Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **СИСТЕМЫ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ**

Отчет по лабораторной работе № 3
«Обработка пропусков в данных, кодирование категориальных признаков, масштабирование данных»
по курсу “Технологии машинного обучения”

Исполнитель:
Студент группы ИУ5-63
Желанкина А.С.
_____ 03.03.2018

Москва, 2019

Задание лабораторной работы

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.

Экранные формы с текстом программы и примерами её выполнения

In [22]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
```

In [23]:
```
data = pd.read_csv('train.csv', sep=",")
data.head()
```

Out[23]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S

In [24]:
```
data.dtypes
```

Out[24]:
```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
Sex             object
Age            float64
SibSp            int64
Parch            int64
Ticket          object
Fare           float64
Cabin           object
Embarked        object
dtype: object
```

In [25]:
```
data.shape
```

Out[25]:
```
(891, 12)
```

In [26]:
```
categorical_columns = [c for c in data.columns if data[c].dtype.name == 'object']
```

```
numerical_columns   = [c for c in data.columns if data[c].dtype.name != 'object']
print(categorical_columns)
print(numerical_columns)
```

```
['Name', 'Sex', 'Ticket', 'Cabin', 'Embarked']
['PassengerId', 'Survived', 'Pclass', 'Age', 'SibSp', 'Parch', 'Fare']
```

In [27]:
```
data.isnull().sum()
```

Out[27]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

In [28]:
```
num_cols = []
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:
        num_cols.append(col)
        temp_perc = round((temp_null_count / data.shape[0]) * 100.0, 2)
        print('Колонка {}. Тип данных {}. Количество пустых значений {}, {}%.'.format(col, dt, temp_nu
```

```
Колонка Age. Тип данных float64. Количество пустых значений 177, 19.87%.
Колонка Cabin. Тип данных object. Количество пустых значений 687, 77.1%.
Колонка Embarked. Тип данных object. Количество пустых значений 2, 0.22%.
```

In [29]:
```
data = data.drop(('Cabin'), axis=1)
```

In [30]:
```
data.isnull().sum()
```

Out[30]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Embarked         2
dtype: int64
```

In [31]:
```
data['Age'].describe()
```

Out[31]:
```
count    714.000000
mean      29.699118
std       14.526497
min        0.420000
25%       20.125000
50%       28.000000
75%       38.000000
max       80.000000
Name: Age, dtype: float64
```

In [32]:
```
from sklearn.impute import SimpleImputer
from sklearn.impute import MissingIndicator

data['Age'] = SimpleImputer(strategy='mean').fit_transform(data[['Age']])
```

In [33]:
```
data['Age'].describe()
```

Out[33]:
```
count    891.000000
mean      29.699118
std       13.002015
min        0.420000
25%       22.000000
50%       29.699118
75%       35.000000
max       80.000000
Name: Age, dtype: float64
```

In [34]:
```
data['Embarked'].describe()
```

Out[34]:
```
count     889
unique      3
```

```
top         S
freq      644
Name: Embarked, dtype: object
```

In [35]:
```
data['Embarked'] = SimpleImputer(missing_values=np.nan, strategy='most_frequent').fit_transform(data[[
```

In [36]:
```
data['Embarked'].describe()
```

Out[36]:
```
count     891
unique      3
top         S
freq      646
Name: Embarked, dtype: object
```

In [37]:
```
data.isnull().sum()
```

Out[37]:
```
PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Embarked       0
dtype: int64
```

In [38]:
```
pd.get_dummies(data).head()
```

Out[38]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare	Name_Abbing, Mr. Anthony	Name_Abbott, Mr. Rossmore Edward	Name_Abbott, Mrs. Stanton (Rosa Hunt)	...	Ticket_W./C. 14263
0	1	0	3	22.0	1	0	7.2500	0	0	0	...	0
1	2	1	1	38.0	1	0	71.2833	0	0	0	...	0
2	3	1	3	26.0	0	0	7.9250	0	0	0	...	0
3	4	1	1	35.0	1	0	53.1000	0	0	0	...	0
4	5	0	3	35.0	0	0	8.0500	0	0	0	...	0

In [46]:
```
from sklearn.preprocessing import MinMaxScaler

data = MinMaxScaler().fit_transform(data)
data
```

Out[46]:
```
array([[0.],
       [1.],
       [1.],
       [1.],
       [0.],
       [0.],
       [0.],
```