Рубежный контроль №1

Калашников Артем ИУ5-63Б

11 Вариант

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?

Дополнительное требование:

Для студентов групп ИУ5-63Б, ИУ5Ц-83Б - для произвольной колонки данных построить график "Ящик с усами (boxplot)".

Рубежный контроль №1

Калашников Артем ИУ5 63Б 11 Вариант

Импорт библиотек

B [57]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
```

B [58]:
```
DB = pd.read_csv('marvel-wikia-data.csv', sep=',')
```

B [59]:
```
# Первые пять строк датасета
DB.head()
```

Out[59]:

	page_id	name	urlslug	ID	ALIGN	EYE	HAIR	SEX	GSM	ALIVE	APPEARANCES	APPE
0	1678	Spider-Man (Peter Parker)	\/Spider-Man_(Peter_Parker)	Secret Identity	Good Characters	Hazel Eyes	Brown Hair	Male Characters	NaN	Living Characters	4043.0	
1	7139	Captain America (Steven Rogers)	\/Captain_America_(Steven_Rogers)	Public Identity	Good Characters	Blue Eyes	White Hair	Male Characters	NaN	Living Characters	3360.0	
2	64786	Wolverine (James \"Logan\" Howlett)	\/Wolverine_(James_%22Logan%22_Howlett)	Public Identity	Neutral Characters	Blue Eyes	Black Hair	Male Characters	NaN	Living Characters	3061.0	
3	1868	Iron Man (Anthony \"Tony\" Stark)	\/Iron_Man_(Anthony_%22Tony%22_Stark)	Public Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	NaN	Living Characters	2961.0	
4	2460	Thor (Thor Odinson)	\/Thor_(Thor_Odinson)	No Dual Identity	Good Characters	Blue Eyes	Blond Hair	Male Characters	NaN	Living Characters	2258.0	

In [60]:
```
# Размер датасета
DB.shape
```

Out[60]:
```
(16376, 13)
```

In [61]:
```
# Количество нулевых элементов
DB.isnull().sum()
```

Out[61]:
```
page_id                 0
name                    0
urlslug                 0
ID                   3770
ALIGN                2812
EYE                  9767
HAIR                 4264
SEX                   854
GSM                 16286
ALIVE                   3
APPEARANCES          1096
FIRST APPEARANCE      815
Year                  815
dtype: int64
```

In [62]:
```
# Колонки и их типы данных
DB.dtypes
```

Out[62]:
```
page_id               int64
name                 object
urlslug              object
ID                   object
ALIGN                object
EYE                  object
HAIR                 object
SEX                  object
GSM                  object
ALIVE                object
APPEARANCES         float64
FIRST APPEARANCE     object
Year                float64
dtype: object
```

In [63]:
```
# Колонки с пропусками и доля пропусков
cols_with_na = [c for c in DB.columns if DB[c].isnull().sum() > 0]
[(c, DB[c].isnull().mean()) for c in cols_with_na]
```

Out[63]:
```
[('ID', 0.23021494870542256),
 ('ALIGN', 0.17171470444553005),
 ('EYE', 0.5964215925744992),
 ('HAIR', 0.26038104543234003),
 ('SEX', 0.052149487054225695),
 ('GSM', 0.9945041524181729),
 ('ALIVE', 0.00018319491939423546),
 ('APPEARANCES', 0.06692721055202736),
 ('FIRST APPEARANCE', 0.04976795310210064),
 ('Year', 0.04976795310210064)]
```

Обработка пропусков для категориального признака

Можно заметить, что столбец "GSM" пропущено слишком много данных(99%). Проанализировав данный столбец, можно понять что пустые элементы означают гетеросексуальность персонажа, следовательно все пустые элементы можно заменить на "Heterosexual".

В [64]:
```
DBNew_3 = DB[['GSM']].fillna('Heterosexual')
DB[['GSM']]=DBNew_3
DB.head(16375)
```

Out[64]:

	page_id	name	urlslug	ID	ALIGN	EYE	HAIR	SEX	GSM	ALIVE	APPEAR
0	1678	Spider-Man (Peter Parker)	VSpider-Man_(Peter_Parker)	Secret Identity	Good Characters	Hazel Eyes	Brown Hair	Male Characters	Heterosexual	Living Characters	
1	7139	Captain America (Steven Rogers)	VCaptain_America_(Steven_Rogers)	Public Identity	Good Characters	Blue Eyes	White Hair	Male Characters	Heterosexual	Living Characters	
2	64786	Wolverine (James \"Logan\" Howlett)	VWolverine_(James_%22Logan%22_Howlett)	Public Identity	Neutral Characters	Blue Eyes	Black Hair	Male Characters	Heterosexual	Living Characters	
3	1868	Iron Man (Anthony \"Tony\" Stark)	VIron_Man_(Anthony_%22Tony%22_Stark)	Public Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	Heterosexual	Living Characters	
4	2460	Thor (Thor Odinson)	VThor_(Thor_Odinson)	No Dual Identity	Good Characters	Blue Eyes	Blond Hair	Male Characters	Heterosexual	Living Characters	
...	...	...	...	...	...	...	...	...	...	...	
16370	674414	Phoenix's Shadow (Earth-616)	VPhoenix%27s_Shadow_(Earth-616)	NaN	Neutral Characters	NaN	NaN	NaN	Heterosexual	Living Characters	
16371	657508	Ru'ach (Earth-616)	VRu%27ach_(Earth-616)	No Dual Identity	Bad Characters	Green Eyes	No Hair	Male Characters	Heterosexual	Living Characters	
16372	665474	Thane (Thanos' son) (Earth-616)	VThane_(Thanos%27_son)_(Earth-616)	No Dual Identity	Good Characters	Blue Eyes	Bald	Male Characters	Heterosexual	Living Characters	
16373	695217	Tinkerer (Skrull) (Earth-616)	VTinkerer_(Skrull)_(Earth-616)	Secret Identity	Bad Characters	Black Eyes	Bald	Male Characters	Heterosexual	Living Characters	
16374	708811	TK421 (Spiderling) (Earth-616)	VTK421_(Spiderling)_(Earth-616)	Secret Identity	Neutral Characters	NaN	NaN	Male Characters	Heterosexual	Living Characters	

16375 rows × 13 columns

Обработка пропусков для количественного признака

В [65]:
```
total_count = DB.shape[0]
Ncols = []
for col in DB.columns:
    temp_null_count = DB[DB[col].isnull()].shape[0]
    dt = str(DB[col].dtype)
    if temp_null_count>0 and (dt=='float64' or dt=='int64'):
        Ncols.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, ter
print(Ncols)
```

