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

In [1]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import sklearn
import matplotlib.pyplot as plt

%matplotlib inline
sns.set(style="ticks")
```

In [2]:

```
data=pd.read_csv('ToyProducts.csv', sep=",")
```

In [3]:

```
data.shape
```

Out[3]:

```
(10000, 17)
```

In [4]:

```
data.dtypes
```

Out[4]:

```
uniq_id                                        object
product_name                                   object
manufacturer                                   object
price                                          object
number_available_in_stock                      object
number_of_reviews                              object
number_of_answered_questions                  float64
average_review_rating                          object
amazon_category_and_sub_category               object
customers_who_bought_this_item_also_bought     object
description                                    object
product_information                            object
product_description                            object
items_customers_buy_after_viewing_this_item    object
customer_questions_and_answers                 object
customer_reviews                               object
sellers                                        object
dtype: object
```

In [5]:

```
data.isnull().sum()
```

Out[5]:

```
uniq_id                                           0
product_name                                      0
manufacturer                                      7
price                                          1435
number_available_in_stock                      2500
number_of_reviews                                18
number_of_answered_questions                    765
average_review_rating                            18
amazon_category_and_sub_category                690
customers_who_bought_this_item_also_bought     1062
description                                     651
product_information                              58
product_description                             651
items_customers_buy_after_viewing_this_item    3065
customer_questions_and_answers                 9086
customer_reviews                                 21
sellers                                        3082
dtype: int64
```

In [6]:

```
data.head()
```

Out[6]:

	uniq_id	product_name	manufacturer	price	number_available_in_stock	number_of_reviews	number_of_answered_ques
0	eac7efa5dbd3d667f26eb3d3ab504464	Hornby 2014 Catalogue	Hornby	£3.42	5 new	15	
1	b17540ef7e86e461d37f3ae58b7b72ac	FunkyBuys® Large Christmas Holiday Express Fes...	FunkyBuys	£16.99	NaN	2	
2	348f344247b0c1a935b1223072ef9d8a	CLASSIC TOY TRAIN SET TRACK CARRIAGES LIGHT EN...	ccf	£9.99	2 new	17	
3	e12b92dbb8eaee78b22965d2a9bbbd9f	HORNBY Coach R4410A BR Hawksworth Corridor 3rd	Hornby	£39.99	NaN	1	
4	e33a9adeed5f36840ccc227db4682a36	Hornby 00 Gauge 0-4-0 Gildenlow Salt Co. Steam...	Hornby	£32.19	NaN	3	

In [7]:

```
total_count=data.shape[0]
print('Всего строк:{}'.format(total_count))
```

```
Всего строк:10000
```

In [8]:

```
#Обработка пропусков
#Удаление колонок, содержащих пустые значения
data_new1=data.dropna(axis=1, how='any')
(data.shape, data_new1.shape)
```

Out[8]:

```
((10000, 17), (10000, 2))
```

In [9]:

```
#Удаление строк, содержащих пустые значения
data_new2=data.dropna(axis=0, how='any')
(data.shape, data_new2.shape)
```

Out[9]:

```
((10000, 17), (511, 17))
```

In [10]:

```
#Заполнение пропущенных значений нулями
data_new3=data.fillna(0)
data_new3.head()
```

Out[10]:

	uniq_id	product_name	manufacturer	price	number_available_in_stock	number_of_reviews	number_of_answered_ques
0	eac7efa5dbd3d667f26eb3d3ab504464	Hornby 2014 Catalogue	Hornby	£3.42	5 new	15	
1	b17540ef7e86e461d37f3ae58b7b72ac	FunkyBuys® Large Christmas Holiday Express Fes...	FunkyBuys	£16.99	0	2	
2	348f344247b0c1a935b1223072ef9d8a	CLASSIC TOY TRAIN SET TRACK CARRIAGES LIGHT EN...	ccf	£9.99	2 new	17	
3	e12b92dbb8eaee78b22965d2a9bbbd9f	HORNBY Coach R4410A BR Hawksworth Corridor 3rd	Hornby	£39.99	0	1	
4	e33a9adeed5f36840ccc227db4682a36	Hornby 00 Gauge 0-4-0 Gildenlow Salt Co. Steam...	Hornby	£32.19	0	3	

In [11]:

```
# 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))
```

```
Колонка number_of_answered_questions. Тип данных float64. Количество пустых значений 765, 7.65%.
```

In [12]:

```
# Фильтр по колонкам с пропущенными значениями
data_num = data[num_cols]
data_num
```

Out[12]:

	number_of_answered_questions
0	1.0
1	1.0
2	2.0
3	2.0
4	2.0
5	1.0
6	1.0
7	7.0
8	1.0
9	1.0
10	1.0
11	1.0

	number_of_answered_questions
12	1.0
13	1.0
14	1.0
15	1.0
16	1.0
17	1.0
18	1.0
19	1.0
20	1.0
21	1.0
22	1.0
23	1.0
24	1.0
25	1.0
26	1.0
27	1.0
28	1.0
29	1.0
...	...
9970	2.0
9971	2.0
9972	1.0
9973	1.0
9974	1.0
9975	1.0
9976	3.0
9977	3.0
9978	3.0
9979	3.0
9980	3.0
9981	3.0
9982	3.0
9983	3.0
9984	3.0
9985	3.0
9986	3.0
9987	3.0
9988	3.0
9989	3.0
9990	3.0
9991	3.0
9992	3.0
9993	3.0
9994	3.0
9995	3.0
9996	3.0
9997	3.0
9998	3.0
9999	3.0

10000 rows × 1 columns

10000 rows × 7 columns

In [13]:

```
# Фильтр по пустым значениям поля
data[data['number_of_answered_questions'].isnull()]
```

Out[13]:

	uniq_id	product_name	manufacturer	price	number_available_in_stock	number_of_reviews	number_of_answerec
128	aedf496c4f0594f1814f301db907ffad	Kato N Gauge Train Set Case (Kato PlaRail Mode...	Kato	£11.04	39 new	2	
199	159b1371be56ec94a1568647669416b3	Smasha Ballz Ninjaaah	Smasha-Ballz	£15.84	6 new	23	
200	eb85d6369c891422a89137b0008f1818	Moomins - 6.5 Inch Moominpappa Soft Toy - 20056	Moomins	£7.29	2 new	1	
201	5e9618d43e14edff1c4bb5cce3d1d2d2	Classic Cuddly Paddington Bear by Rainbow Desi...	Paddington Bear	£14.60	21 new	41	
202	42fccdc1368987b8b10486d060504d54	Charlie Bears Rainbow Teddy Bear from the Char...	Charlie Bears	£59.90	3 new	1	
203	115a5c70a72db6c6007a00b6aeaf59bd	Yoohoo & Friends - Bush Baby with Pink Love He...	Yoohoo & Friends	£6.95	2 new	1	
204	f02affe0ff40073cf5b5de05c26f9b7c	Monchhichi 45 cm Classic Boy	Monchhichi	£59.99	3 new	1	
205	52334f0d3aac0840e8585d8fba55e01a	Aurora 5 inch Yoohoo and Friends Beaver	Aurora	£5.99	5 new	5	
206	89d45506cb2dc4e010df9c2bab1c9c89	The Puppet Company - Finger Puppets - Silverba...	The Puppet Company	£3.50	NaN	2	
207	32b302e5179bc2cbc99c71c03c5c25f0	Burrows The Meerkat - TY Beanies 11" Classic	Ty	NaN	1 new	1	
208	59e4236135f06417c21596277224eb76	Henry Hugglemonster Summer 15 cm Soft Toy	Henry Hugglemonster	£2.00	19 new	11	
209	e0ad51b34d0dc4a39903998c71e0222a	Melissa & Doug Sack of Snakes	Melissa & Doug	£4.99	6 new	1	
210	dd3abd403d167cff807526bef892fbdc	Animal Babies Crunchy Munchy Baby Panda	Animal Babies	£17.99	10 new	16	
211	833aef8b366a8120582f992ab2b31c9f	Olympic Mascots Union Jack Winning Wenlock	Olympic Mascots	£8.00	2 new	4	
212	54ef4e30ded2184bc8b979a82f6a31f8	Aurora 7-inch Gruffalo Owl	Aurora	£7.94	17 new	36	

	uniq_id	product_name	manufacturer	price	number_available_in_stock	number_of_reviews	number_of_answered
213	f8bb36ac1218670f2e8ac7dbf8da2a17	Suma Collection Plush Soft Toy Meerkat by Rave...	Suma Collection	£12.44	NaN	2	
214	65f6a2412a964c04a262e372cb19ab4c	Angry Birds Rio Soft Plush Toy Jewel Light Blu...	Whitehouse Leisure	£5.50	4 new	1	
215	5db9347343582f2f3d9d046c9b64282c	Thank You Teacher Me to You Bear Cards	Me To You	£0.85	NaN	1	
216	e94f277425bdd57dd3173f1413b52a8f	Creative Halloween Toy soft Plush Pumpkin - 35cm	Creative Toys	NaN	NaN	1	
399	59dcb62489651afd1ee477fbcee8a813	Pintoy Wooden Shut The Box	Pintoy	£7.38	4 new	8	
400	af1251e7f47316afaccb31abf394ebe7	Oblivion Set, 7 Polyhedron Dice D4 D6 D8 D10 D...	Poly Dice Sets	£2.94	2 new	17	
401	a8884bc78ee285e75177c96d1fc455b9	Q-workshop Elven Bag	Q-Workshop	£8.56	7 new	5	
402	402c446d75f54056cd1926d5e7f466f4	50 x 12mm Pearl Plastic dice (Assorted)	Forlorn Hope Games	£4.81	NaN	25	
403	0e243f5d9fe2c3c7f2699f1b93055d9e	Pearl Grey Set (dice0120)	Poly Dice Sets	£3.99	2 new	11	
404	f0b941e532534a84b43bf099f6f14cab	SmartDealsPro LCR Dice Game Left Right Center ...	LCR	NaN	NaN	43	
405	04c8cb55e94dfa2b0e7e7346af761585	Poker Dice in wooden box	Chavet, Chavet	£16.59	NaN	1	
597	ac200dc1631de8351b8bac4c13e91970	Coloured Ice Cream Play Sand Set For Kids - 1 ...	Slammer	£10.95	3 new	24	
598	fbb36cc415de4f799bb260684701c2cf	Boys Girls Children Kids Arts & Crafts Activit...	Creative Fingers	NaN	NaN	4	
599	69d6eeec76b905f67a8db5a3bea51730	Chimp N Zee Head In The Sand Game	Paul Lamond Games	£7.99	5 new	4	
600	68750ff6d9a5808ed0360e48d1204215	Security Fashion Hourglass 10 Minutes Sand Tim...	Generic	£5.21	10 new	8	
...	...	...	...	...	...	...	
9640	7ca26467b36495e3d43a34b122a2cf6a	Mould & Paint - Dinosaur - Boy Boys Child Chil...	Little Sculptures	NaN	NaN	1	
9641	7f28a4244edcd4b55e73bfc7c089ebf3	100 pieces Silver Tone Watermelon Metal Beads ...	k2-accessories Alloy Metal Spacer Beads	£1.20	NaN	3	
9642	6a46c33ae4c7618bdc586c2729987af7	Hama Beads - Large Gift Box	Hama	£15.85	NaN	1	

(Midi Beads)

	uniq_id	product_name	manufacturer	price	number_available_in_stock	number_of_reviews	number_of_answered
9643	650d7031680c5d7ccf1b89a525e389ca	15 pieces Tibetan silver style Alloy connector...	k2-accessories Alloy Metal Spacer Beads	£1.00	NaN	1	
9781	df5e4800d31d445f25178f4905769f27	Polyhedral 7-Die Translucent Dice Set - Blue	Chessex	£8.07	6 new	6	
9782	ca5aff4e5a33e88881fe99f85959fd6e	Big Cherry Giant Dice! One 7cm (70mm) Giant Fo...	BigCherry	£6.25	NaN	2	
9783	30c5e22f878b6b90c13af5892c7bb146	10 x LARGE 19MM BLACK DICE / CRAPS	Dice	£2.50	NaN	1	
9784	9cd42e904b9729924afe8d832430c196	LCR - Left Center Right - Family Dice Game - G...	Left Center Right	£7.95	3 new	12	
9785	ace4eaed8d5dc0b059e8813800324ef7	Sports Dice - Tennis	Paul Lamond Games	£5.99	2 new	1	
9786	63cf6b68658782c63d48496816af5b63	Handmade Wooden Dice Shaker Set - Includes Fiv...	ShalinIndia	£14.00	NaN	1	
9787	2c088f60dcb8876aa5a7c9e058cd0abb	Ultra Pro SLEEVES Pro-Fit Clear C100 Card Game...	Ultra Pro	£4.61	9 new	1	
9788	b9b3c8d206cab575c237f636477ab087	Rory's Story Cubes Moomin	Rory's Story Cubes	£12.00	5 new	3	
9789	a950df0badcdc779a4f172aec0b3a311	The Creativity Hub Rory's Story Cubes Voyages	The Creativity Hub	£8.75	37 new	125	
9790	73e3b7bcf3c3d5829e64e6679335f2b8	Pathfinder Legacy of Fire (7) Dice Set	Q-Workshop	£7.95	3 new	1	
9791	b901f78333c472078401861fe7de40c2	Luxury Wooden Dice Cup With 5 Dice - Jaques Lo...	Jaques of London	NaN	NaN	3	
9792	a1172b930e59ac341cb55cac8e547497	Poly Dice Set, Assorted, 7 Polyhedron Dice Set...	Poly Dice Sets	£1.80	3 new	140	
9793	f2f6271bd523d7b1d8704b6141ea44ad	Dice, Pack of 20 x 16mm Green Pearl Spot Dice D6	Dice World	£4.30	NaN	1	
9794	4ffba4e5416ddca4dfafd47e96e7972e	2 traditional family games Giant Ludo and Gian...	Jesters	£4.39	2 new	15	
9795	a201ee4a6d97f2d6c8749a67072282b9	Galt Toys Ants in your Pants Game	Galt Toys	£6.99	7 new	9	
9796	cae8dfa5cf396d66f9f63d4ed0035157	Pathfinder Skull & Shackles Dice Set	Q-Workshop	£7.95	2 new	4	
9797	20517e8a802ec6b51a6eddaf65bbd6f1	Dragon Dice White/Black (7)	Q-Workshop	£11.28	5 new	9	

	uniq_id	product_name	manufacturer	price	number_available_in_stock	number_of_reviews	number_of_answerec
9798	79d409c9a796b7ee0221995e1f2e9569	SmartDealsPro Set of 5 3.5 x 5" Durable Velvet...	Smartdealspro	NaN	NaN	1	
9799	8eabe8ed6225639710d740a91d753e8a	Yatzy - traditional dice game	Brimtoy Dice Games	£7.99	2 new	7	
9957	6e9b3f173b30f111cb2faa82fc3fba78	Star Wars Return of the Jedi 3.75-inch Desert ...	Star Wars	£10.14	11 new	1	
9958	62fa7a1bc38464b71ab185f149477379	Schleich Superman Vs Darkseid Scenery Pack	Schleich	£10.99	41 new	3	
9959	a8901f6d70002315796fbcab5ab14b0f	Thundercats 10cm Action Figure: Cheetara	Thundercats	£9.50	12 new	6	
9960	8ccacf4e95d4914e4094e0b037501c25	Drogon Egg Game Of Thrones The Noble Collection	Noble Collection	£42.00	8 new	1	
9961	1d7b3f0821b2f66d2443eca1069c4304	Stark Sigil Pendant (costume) Game of Thrones ...	Noble Collection	£10.50	15 new	2	
9962	ef388025f4b2803c49f53fc1c67fa30e	Star Wars 30th Anniversary #14 Biggs Darklight...	Hasbro	£14.99	5 new	2	
9963	b11990bb904deecae1f105450c77ddcc	Orcrist 36" Prop Replica The Hobbit The Noble ...	Noble Collection	£173.41	3 new	19	

765 rows × 17 columns

In [14]:

```
# Запоминаем индексы строк с пустыми значениями
flt_index = data[data['number_of_answered_questions'].isnull()].index
flt_index
```

Out[14]:

```
Int64Index([ 128,  199,  200,  201,  202,  203,  204,  205,  206,  207,
            ...
            9797, 9798, 9799, 9957, 9958, 9959, 9960, 9961, 9962, 9963],
           dtype='int64', length=765)
```

In [15]:

```
# Проверяем что выводятся нужные строки
data[data.index.isin(flt_index)]
```

Out[15]:

	uniq_id	product_name	manufacturer	price	number_available_in_stock	number_of_reviews	number_of_answerec
128	aedf496c4f0594f1814f301db907ffad	Kato N Gauge Train Set Case (Kato PlaRail Mode...	Kato	£11.04	39 new	2	
199	159b1371be56ec94a1568647669416b3	Smasha Ballz Ninjaaah	Smasha-Ballz	£15.84	6 new	23	
200	eb85d6369c891422a89137b0008f1818	Moomins - 6.5 Inch Moominpappa Soft Toy -	Moomins	£7.29	2 new	1	

	uniq_id	product_name	manufacturer	price	number_available_in_stock	number_of_reviews	number_of_answerec
201	5e9618d43e14edff1c4bb5cce3d1d2d2	Classic Cuddly Paddington Bear by Rainbow Desi...	Paddington Bear	£14.60	21 new	41	
202	42fccdc1368987b8b10486d060504d54	Charlie Bears Rainbow Teddy Bear from the Char...	Charlie Bears	£59.90	3 new	1	
203	115a5c70a72db6c6007a00b6aeaf59bd	Yoohoo & Friends - Bush Baby with Pink Love He...	Yoohoo & Friends	£6.95	2 new	1	
204	f02affe0ff40073cf5b5de05c26f9b7c	Monchhichi 45 cm Classic Boy	Monchhichi	£59.99	3 new	1	
205	52334f0d3aac0840e8585d8fba55e01a	Aurora 5 inch Yoohoo and Friends Beaver	Aurora	£5.99	5 new	5	
206	89d45506cb2dc4e010df9c2bab1c9c89	The Puppet Company - Finger Puppets - Silverba...	The Puppet Company	£3.50	NaN	2	
207	32b302e5179bc2cbc99c71c03c5c25f0	Burrows The Meerkat - TY Beanies 11" Classic	Ty	NaN	1 new	1	
208	59e4236135f06417c21596277224eb76	Henry Hugglemonster Summer 15 cm Soft Toy	Henry Hugglemonster	£2.00	19 new	11	
209	e0ad51b34d0dc4a39903998c71e0222a	Melissa & Doug Sack of Snakes	Melissa & Doug	£4.99	6 new	1	
210	dd3abd403d167cff807526bef892fbdc	Animal Babies Crunchy Munchy Baby Panda	Animal Babies	£17.99	10 new	16	
211	833aef8b366a8120582f992ab2b31c9f	Olympic Mascots Union Jack Winning Wenlock	Olympic Mascots	£8.00	2 new	4	
212	54ef4e30ded2184bc8b979a82f6a31f8	Aurora 7-inch Gruffalo Owl	Aurora	£7.94	17 new	36	
213	f8bb36ac1218670f2e8ac7dbf8da2a17	Suma Collection Plush Soft Toy Meerkat by Rave...	Suma Collection	£12.44	NaN	2	
214	65f6a2412a964c04a262e372cb19ab4c	Angry Birds Rio Soft Plush Toy Jewel Light Blu...	Whitehouse Leisure	£5.50	4 new	1	
215	5db9347343582f2f3d9d046c9b64282c	Thank You Teacher Me to You Bear Cards	Me To You	£0.85	NaN	1	
216	e94f277425bdd57dd3173f1413b52a8f	Creative Halloween Toy soft Plush Pumpkin - 35cm	Creative Toys	NaN	NaN	1	
399	59dcb62489651afd1ee477fbcee8a813	Pintoy Wooden Shut The Box	Pintoy	£7.38	4 new	8	
		Oblivion Set, 7					

	uniq_id	product_name	manufacturer	price	number_available_in_stock	number_of_reviews	number_of_answered
400	af1251e7f47316afaccb31abf394e[illegible]	Polyhedron Dice D4 D6 D8 D10 D...	Poly Dice Sets	£2.04	8 new	17	
401	a8884bc78ee285e75177c96d1fc455b9	Q-workshop Elven Bag	Q-Workshop	£8.56	7 new	5	
402	402c446d75f54056cd1926d5e7f466f4	50 x 12mm Pearl Plastic dice (Assorted)	Forlorn Hope Games	£4.81	NaN	25	
403	0e243f5d9fe2c3c7f2699f1b93055d9e	Pearl Grey Set (dice0120)	Poly Dice Sets	£3.99	2 new	11	
404	f0b941e532534a84b43bf099f6f14cab	SmartDealsPro LCR Dice Game Left Right Center ...	LCR	NaN	NaN	43	
405	04c8cb55e94dfa2b0e7e7346af761585	Poker Dice in wooden box	Chavet, Chavet	£16.59	NaN	1	
597	ac200dc1631de8351b8bac4c13e91970	Coloured Ice Cream Play Sand Set For Kids - 1 ...	Slammer	£10.95	3 new	24	
598	fbb36cc415de4f799bb260684701c2cf	Boys Girls Children Kids Arts & Crafts Activit...	Creative Fingers	NaN	NaN	4	
