/2015
308          NaN
309          NaN
310          NaN
Name: DateofTermination, Length: 311, dtype: object
```

Если количественными пропусками можно было пренебречь, так как было пропущено всего 8 значений, то качественных значений пропущенно слишком много. Применю тот же подход: не буду сильно отдаляться от уже записанных дат (по годам), а значения дней и месяцев буду генерировать наугад.

In [78]:
```
not_null_cat_data = data[data[cat_cols[0]].notna()]
dates = not_null_cat_data[cat_cols[0]]
years = set()
for date in dates:
    date = date.split('/')
    years.add(date[2])
years = sorted(years)
print(years)
```

```
['2010', '2011', '2012', '2013', '2014', '2015', '2016', '2017', '2018']
```

Сгенерируем рандомные даты в эти года след образом:

In [79]:
```
day = ri(1,28)
month = ri(1,12)
```

```
year = ri(int(years[0]), int(years[-1]))
print(day, month, year)
```

```
22 8 2013
```

In [81]:
```
data[cat_cols[0]].fillna(f'{ri(1,28)}/{ri(1,12)}/{ri(int(years[0]), int(
data.isnull().sum()
```

Out[81]:

Employee_Name	0
EmpID	0
MarriedID	0
MaritalStatusID	0
GenderID	0
EmpStatusID	0
DeptID	0
PerfScoreID	0
FromDiversityJobFairID	0
Salary	0
Termd	0
PositionID	0
Position	0
State	0
Zip	0
DOB	0
Sex	0
MaritalDesc	0
CitizenDesc	0
HispanicLatino	0
RaceDesc	0
DateofHire	0
DateofTermination	0
TermReason	0
EmploymentStatus	0
Department	0
ManagerName	0
ManagerID	0
RecruitmentSource	0
PerformanceScore	0
EngagementSurvey	0
EmpSatisfaction	0
SpecialProjectsCount	0
LastPerformanceReview_Date	0
DaysLateLast30	0
Absences	0

dtype: int64

в итоге обработали все пропущенные данные