Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Технологии машинного обучения»

Отчет по лабораторной работе №2

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

Выполнил:
студент группы ИУ5-62Б
Перова Анна

Проверил:
преподаватель каф. ИУ5
Гапанюк Ю.Е.

Москва, 2022 г.

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.

Описание датасета

Датасет `SHOP-SALES.xlsx` содержит информацию о продажах магазина.

Параметры покупки:

- *Date* - дата покупки,
- *Hour* - час совершения покупки,
- *Product* - тип купленного товара,
- *Gender* - пол,
- *Color* - цвет товара,
- *Size* - размер товара,
- *Price* - цена товара без учета НДС,
- *Vat* - НДС,
- *Total* - Цена товара с НДС.

Подключение библиотек для анализа данных

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split, GridSearchCV, RandomizedSearchCV
from sklearn.neighbors import KNeighborsRegressor
from sklearn.preprocessing import MinMaxScaler, StandardScaler
from matplotlib import pyplot as plt
import seaborn as sns
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
from warnings import simplefilter
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
##warnings.simplefilter('ignore')
```

Загрузка датасета из файла `shop-sales.csv`

```
data = pd.read_excel('SHOP-SALES.xlsx')
```

Проверка данных

Выведем первые 5 строк датасета для проверки корректного импорта данных:

```
data.head()
```

	Date	HOUR	GENDER	PRODUCT	COLOR	SIZE	SALE CONSULTANT	QUANTITY	PRICE	VAT	TOTAL
0	2020-06-01	10:21:00	MEN	T SHIRT	2160	XS	Mary Taylor	1	36.94	2.96	39.9
1	2020-06-01	10:24:00	WOMEN	SHIRT LONG SLEEVE	2550	40	Kelli Cooley	1	73.98	5.92	79.9
2	2020-06-01	10:26:00	MEN	SHIRT LONG SLEEVE	900	45X	Bradley Saldana	1	73.98	5.92	79.9
3	2020-06-01	10:29:00	WOMEN	KNIT TROUSERS	600	XL	Kelli Cooley	1	46.20	3.70	49.9
4	2020-06-01	10:30:00	MEN	SHIRT LONG SLEEVE	1800	39X	Robert Moran	1	82.32	6.58	88.9

Узнаем размер датасета:

```
print(f'Количество записей: {data.shape[0]}\nКоличество параметров: {data.shape[1]}')
```

```
Количество записей: 9999
Количество параметров: 11
```

Посмотрим краткую информацию обо всех параматрах датасета:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9999 entries, 0 to 9998
Data columns (total 11 columns):
Date                9999 non-null datetime64[ns]
HOUR                9994 non-null object
GENDER              9996 non-null object
PRODUCT             9992 non-null object
COLOR               9999 non-null int64
SIZE                9997 non-null object
SALE CONSULTANT     9999 non-null object
QUANTITY            9999 non-null int64
PRICE               9999 non-null float64
VAT                 9999 non-null float64
TOTAL               9999 non-null float64
dtypes: datetime64[ns](1), float64(3), int64(2), object(5)
memory usage: 859.4+ KB
```

Видим, что в датасете присутствуют данные нескольких типов: вещественные (`float64`),строковые (`object`), дата (`datetime64`). Также узнаём, что в каждом столбце присутствует ровно `9999` значения, следовательно у нас отсутствуют пустые ячейки, что говорит об отсутствии явных пропусков данных в датасете.

Очистка данных

Пропущенные данные

выведем список параметров датасета и для каждого из них найдём количество `null` значений.

```
for column in data.columns:
    print(f'{column}: {data[column].isnull().sum()} null values')
```

```
Date: 0 null values
HOUR: 5 null values
GENDER: 3 null values
PRODUCT: 7 null values
COLOR: 0 null values
SIZE: 2 null values
SALE CONSULTANT: 0 null values
QUANTITY: 0 null values
PRICE: 0 null values
VAT: 0 null values
TOTAL: 0 null values
```

Так как пропущенных значений очень мало относительно количества всех строк, то удалим строки с пропущенными значениями

