Московский государственный технический университет им. Н.Э. Баумана

Кафедра «Системы обработки информации и управления» – ИУ5

Факультет «Радиотехнический» – РТ5

**Отчёт по рубежному контролю №1 по курсу**

**Технологии машинного обучения**

6

(количество листов)

|  |  |  |
| --- | --- | --- |
| Исполнитель |  |  |
| студент группы РТ5-61б | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Нижаметдинов М. Ш. |
|  |  | “\_\_\_” \_\_\_\_\_\_\_\_\_\_ 2023 г. |
|  |  |  |
| Проверил |  |  |
| Преподаватель кафедры ИУ5 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Гапанюк Ю. Е. |
|  |  | “\_\_\_” \_\_\_\_\_\_\_\_\_\_ 2023 г. |

Москва, 2023 г.

**Задание**

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?

**Набор данных**

<https://www.kaggle.com/lava18/google-play-store-apps>

**Исходный текст проекта**

import numpy as np

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

%matplotlib inline

sns.set(style="ticks")

## Загрузка и первичный анализ данных

data = pd.read\_csv('https://raw.githubusercontent.com/mansurik1/MLT/master/MC%201/Project/data/googleplaystore.csv', sep=",")

# размер набора данных

data.shape

# типы колонок

data.dtypes

# проверим есть ли пропущенные значения

data.isnull().sum()

# Первые 5 строк датасета

data.head()

total\_count = data.shape[0]

print('Всего строк: {}'.format(total\_count))

# Обработка пропусков в данных

## Простые стратегии - удаление или заполнение нулями

Удаление колонок, содержащих пустые значения

`res = data.dropna(axis=1, how='any')`

Удаление строк, содержащих пустые значения

`res = data.dropna(axis=0, how='any')`

\*\*Удаление может производиться для группы строк или колонок.\*\*

# 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)

data.head()

# Заполнение всех пропущенных значений нулями

# В данном случае это некорректно, так как нулями заполняются в том числе категориальные колонки

data\_new\_3 = data.fillna(0)

data\_new\_3.head()

## "Внедрение значений" - импьютация (imputation)

### Обработка пропусков в числовых данных

# 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()

data\_num\_MasVnrArea = data\_num[['Rating']]

data\_num\_MasVnrArea.head()

from sklearn.impute import SimpleImputer

from sklearn.impute import MissingIndicator

# Фильтр для проверки заполнения пустых значений

indicator = MissingIndicator()

mask\_missing\_values\_only = indicator.fit\_transform(data\_num\_MasVnrArea)

mask\_missing\_values\_only

С помощью класса [SimpleImputer](https://scikit-learn.org/stable/modules/generated/sklearn.impute.SimpleImputer.html#sklearn.impute.SimpleImputer) можно проводить импьютацию различными [показателями центра распределения](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BA%D0%B0%D0%B7%D0%B0%D1%82%D0%B5%D0%BB%D0%B8\_%D1%86%D0%B5%D0%BD%D1%82%D1%80%D0%B0\_%D1%80%D0%B0%D1%81%D0%BF%D1%80%D0%B5%D0%B4%D0%B5%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F)

strategies=['mean', 'median', 'most\_frequent']

def test\_num\_impute(strategy\_param):

imp\_num = SimpleImputer(strategy=strategy\_param)

data\_num\_imp = imp\_num.fit\_transform(data\_num\_MasVnrArea)

return data\_num\_imp[mask\_missing\_values\_only]

strategies[0], test\_num\_impute(strategies[0])

strategies[1], test\_num\_impute(strategies[1])

strategies[2], test\_num\_impute(strategies[2])

# Более сложная функция, которая позволяет задавать колонку и вид импьютации

def test\_num\_impute\_col(dataset, column, strategy\_param):

temp\_data = dataset[[column]]

indicator = MissingIndicator()

mask\_missing\_values\_only = indicator.fit\_transform(temp\_data)

imp\_num = SimpleImputer(strategy=strategy\_param)

data\_num\_imp = imp\_num.fit\_transform(temp\_data)

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[['Rating']].describe()

test\_num\_impute\_col(data, 'Rating', strategies[0])

test\_num\_impute\_col(data, 'Rating', strategies[1])

test\_num\_impute\_col(data, 'Rating', strategies[2])

### Обработка пропусков в категориальных данных

# 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))

cat\_temp\_data = data[['Current Ver']]

cat\_temp\_data.head()

cat\_temp\_data['Current Ver'].unique()

cat\_temp\_data[cat\_temp\_data['Current Ver'].isnull()].shape

# Импьютация наиболее частыми значениями (модой)

imp2 = SimpleImputer(missing\_values=np.nan, strategy='most\_frequent')

data\_imp2 = imp2.fit\_transform(cat\_temp\_data)

data\_imp2

# Пустые значения отсутствуют

np.unique(data\_imp2)

# Импьютация константой

imp3 = SimpleImputer(missing\_values=np.nan, strategy='constant', fill\_value='NA')

data\_imp3 = imp3.fit\_transform(cat\_temp\_data)

data\_imp3

np.unique(data\_imp3)

data\_imp3[data\_imp3=='NA'].size

## Подключение другого набора данных (для получения большего количества числовых значений)

from sklearn.datasets import \*

wine = load\_wine()

wine['data'].shape

data1 = pd.DataFrame(data= np.c\_[wine['data'], wine['target']],

columns= wine['feature\_names'] + ['target'])

data1

Чтобы определить, какие признаки я буду использовать для дальнейшего построения моделей машинного обучения и почему, нужно сделать проверку корреляции признаков, которая поможет определить, какие признаки наиболее сильное коррелируют с целевым признаком.

data1.corr()

На основе корреляционной матрицы можно сделать следующие выводы:

- Целевой признак наиболее сильно коррелирует с пропиленом (-0.63), OD280/OD315 разбавленных вин (-0.79), общим кол-во фенолов (-0,71) и флавоноидами (-0,84). Эти признаки обязательно следует оставить в модели.

- Целевой признак отчасти коррелирует с алкоголем (-0.33). Этот признак стоит также оставить в модели.

- Целевой признак слабо коррелирует с золой (-0.05) и магнием (-0.2). Скорее всего эти признаки стоит исключить из модели, возможно они только ухудшат качество модели.