Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления

Рубежный контроль 1

Вариант №2

Выполнил:

студент группы ИУ5-25М

Ветошкин Артём

Задание

Каждая задача предполагает использование набора данных. Набор данных выбирается Вами произвольно с учетом следующих условий:

- Вы можете использовать один набор данных для решения всех задач, или решать каждую задачу на своем наборе данных.
- Набор данных должен отличаться от набора данных, который использовался в лекции для решения рассматриваемой задачи.
- Вы можете выбрать произвольный набор данных (например тот, который Вы использовали в лабораторных работах) или создать собственный набор данных (что актуально для некоторых задач, например, для задач удаления псевдоконстантных или повторяющихся признаков).
- Выбранный или созданный Вами набор данных должен удовлетворять условиям поставленной задачи. Например, если решается задача устранения пропусков, то набор данных должен содержать пропуски.

Задача 1

Для набора данных проведите кодирование одного (произвольного) категориального признака с использованием метода "target (mean) encoding".

Задача 2

Для набора данных проведите масштабирование данных для одного (произвольного) числового признака с использованием масштабирования по максимальному значению.

Дополнительно

Для произвольной колонки данных построить парные диаграммы (pairplot).

Импортируем библиотеки

```
!pip install category_encoders
```

```
Requirement already satisfied: category_encoders in /usr/local/lib/python3.10/dist-packages (2.
Requirement already satisfied: numpy>=1.14.0 in /usr/local/lib/python3.10/dist-packages (from c
Requirement already satisfied: scikit-learn>=0.20.0 in /usr/local/lib/python3.10/dist-packages
Requirement already satisfied: scipy>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from ca
Requirement already satisfied: statsmodels>=0.9.0 in /usr/local/lib/python3.10/dist-packages (f
Requirement already satisfied: pandas>=1.0.5 in /usr/local/lib/python3.10/dist-packages (from c
Requirement already satisfied: patsy>=0.5.1 in /usr/local/lib/python3.10/dist-packages (from ca
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.10/dist-package
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pa
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from patsy>=0.5.
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from s
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages
Requirement already satisfied: packaging>=21.3 in /usr/local/lib/python3.10/dist-packages (from
```

```
from category_encoders.target_encoder import TargetEncoder as ce_TargetEncoder
from sklearn.preprocessing import MaxAbsScaler
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
```

Загружаем набор данных

```
df_task_1 = pd.read_csv('car_prices.csv')
```

Задание 1

Набор данных имеет несколько категориальных признаков. В качестве целевого признака будем рассматривать цену продажи автомоболя "sellingprice".

```
df_task_1
```

	year	make	model	trim	body	transmission	vin	state	cond
0	2015	Kia	Sorento	LX	SUV	automatic	5xyktca69fg566472	ca	
1	2015	Kia	Sorento	LX	SUV	automatic	5xyktca69fg561319	ca	
2	2014	BMW	3 Series	328i SULEV	Sedan	automatic	wba3c1c51ek116351	ca	
3	2015	Volvo	S60	T5	Sedan	automatic	yv1612tb4f1310987	ca	
4	2014	BMW	6 Series Gran	650i	Sedan	automatic	wba6b2c57ed129731	ca	

Закодируем катигариальные признаки с помщью метода "target (mean) encoding":

```
TargetEncoder = ce_TargetEncoder()
df_encoded = TargetEncoder.fit_transform(df_task_1[df_task_1.columns.difference(['sellingprice'])],
```

Чтобы изменить содержимое ячейки, дважды нажмите на нее (или выберите "Ввод")

```
df_encoded
```

	body	color	condition	interior	make	mmr	model
0	15905.503680	14740.698566	5.0	15679.512021	11808.672918	20500.0	14643.257923
1	15905.503680	14740.698566	5.0	13348.510794	11808.672918	20800.0	14643.257923
2	11593.969478	13951.472880	45.0	15679.512021	21441.895748	31900.0	16809.633106
3	11593.969478	14740.698566	41.0	15679.512021	11463.952482	27500.0	13766.587164
4	11593.969478	13951.472880	43.0	15679.512021	21441.895748	66000.0	55156.728479
...	...	...	...	...	...	...	...
558832	11593.969478	11781.008599	45.0	15679.512021	11808.672918	35300.0	38150.694603
558833	21342.483584	14740.698566	5.0	15679.512021	25299.936817	30200.0	32911.120043
558834	15905.503680	15509.004749	48.0	15679.512021	21441.895748	29800.0	22717.871649
558835	12299.985824	14740.698566	38.0	15679.512021	11739.015960	15100.0	11421.055765
558836	21849.367776	13951.472880	34.0	11026.952155	14540.469648	29600.0	18832.085020