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

```
data.head()
```

	Date	HOUR	GENDER	PRODUCT	COLOR	SIZE	SALE CONSULTANT	QUANTITY	PRICE	VAT	TOTAL
0	2020-06-01	10:21:00	MEN	T SHIRT	2160	XS	Mary Taylor	1	36.94	2.96	39.9
1	2020-06-01	10:24:00	WOMEN	SHIRT LONG SLEEVE	2550	40	Kelli Cooley	1	73.98	5.92	79.9
2	2020-06-01	10:26:00	MEN	SHIRT LONG SLEEVE	900	45X	Bradley Saldana	1	73.98	5.92	79.9
3	2020-06-01	10:29:00	WOMEN	KNIT TROUSERS	600	XL	Kelli Cooley	1	46.20	3.70	49.9
4	2020-06-01	10:30:00	MEN	SHIRT LONG SLEEVE	1800	39X	Robert Moran	1	82.32	6.58	88.9

```
for column in data.columns:
    print(f'{column}: {data[column].isnull().sum()} null values')
```

```
Date: 0 null values
GENDER: 0 null values
PRODUCT: 0 null values
SIZE: 0 null values
SALE CONSULTANT: 0 null values
PRICE: 0 null values
VAT: 0 null values
TOTAL: 0 null values
houres: 0 null values
minutes: 0 null values
TIME OF DAY: 0 null values
DAY_OF_WEEK: 0 null values
CATEGORY: 0 null values
```

Кодирование категорий товаров и времени суток

```
le = LabelEncoder()
data['TIME OF DAY'] = le.fit_transform(data['TIME OF DAY'])
```

```
data['GENDER'] = le.fit_transform(data[ 'GENDER'])
```

```
data['PRODUCT'] = le.fit_transform(data['PRODUCT'])
```

```
data['SALE CONSULTANT'] = le.fit_transform(data['SALE CONSULTANT'])
```

```
data['DAY_OF_WEEK'] = le.fit_transform(data['DAY_OF_WEEK'])
```

```
data['CATEGORY'] = le.fit_transform(data['CATEGORY'])
```

```
data['TIME OF DAY'].unique()
```

```
array(['Утро', 'День', 'Вечер'], dtype=object)
```

```
data.head()
```

	Date	GENDER	PRODUCT	SIZE	SALE CONSULTANT	PRICE	VAT	TOTAL	houres	minutes	TIME OF DAY	DAY_OF_WEEK	CATEGORY
0	2020-06-01	2	25	XS	4	0.142995	2.96	39.9	10	21	2	1	7
1	2020-06-01	3	19	40	3	0.286378	5.92	79.9	10	24	2	1	7
2	2020-06-01	2	19	45X	0	0.286378	5.92	79.9	10	26	2	1	7
3	2020-06-01	3	11	XL	3	0.178841	3.70				2	1	5
4	2020-06-01	2	19	39X	5	0.318662	6.58	88.9	.0	30	2	1	7

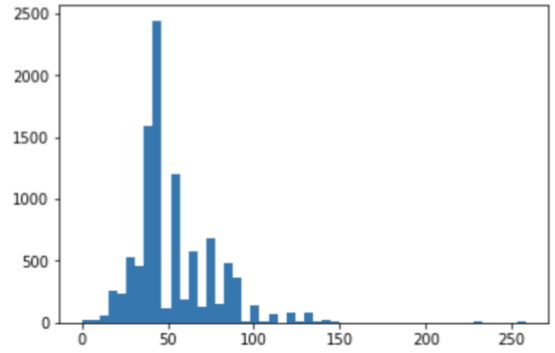
Снимок экрана

Масштабирование данных

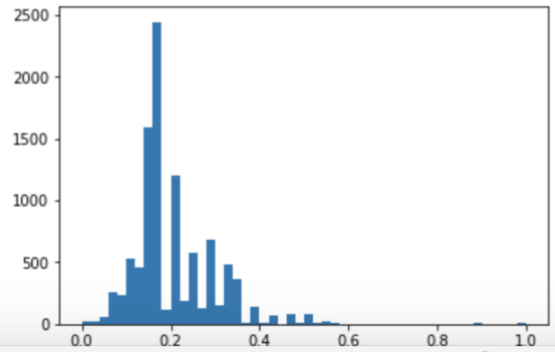
MinMax масштабирование

```
sc1 = MinMaxScaler()
sc1_data = sc1.fit_transform(data[['PRICE']])
```

```
plt.hist(data['PRICE'], 50)
plt.show()
```




```
plt.hist(sc1_data, 50)
plt.show()
```



Масштабирование данных на основе Z-оценки - StandardScaler

```
sc2 = StandardScaler()
sc2_data = sc2.fit_transform(data[['PRICE']])
```

```
plt.hist(sc2_data, 50)
plt.show()
```