```
Колонка APPEARANCES. Тип данных float64. Количество пустых значений 1096, 6.69%.
Колонка Year. Тип данных float64. Количество пустых значений 815, 4.98%.
['APPEARANCES', 'Year']
```

Appearances

В столбце 'APPEARANCES' содержиться информация количестве появлений персонажа. Будем считать, что отсутствие информации говорит о том, что персонаж не появлялся. Поэтому заполним пропуски нулями.

В [66]:
```
DBNew = DB[['APPEARANCES']].fillna(0)
```

В [67]:
```
DB[['APPEARANCES']]=DBNew
```

Year

В столбце 'YEAR' содержиться информация о годе создания персонажа. Будем считать, что отсутствие информации говорит о том, что информация о создание персонажа отсутствует. Поэтому заполним пропуски в этих колонках '-1'.

В [68]:
```
DBNew_1 = DB[['Year']].fillna(-1)
DB[['Year']]=DBNew_1
```

Получившиеся значения

In [69]:
```
DB.head(16375)
```

Out[69]:

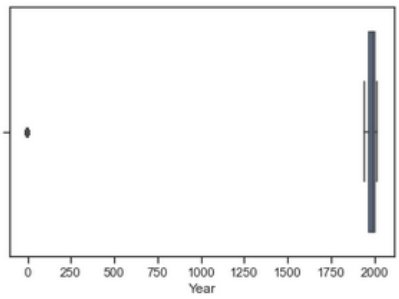
	page_id	name	urlslug	ID	ALIGN	EYE	HAIR	SEX	GSM	ALIVE	APPEAR
0	1678	Spider-Man (Peter Parker)	\/Spider-Man_(Peter_Parker)	Secret Identity	Good Characters	Hazel Eyes	Brown Hair	Male Characters	Heterosexual	Living Characters	
1	7139	Captain America (Steven Rogers)	\/Captain_America_(Steven_Rogers)	Public Identity	Good Characters	Blue Eyes	White Hair	Male Characters	Heterosexual	Living Characters	
2	64786	Wolverine (James \"Logan\" Howlett)	\/Wolverine_(James_%22Logan%22_Howlett)	Public Identity	Neutral Characters	Blue Eyes	Black Hair	Male Characters	Heterosexual	Living Characters	
3	1868	Iron Man (Anthony \"Tony\" Stark)	\/Iron_Man_(Anthony_%22Tony%22_Stark)	Public Identity	Good Characters	Blue Eyes	Black Hair	Male Characters	Heterosexual	Living Characters	
4	2460	Thor (Thor Odinson)	\/Thor_(Thor_Odinson)	No Dual Identity	Good Characters	Blue Eyes	Blond Hair	Male Characters	Heterosexual	Living Characters	
...	...	...	...	...	...	...	...	...	...	...	
16370	674414	Phoenix's Shadow (Earth-616)	\/Phoenix%27s_Shadow_(Earth-616)	NaN	Neutral Characters	NaN	NaN	NaN	Heterosexual	Living Characters	
16371	657508	Ru'ach (Earth-616)	\/Ru%27ach_(Earth-616)	No Dual Identity	Bad Characters	Green Eyes	No Hair	Male Characters	Heterosexual	Living Characters	
16372	665474	Thane (Thanos' son) (Earth-616)	\/Thane_(Thanos%27_son)_(Earth-616)	No Dual Identity	Good Characters	Blue Eyes	Bald	Male Characters	Heterosexual	Living Characters	
16373	695217	Tinkerer (Skrull) (Earth-616)	\/Tinkerer_(Skrull)_(Earth-616)	Secret Identity	Bad Characters	Black Eyes	Bald	Male Characters	Heterosexual	Living Characters	
16374	708811	TK421 (Spiderling) (Earth-616)	\/TK421_(Spiderling)_(Earth-616)	Secret Identity	Neutral Characters	NaN	NaN	Male Characters	Heterosexual	Living Characters	

16375 rows × 13 columns

Ящик с усами (boxplot)

In [70]:
```
sns.boxplot(x=DB['Year'])
```

Out[70]:
```
<AxesSubplot:xlabel='Year'>
```



In [71]:
```
sns.boxplot(x=DB['APPEARANCES'])
```

Out[71]:
```
<AxesSubplot:xlabel='APPEARANCES'>
```