599	69d6eeec76b905f67a8db5a3bea51730	Chimp N Zee Head In The Sand Game	Paul Lamond Games	£7.99	5 new	4	
600	68750ff6d9a5808ed0360e48d1204215	Security Fashion Hourglass 10 Minutes Sand Tim...	Generic	£5.21	10 new	8	
...	...	...	...	...	...	...	
9640	7ca26467b36495e3d43a34b122a2cf6a	Mould & Paint - Dinosaur - Boy Boys Child Chil...	Little Sculptures	NaN	NaN	1	
9641	7f28a4244edcd4b55e73bfc7c089ebf3	100 pieces Silver Tone Watermelon Metal Beads ...	k2-accessories Alloy Metal Spacer Beads	£1.20	NaN	3	
9642	6a46c33ae4c7618bdc586c2729987af7	Hama Beads - Large Gift Box (Midi Beads)	Hama	£15.85	NaN	1	
9643	650d7031680c5d7ccf1b89a525e389ca	15 pieces Tibetan silver style Alloy connector...	k2-accessories Alloy Metal Spacer Beads	£1.00	NaN	1	
9781	df5e4800d31d445f25178f4905769f27	Polyhedral 7-Die Translucent Dice Set - Blue	Chessex	£8.07	6 new	6	
9782	ca5aff4e5a33e88881fe99f85959fd6e	Big Cherry Giant Dice! One 7cm (70mm) Giant Fo...	BigCherry	£6.25	NaN	2	
9783	30c5e22f878b6b90c13af5892c7bb146	10 x LARGE 19MM BLACK DICE / CRAPS	Dice	£2.50	NaN	1	
9784	9cd42e904b9729924afe8d832430c196	LCR - Left Center Right - Family Dice Game - G...	Left Center Right	£7.95	3 new	12	

	uniq_id	product_name	manufacturer	price	number_available_in_stock	number_of_reviews	number_of_answered
9785	ace4eaed8d5dc0b059e8813800324ef7	Sports Dice - Tennis	Paul Lamond Games	£5.99	2 new	1	
9786	63cf6b68658782c63d48496816af5b63	Handmade Wooden Dice Shaker Set - Includes Fiv...	ShalinIndia	£14.00	NaN	1	
9787	2c088f60dcb8876aa5a7c9e058cd0abb	Ultra Pro SLEEVES Pro-Fit Clear C100 Card Game...	Ultra Pro	£4.61	9 new	1	
9788	b9b3c8d206cab575c237f636477ab087	Rory's Story Cubes Moomin	Rory's Story Cubes	£12.00	5 new	3	
9789	a950df0badcdc779a4f172aec0b3a311	The Creativity Hub Rory's Story Cubes Voyages	The Creativity Hub	£8.75	37 new	125	
9790	73e3b7bcf3c3d5829e64e6679335f2b8	Pathfinder Legacy of Fire (7) Dice Set	Q-Workshop	£7.95	3 new	1	
9791	b901f78333c472078401861fe7de40c2	Luxury Wooden Dice Cup With 5 Dice - Jaques Lo...	Jaques of London	NaN	NaN	3	
9792	a1172b930e59ac341cb55cac8e547497	Poly Dice Set, Assorted, 7 Polyhedron Dice Set...	Poly Dice Sets	£1.80	3 new	140	
9793	f2f6271bd523d7b1d8704b6141ea44ad	Dice, Pack of 20 x 16mm Green Pearl Spot Dice D6	Dice World	£4.30	NaN	1	
9794	4ffba4e5416ddca4dfafd47e96e7972e	2 traditional family games Giant Ludo and Gian...	Jesters	£4.39	2 new	15	
9795	a201ee4a6d97f2d6c8749a67072282b9	Galt Toys Ants in your Pants Game	Galt Toys	£6.99	7 new	9	
9796	cae8dfa5cf396d66f9f63d4ed0035157	Pathfinder Skull & Shackles Dice Set	Q-Workshop	£7.95	2 new	4	
9797	20517e8a802ec6b51a6eddaf65bbd6f1	Dragon Dice White/Black (7)	Q-Workshop	£11.28	5 new	9	
9798	79d409c9a796b7ee0221995e1f2e9569	SmartDealsPro Set of 5 3.5 x 5" Durable Velvet...	Smartdealspro	NaN	NaN	1	
9799	8eabe8ed6225639710d740a91d753e8a	Yatzy - traditional dice game	Brimtoy Dice Games	£7.99	2 new	7	
9957	6e9b3f173b30f111cb2faa82fc3fba78	Star Wars Return of the Jedi 3.75-inch Desert ...	Star Wars	£10.14	11 new	1	
9958	62fa7a1bc38464b71ab185f149477379	Schleich Superman Vs Darkseid Scenery Pack	Schleich	£10.99	41 new	3	
9959	a8901f6d70002315796fbcab5ab14b0f	Thundercats 10cm Action Figure: Cheetara	Thundercats	£9.50	12 new	6	
9960	8ccacf4e95d4914e4094e0b037501c25	Drogon Egg Game Of Thrones The	Noble Collection	£42.00	8 new	1	

	uniq_id	product_name	manufacturer	price	number_available_in_stock	number_of_reviews	number_of_answerec
9961	1d7b3f0821b2f66d2443eca1069c4304	Stark Sigil Pendant (costume) Game of Thrones ...	Noble Collection	£10.50	15 new	2	
9962	ef388025f4b2803c49f53fc1c67fa30e	Star Wars 30th Anniversary #14 Biggs Darklight...	Hasbro	£14.99	5 new	2	
9963	b11990bb904deecae1f105450c77ddcc	Orcrist 36" Prop Replica The Hobbit The Noble ...	Noble Collection	£173.41	3 new	19	

765 rows × 17 columns

In [16]:

```
# фильтр по колонке
data_num[data_num.index.isin(flt_index)]['number_of_answered_questions']
```

Out[16]:

```
128    NaN
199    NaN
200    NaN
201    NaN
202    NaN
203    NaN
204    NaN
205    NaN
206    NaN
207    NaN
208    NaN
209    NaN
210    NaN
211    NaN
212    NaN
213    NaN
214    NaN
215    NaN
216    NaN
399    NaN
400    NaN
401    NaN
402    NaN
403    NaN
404    NaN
405    NaN
597    NaN
598    NaN
599    NaN
600    NaN
        ..
9640   NaN
9641   NaN
9642   NaN
9643   NaN
9781   NaN
9782   NaN
9783   NaN
9784   NaN
9785   NaN
9786   NaN
9787   NaN
9788   NaN
9789   NaN
9790   NaN
9791   NaN
9792   NaN
9793   NaN
9794   NaN
9795   NaN
9796   NaN
9797   NaN
9798   NaN
9799   NaN
9957   NaN
```

```
9957   NaN
9958   NaN
9959   NaN
9960   NaN
9961   NaN
9962   NaN
9963   NaN
Name: number_of_answered_questions, Length: 765, dtype: float64
```

In [17]:

```
data_num_number_of_answered_questions = data_num[['number_of_answered_questions']]
data_num_number_of_answered_questions.head()
```

Out[17]:

	number_of_answered_questions
0	1.0
1	1.0
2	2.0
3	2.0
4	2.0

In [18]:

```
from sklearn.impute import SimpleImputer
from sklearn.impute import MissingIndicator
```

In [19]:

```
import sklearn
sklearn.__version__
```

Out[19]:

```
'0.20.3'
```

In [20]:

```
# Фильтр для проверки заполнения пустых значений
indicator = MissingIndicator()
mask_missing_values_only = indicator.fit_transform(data_num_number_of_answered_questions)
mask_missing_values_only
```

Out[20]:

```
array([[False],
       [False],
       [False],
       ...,
       [False],
       [False],
       [False]])
```

In [21]:

```
strategies=['mean', 'median','most_frequent']
```

In [22]:

```
def test_num_impute(strategy_param):
    imp_num = SimpleImputer(strategy=strategy_param)
    data_num_imp = imp_num.fit_transform(data_num_number_of_answered_questions)
    return data_num_imp[mask_missing_values_only]
```

In [23]:

```
strategies[0], test_num_impute(strategies[0])
```

Out[23]:

```
('mean', array([1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564,
       1.83497564, 1.83497564, 1.83497564, 1.83497564, 1.83497564]))
```

In [24]:

```
strategies[1], test_num_impute(strategies[1])
```

Out[24]:

```
('median',
 array([1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
       1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.]))
```

In [25]:

```
strategies[2], test_num_impute(strategies[2])
```

Out[25]:

