*Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования*

***«Московский государственный технический университет имени Н.Э. Баумана»***

***(МГТУ им. Н.Э. Баумана)***

**Лабораторная работа №2**
**по дисциплине «Технологии машинного обучения»**

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

Студент: Коростелев Андрей Михайлович

Группа: ИУ5–64Б

Москва, 2021

**Описание задания:**

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.

**Текст программы:**

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
data = pd.read_csv('data/weatherHistory.csv', sep=",")
# Размер датасета (строки, столбцы)
data.shape
# Список колонок с типами данных
data.dtypes
# Количество пропущенных значений
data.isnull().sum()
# Первые 5 строк датасета
data.head()
total_count = data.shape[0]
print('Всего строк: {}'.format(total_count))
# Удаление колонок, содержащих пустые значения
data_new_1 = data.dropna(axis=1, how='any')
(data.shape, data_new_1.shape)
# Удаление строк, содержащих пустые значения
data_new_2 = data.dropna(axis=0, how='any')
(data.shape, data_new_2.shape)
# Выберем числовые колонки с пропущенными значениями
# Цикл по колонкам датасета
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 and (dt=='float64' or dt=='int64'):
        num_cols.append(col)
```

```
        temp_perc = round((temp_null_count / total_count) * 100.0, 2)
        print('Колонка {}. Тип данных {}. Количество пустых значений {},
{}%.'.format(col, dt, temp_null_count, temp_perc))
# Фильтр по колонкам с пропущенными значениями
data_num = data[num_cols]
data_num
# Гистограмма по признакам
for col in data_num:
    plt.hist(data[col], 50)
    plt.xlabel(col)
    plt.show()
from sklearn.impute import SimpleImputer
from sklearn.impute import MissingIndicator
# Более сложная функция, которая позволяет задавать колонку и вид
импьютации
def test_num_impute_col(dataset, column, strategy_param):

    indicator = MissingIndicator()
    mask_missing_values_only = indicator.fit_transform(dataset[[column]])

    imp_num = SimpleImputer(strategy=strategy_param)
    data_num_imp = imp_num.fit_transform(dataset[[column]])

    dataset[column] = data_num_imp
    filled_data = data_num_imp[mask_missing_values_only]

    return column, strategy_param, filled_data.size, filled_data[0],
filled_data[filled_data.size-1]
data[['Temperature (C)']].describe()
test_num_impute_col(data, 'Temperature (C)', 'most_frequent')
data[['Wind Speed (km/h)']].describe()
test_num_impute_col(data, 'Wind Speed (km/h)', 'most_frequent')
data[['Apparent Temperature (C)']].describe()
test_num_impute_col(data, 'Apparent Temperature (C)', 'mean')
data[['Humidity']].describe()
test_num_impute_col(data, 'Humidity', 'mean')
data[['Visibility (km)']].describe()
test_num_impute_col(data, 'Visibility (km)', 'mean')
# Количество пропущенных значений
data.isnull().sum()
# Выберем категориальные колонки с пропущенными значениями
# Цикл по колонкам датасета
cat_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 and (dt=='object'):
        cat_cols.append(col)
        temp_perc = round((temp_null_count / total_count) * 100.0, 2)
        print('Колонка {}. Тип данных {}. Количество пустых значений {},
{}%.'.format(col, dt, temp_null_count, temp_perc))
# Удаление строк, содержащих пустые значения
data_new_3 = data.dropna(axis=0, how='any', subset=['Precip Type'])
(data.shape, data_new_3.shape)
```

```
data[data['Precip Type'].isnull()].shape
# Ищем самые применяемые значения для данного атрибута
data['Precip Type'].value_counts()
imp = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
data['Precip Type'] = imp.fit_transform(data[['Precip Type']])
data['Precip Type'].value_counts()
data.isnull().sum()
data['Precip Type'].unique()
print(f"Количество уникальных записей атрибута 'Precip Type' =
{data['Precip Type'].nunique()} из {data.shape[0]}")
data['Summary'].unique()
print(f"Количество уникальных записей атрибута 'Summary' =
{data['Summary'].nunique()} из {data.shape[0]}")
data['Daily Summary'].unique()
print(f"Количество уникальных записей атрибута 'Daily Summary' =
{data['Daily Summary']
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
le = LabelEncoder()
data['Daily Summary'] = le.fit_transform(data['Daily Summary'])
data.info()
le = LabelEncoder()
data['Summary'] = le.fit_transform(data['Summary'])
data.info()
# кодирование категорий one-hot encoding
pd.get_dummies(data, columns = ['Precip Type'])
from sklearn.preprocessing import MinMaxScaler, StandardScaler,
Normalizer
plt.hist(data['Temperature (C)'], 50)
plt.show()

plt.hist(MinMaxScaler().fit_transform(data[['Temperature (C)']]), 50)
plt.show()
plt.hist(data['Temperature (C)'], 50)
plt.show()

plt.hist(StandardScaler().fit_transform(data[['Temperature (C)']]), 50)
plt.show()
```

**Экранные формы с примерами выполнения программы:**

Находятся в приложенном ноутбуке.