558837 rows × 15 columns

Задание 2

Возьмём закодированный датасет из задания 1.

```
df_task_2 = df_encoded
df_task_2
```

	body	color	condition	interior	make	mmr	model
0	15905.503680	14740.698566	5.0	15679.512021	11808.672918	20500.0	14643.257923
1	15905.503680	14740.698566	5.0	13348.510794	11808.672918	20800.0	14643.257923
2	11593.969478	13951.472880	45.0	15679.512021	21441.895748	31900.0	16809.633106
3	11593.969478	14740.698566	41.0	15679.512021	11463.952482	27500.0	13766.587164
4	11593.969478	13951.472880	43.0	15679.512021	21441.895748	66000.0	55156.728479
...	...	...	...	...	...	...	...
558832	11593.969478	11781.008599	45.0	15679.512021	11808.672918	35300.0	38150.694603
558833	21342.483584	14740.698566	5.0	15679.512021	25299.936817	30200.0	32911.120043
558834	15905.503680	15509.004749	48.0	15679.512021	21441.895748	29800.0	22717.871649
558835	12299.985824	14740.698566	38.0	15679.512021	11739.015960	15100.0	11421.055765
558836	21849.367776	13951.472880	34.0	11026.952155	14540.469648	29600.0	18832.085020

558837 rows × 15 columns

Прведём маштабирование численных признаков.

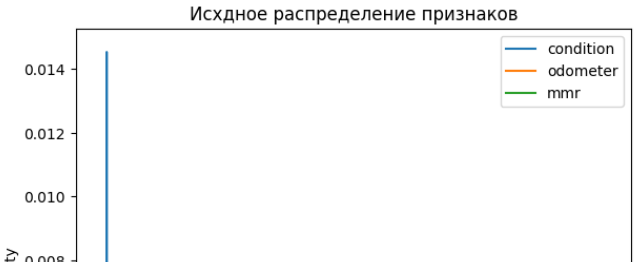
```
mxa = MaxAbsScaler()
data_scaled = mxa.fit_transform(df_task_2)

res = pd.DataFrame(data_scaled, columns=df_task_2.columns)
res
```

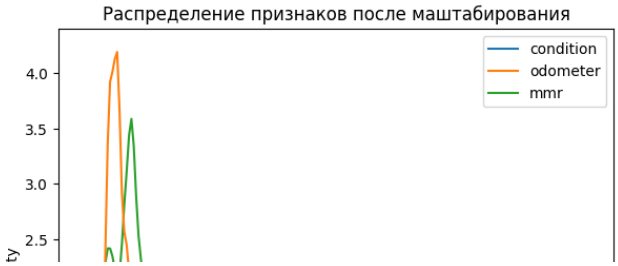
	body	color	condition	interior	make	mmr	model	odometer	saledate
0	0.476261	0.661195	0.102041	0.539817	0.158797	0.112637	0.179354	0.016639	0.284357
1	0.476261	0.661195	0.102041	0.459565	0.158797	0.114286	0.179354	0.009393	0.284357
2	0.347160	0.625794	0.918367	0.539817	0.288339	0.175275	0.205889	0.001331	0.458147
3	0.347160	0.661195	0.836735	0.539817	0.154161	0.151099	0.168617	0.014282	0.458938
4	0.347160	0.625794	0.877551	0.539817	0.288339	0.362637	0.675574	0.002641	0.502879
...	...	...	...	...	...	...	...	...	...
558832	0.347160	0.528438	0.918367	0.539817	0.158797	0.193956	0.467280	0.018255	0.303718
558833	0.639061	0.661195	0.102041	0.539817	0.340220	0.165934	0.403104	0.054393	0.283250
558834	0.476261	0.695657	0.979592	0.539817	0.288339	0.163736	0.278254	0.050561	0.283250
558835	0.368300	0.661195	0.775510	0.539817	0.157860	0.082967	0.139888	0.016658	0.257077
558836	0.654239	0.625794	0.693878	0.379638	0.195532	0.162637	0.230660	0.015008	0.254267

558837 rows × 15 columns

Посмотрим результат маштабирования на признаках "condition", "odometer", "mmr".

```
fig, (ax1, ax2) = plt.subplots(ncols=2, figsize=(15, 6))
ax1.set_title("Исхдное распределение признаков")
sns.kdeplot(data=df_task_2[["condition", "odometer", "mmr"]], ax=ax1)
ax2.set_title("Распределение признаков после маштабирования")
sns.kdeplot(data=res[["condition", "odometer", "mmr"]], ax=ax2)
plt.show()
```





Дополнительное задание

Построим парные графики для той же троице признаков: "condition", "odometer", "mmr".

```
sns.pairplot(df_task_1[["condition", "odometer", "mmr"]])
```

```
<seaborn.axisgrid.PairGrid at 0x7f4bd47c53f0>
```