```
('most_frequent',
 array([1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
        1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.]))
```

In [26]:

```python
# 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]
```

In [27]:

```python
data[['number_of_answered_questions']].describe()
```

Out[27]:

	number_of_answered_questions
count	9235.000000
mean	1.834976
std	2.517268
min	1.000000
25%	1.000000
50%	1.000000
75%	2.000000
max	39.000000

In [28]:

```python
test_num_impute_col(data, 'number_of_answered_questions', strategies[0])
```

Out[28]:

```
('number_of_answered_questions',
 'mean',
 765,
 1.8349756361667569,
 1.8349756361667569)
```

In [29]:

```python
test_num_impute_col(data, 'number_of_answered_questions', strategies[1])
```

Out[29]:

```
('number_of_answered_questions', 'median', 765, 1.0, 1.0)
```

In [30]:

```python
test_num_impute_col(data, 'number_of_answered_questions', strategies[2])
test_num_impute_col(data, 'number_of_answered_questions', strategies[2])
```

Out[30]:

```
('number_of_answered_questions', 'most_frequent', 765, 1.0, 1.0)
```

Обработка пропусков в категориальных данных

In [31]:

```
# 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))
```

```
Колонка manufacturer. Тип данных object. Количество пустых значений 7, 0.07%.
Колонка price. Тип данных object. Количество пустых значений 1435, 14.35%.
Колонка number_available_in_stock. Тип данных object. Количество пустых значений 2500, 25.0%.
Колонка number_of_reviews. Тип данных object. Количество пустых значений 18, 0.18%.
Колонка average_review_rating. Тип данных object. Количество пустых значений 18, 0.18%.
Колонка amazon_category_and_sub_category. Тип данных object. Количество пустых значений 690, 6.9%.
Колонка customers_who_bought_this_item_also_bought. Тип данных object. Количество пустых значений 1062, 10.62%.
Колонка description. Тип данных object. Количество пустых значений 651, 6.51%.
Колонка product_information. Тип данных object. Количество пустых значений 58, 0.58%.
Колонка product_description. Тип данных object. Количество пустых значений 651, 6.51%.
Колонка items_customers_buy_after_viewing_this_item. Тип данных object. Количество пустых значений 3065, 30.65%.
Колонка customer_questions_and_answers. Тип данных object. Количество пустых значений 9086, 90.86%.
Колонка customer_reviews. Тип данных object. Количество пустых значений 21, 0.21%.
Колонка sellers. Тип данных object. Количество пустых значений 3082, 30.82%.
```

In [32]:

```
cat_temp_data = data[['number_of_reviews']]
cat_temp_data.head()
```

Out[32]:

	number_of_reviews
0	15
1	2
2	17
3	1
4	3

In [33]:

```
cat_temp_data['number_of_reviews'].unique()
```

Out[33]:

```
array(['15', '2', '17', '1', '3', '36', '8', '21', '4', '5', '19', '53',
       '6', '38', '10', '7', nan, '9', '13', '18', '97', '28', '12', '67',
       '81', '23', '41', '11', '16', '45', '42', '32', '27', '40', '31',
       '35', '29', '120', '33', '26', '24', '85', '25', '43', '138', '82',
       '46', '14', '72', '22', '106', '76', '420', '160', '39', '30',
       '199', '129', '56', '291', '87', '86', '20', '34', '142', '92',
       '55', '64', '77', '243', '130', '68', '253', '101', '102', '122',
       '73', '118', '145', '381', '802', '299', '59', '518', '158', '44',
       '98', '58', '47', '600', '57', '50', '78', '94', '51', '37', '210',
       '165', '103', '185', '116', '149', '168', '71', '649', '265',
       '355', '79', '100', '83', '61', '48', '70', '65', '49', '54', '66',
       '60', '238', '133', '88', '110', '117', '109', '127', '63', '111',
       '141', '99', '204', '124', '95', '220', '137', '90', '172', '134',
       '115', '80', '91', '125', '119', '131', '146', '517', '512', '151',
       '62', '113', '104', '108', '126', '75', '107', '183', '313', '139',
       '230', '93', '181', '114', '52', '74', '96', '561', '144', '147',
       '132', '136', '105', '535', '156', '155', '516', '112', '200',
       '128', '136', '105', '585', '156', '155', '516', '112', '200',
       '379', '177', '132', '69', '140', '1,040', '337', '164', '193',
       '1,399', '690', '123', '263', '249', '287', '202', '304', '262',
       '241'], dtype=object)
```

In [34]:

```
cat_temp_data[cat_temp_data['number_of_reviews'].isnull()].shape
```

Out[34]:

```
(18, 1)
```

In [35]:

```
# Импьютация наиболее частыми значениями
imp2 = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
data_imp2 = imp2.fit_transform(cat_temp_data)
data_imp2
```

Out[35]:

```
array([['15'],
       ['2'],
       ['17'],
       ...,
       ['1'],
       ['1'],
       ['11']], dtype=object)
```

In [36]:

```
# Пустые значения отсутствуют
np.unique(data_imp2)
```

Out[36]:

```
array(['1', '1,040', '1,399', '10', '100', '101', '102', '103', '104',
       '105', '106', '107', '108', '109', '11', '110', '111', '112',
       '113', '114', '115', '116', '117', '118', '119', '12', '120',
       '122', '123', '124', '125', '126', '127', '128', '129', '13',
       '130', '131', '132', '133', '134', '136', '137', '138', '139',
       '14', '140', '141', '142', '144', '145', '146', '147', '149', '15',
       '151', '155', '156', '158', '16', '160', '164', '165', '168', '17',
       '172', '177', '18', '181', '183', '185', '19', '193', '199', '2',
       '20', '200', '202', '204', '21', '210', '22', '220', '23', '230',
       '238', '24', '241', '243', '249', '25', '253', '26', '262', '263',
       '265', '27', '28', '287', '29', '291', '299', '3', '30', '304',
       '31', '313', '32', '33', '337', '34', '35', '355', '36', '37',
       '379', '38', '381', '39', '4', '40', '41', '42', '420', '43', '44',
       '45', '46', '47', '48', '49', '5', '50', '51', '512', '516', '517',
       '518', '52', '53', '54', '55', '56', '561', '57', '58', '585',
       '59', '6', '60', '600', '61', '62', '63', '64', '649', '65', '66',
       '67', '68', '69', '690', '7', '70', '71', '72', '73', '74', '75',
       '76', '77', '78', '79', '8', '80', '802', '81', '82', '83', '85',
       '86', '87', '88', '9', '90', '91', '92', '93', '94', '95', '96',
       '97', '98', '99'], dtype=object)
```

In [37]:

```
# Импьютация константой
imp3 = SimpleImputer(missing_values=np.nan, strategy='constant', fill_value='!!!')
data_imp3 = imp3.fit_transform(cat_temp_data)
data_imp3
```

Out[37]:

```
array([['15'],
       ['2'],
       ['17'],
       ...,
       ['1'],
       ['1'],
       ['11']], dtype=object)
```

In [38]:

```
np.unique(data_imp3)
```

Out[38]:

```
array(['!!!', '1', '1,040', '1,399', '10', '100', '101', '102', '103',
       '104', '105', '106', '107', '108', '109', '11', '110', '111',
       '112', '113', '114', '115', '116', '117', '118', '119', '12',
       '120', '122', '123', '124', '125', '126', '127', '128', '129',
       '13', '130', '131', '132', '133', '134', '136', '137', '138',
       '139', '14', '140', '141', '142', '144', '145', '146', '147',
       '149', '15', '151', '155', '156', '158', '16', '160', '164', '165',
       '168', '17', '172', '177', '18', '181', '183', '185', '19', '193',
       '199', '2', '20', '200', '202', '204', '21', '210', '22', '220',
       '23', '230', '238', '24', '241', '243', '249', '25', '253', '26',
       '262', '263', '265', '27', '28', '287', '29', '291', '299', '3',
       '30', '304', '31', '313', '32', '33', '337', '34', '35', '355',
       '36', '37', '379', '38', '381', '39', '4', '40', '41', '42', '420',
       '43', '44', '45', '46', '47', '48', '49', '5', '50', '51', '512',
       '516', '517', '518', '52', '53', '54', '55', '56', '561', '57',
       '58', '585', '59', '6', '60', '600', '61', '62', '63', '64', '649',
       '65', '66', '67', '68', '69', '690', '7', '70', '71', '72', '73',
       '74', '75', '76', '77', '78', '79', '8', '80', '802', '81', '82',
       '83', '85', '86', '87', '88', '9', '90', '91', '92', '93', '94',
       '95', '96', '97', '98', '99'], dtype=object)
```

In [39]:

```
data_imp3[data_imp3=='!!!'].size
```

Out[39]:

```
18
```

Преобразование категориальных признаков в числовые

In [40]:

```
cat_enc = pd.DataFrame({'c1':data_imp2.T[0]})
cat_enc
```

Out[40]:

	c1
0	15
1	2
2	17
3	1
4	3
5	2
6	2
7	36
8	1
9	8
10	1
11	1
12	1
13	3
14	1
15	1
16	2
17	2
18	1
19	1
20	1

	c1
21	1
22	1
23	1
24	1
25	21
26	1
27	8
28	4
29	5
...	...
9970	1
9971	2
9972	1
9973	4
9974	1
9975	3
9976	5
9977	1
9978	1
9979	1
9980	5
9981	2
9982	1
9983	2
9984	1
9985	12
9986	2
9987	3
9988	2
9989	1
9990	1
9991	7
9992	2
9993	2
9994	1
9995	3
9996	1
9997	1
9998	1
9999	11

10000 rows × 1 columns

Кодирование категорий целочисленными значениями - label encoding

In [41]:

```
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
```

In [42]:

```
le = LabelEncoder()
cat_enc_le = le.fit_transform(cat_enc['c1'])
```

In [43]:

```
cat_enc['c1'].unique()
```

Out[43]:

```
array(['15', '2', '17', '1', '3', '36', '8', '21', '4', '5', '19', '53',
       '6', '38', '10', '7', '9', '13', '18', '97', '28', '12', '67',
       '81', '23', '41', '11', '16', '45', '42', '32', '27', '40', '31',
       '35', '29', '120', '33', '26', '24', '85', '25', '43', '138', '82',
       '46', '14', '72', '22', '106', '76', '420', '160', '39', '30',
       '199', '129', '56', '291', '87', '86', '20', '34', '142', '92',
       '55', '64', '77', '243', '130', '68', '253', '101', '102', '122',
       '73', '118', '145', '381', '802', '299', '59', '518', '158', '44',
       '98', '58', '47', '600', '57', '50', '78', '94', '51', '37', '210',
       '165', '103', '185', '116', '149', '168', '71', '649', '265',
       '355', '79', '100', '83', '61', '48', '70', '65', '49', '54', '66',
       '60', '238', '133', '88', '110', '117', '109', '127', '63', '111',
       '141', '99', '204', '124', '95', '220', '137', '90', '172', '134',
       '115', '80', '91', '125', '119', '131', '146', '517', '512', '151',
       '62', '113', '104', '108', '126', '75', '107', '183', '313', '139',
       '230', '93', '181', '114', '52', '74', '96', '561', '144', '147',
       '128', '136', '105', '585', '156', '155', '516', '112', '200',
       '379', '177', '132', '69', '140', '1,040', '337', '164', '193',
       '1,399', '690', '123', '263', '249', '287', '202', '304', '262',
       '241'], dtype=object)
```

In [44]:

```
np.unique(cat_enc_le)
```

Out[44]:

```
array([  0,   1,   2,   3,   4,   5,   6,   7,   8,   9,  10,  11,  12,
        13,  14,  15,  16,  17,  18,  19,  20,  21,  22,  23,  24,  25,
        26,  27,  28,  29,  30,  31,  32,  33,  34,  35,  36,  37,  38,
        39,  40,  41,  42,  43,  44,  45,  46,  47,  48,  49,  50,  51,
        52,  53,  54,  55,  56,  57,  58,  59,  60,  61,  62,  63,  64,
        65,  66,  67,  68,  69,  70,  71,  72,  73,  74,  75,  76,  77,
        78,  79,  80,  81,  82,  83,  84,  85,  86,  87,  88,  89,  90,
        91,  92,  93,  94,  95,  96,  97,  98,  99, 100, 101, 102, 103,
       104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116,
       117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129,
       130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142,
       143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155,
       156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168,
       169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181,
       182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193])
```

In [45]:

```
le.inverse_transform([0, 1, 2, 3])
```

Out[45]:

```
array(['1', '1,040', '1,399', '10'], dtype=object)
```

Кодирование категорий наборами бинарных значений - one-hot encoding

In [46]:

```
ohe = OneHotEncoder()
cat_enc_ohe = ohe.fit_transform(cat_enc[['c1']])
```

In [47]:

```
cat_enc.shape
```

Out[47]:

```
(10000, 1)
```

In [48]:

```
cat_enc_ohe.shape
```

Out[48]:

```
(10000, 194)
```

In [49]:

```
cat_enc_ohe
```

Out[49]:

```
<10000x194 sparse matrix of type '<class 'numpy.float64'>'
 with 10000 stored elements in Compressed Sparse Row format>
```

In [50]:

```
cat_enc_ohe.todense()[0:10]
```

Out[50]:

```
matrix([[0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [1., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]])
```

In [51]:

```
cat_enc.head(10)
```

Out[51]:

	c1
0	15
1	2
2	17
3	1
4	3
5	2
6	2
7	36
8	1
9	8

Pandas get_dummies - быстрый вариант one-hot кодирования

In [52]:

```
pd.get_dummies(cat_enc).head()
```

Out[52]:

	c1_1	c1_1,040	c1_1,399	c1_10	c1_100	c1_101	c1_102	c1_103	c1_104	c1_105	...	c1_90	c1_91	c1_92	c1_93	c1_94	c1_95	c1_96	c1_97
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0
3	1	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0

5 rows × 194 columns

In [53]:

```
pd.get_dummies(cat_temp_data, dummy_na=True).head()
```

Out[53]:

	number_of_reviews_1	number_of_reviews_1,040	number_of_reviews_1,399	number_of_reviews_10	number_of_reviews_100	number_of_reviews_101
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	1	0	0	0	0	0
4	0	0	0	0	0	0

5 rows × 195 columns

Масштабирование данных

In [54]:

```
from sklearn.preprocessing import MinMaxScaler, StandardScaler, Normalizer
```

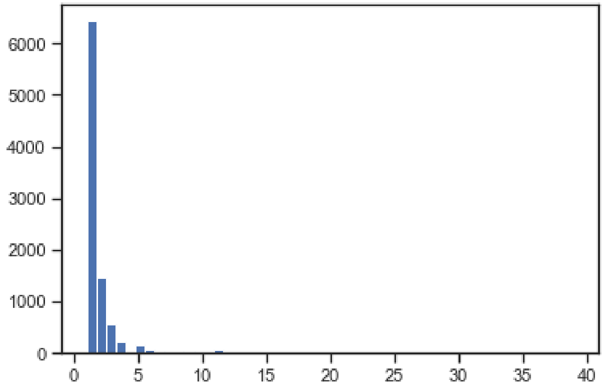
In [55]:

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

In [56]:

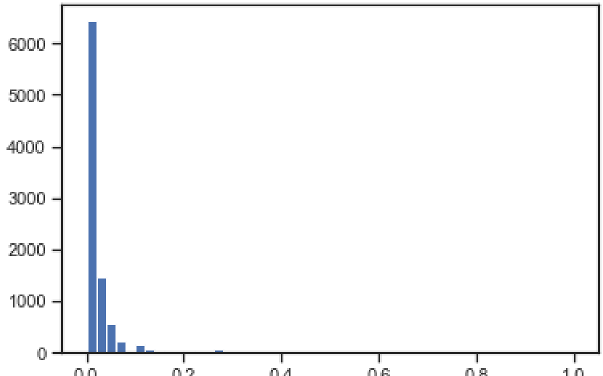
```
plt.hist(data['number_of_answered_questions'], 50)
plt.show()
```

```
C:\Users\Dovlat\Anaconda3\lib\site-packages\numpy\lib\histograms.py:824: RuntimeWarning: invalid value encountered in greater_equal
  keep = (tmp_a >= first_edge)
C:\Users\Dovlat\Anaconda3\lib\site-packages\numpy\lib\histograms.py:825: RuntimeWarning: invalid value encountered in less_equal
  keep &= (tmp_a <= last_edge)
```



In [57]:

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



Масштабирование данных на основе Z-оценки - StandardScaler

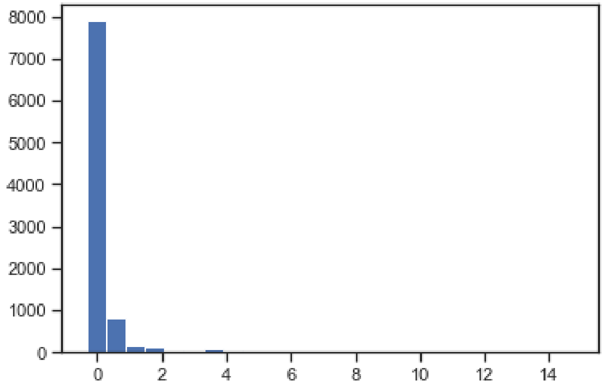
In [58]:

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

In [59]:

```
plt.hist(sc2_data, 25)
plt.show()
```

```
C:\Users\Dovlat\Anaconda3\lib\site-packages\numpy\lib\histograms.py:824: RuntimeWarning: invalid value encountered in greater_equal
  keep = (tmp_a >= first_edge)
C:\Users\Dovlat\Anaconda3\lib\site-packages\numpy\lib\histograms.py:825: RuntimeWarning: invalid value encountered in less_equal
  keep &= (tmp_a <= last_edge)
```


8000
7000
6000
5000
4000
3000
2000
1000
0
0
2
4
6
8
10
12
14