Лабораторная работа №2

Обработка признаков (часть 1).

Рассмотрим исторические данные по выходу и продажам видеоигр из на 2019 год

Задание:

Оригинал:

- Выбрать набор данных (датасет), содержащий категориальные и числовые 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.) Просьба не использовать датасет, на котором данная задача решалась в лекции.
- Для выбранного датасета (датасетов) на основе материалов лекций решить следующие задачи:
    - устранение пропусков в данных;
    - кодирование категориальных признаков;
    - нормализацию числовых признаков.

```python
import numpy as np
import pandas as pd
import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

```python
df = pd.read_csv('vgsales-12-4-2019-short.csv')
print(f'Total loaded {len(df.index)} video games')
```

```
Total loaded 55792 video games
```

```python
# Объединим продажи в одну колонку
df['Sales'] = df[['Total_Shipped', 'Global_Sales']].max(axis=1)
df['Sales'].fillna(0, inplace=True)
```

```python
# Удалим лишнее
df = df.drop(columns = ['Total_Shipped', 'Global_Sales', 'NA_Sales', 'PAL_Sales', 'JP_Sales',
'Other_Sales'])
```

Добавим выброс

```python
df.tail(1)
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Rank	Name	Genre	ESRB_Rating	Platform	Publisher	Developer	Critic_Score	User_Score	Year	Sales
55791	55792	Falcon Age	Action-Adventure	NaN	PS4	Unknown	Outerloop Games	NaN	NaN	NaN	0.0

```python
fake_row = (66666, 'FakeGame', 'Music', 'E', 'NS', 'Drool', 'Drool', 29.0, 64.0, 2018.0, 0.0)
appendix = pd.DataFrame((fake_row,), columns=df.columns)
df = df.append(appendix, ignore_index=True)
df.tail(2)
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Rank	Name	Genre	ESRB_Rating	Platform	Publisher	Developer	Critic_Score	User_Score	Year	Sales
55791	55792	Falcon Age	Action-Adventure	NaN	PS4	Unknown	Outerloop Games	NaN	NaN	NaN	0.0
55792	66666	FakeGame	Music	E	NS	Drool	Drool	29.0	64.0	2018.0	0.0

Устранение пропусков в данных

```python
# Подсчитаем NaN-ы по каждой колонке
df.isna().sum()
```

```
Rank                0
Name                0
Genre               0
ESRB_Rating     32169
Platform            0
Publisher           0
Developer          17
Critic_Score    49256
User_Score      55457
Year              979
Sales               0
dtype: int64
```

```python
# Удалим неизвестные игры
df.dropna(subset=['Developer', 'Year'], inplace=True)
```

```python
# Слишком большой перевес NaN-ов относительно настоящих оценок. Лучше удалим
df.dropna(subset=['Critic_Score', 'User_Score'], inplace=True)
```

```python
df['ESRB_Rating'].unique()
```

```
array(['E', 'M', 'T', 'E10', nan], dtype=object)
```

```python
# «E» («Everyone») — «Для всех»
# Заменим пропуски на значение по умолчанию
df['ESRB_Rating'].fillna('E', inplace=True)
```

```python
# Заменим оценки критиков и пользователей на средние значения
for col in ('Critic_Score', 'User_Score'):
    df[col].fillna(df[col].mean(), inplace=True)
```

```python
# Подсчитаем NaN-ы по каждой колонке
df.isna().sum()
```

```
Rank            0
Name            0
Genre           0
ESRB_Rating     0
Platform        0
Publisher       0
Developer       0
Critic_Score    0
User_Score      0
Year            0
Sales           0
dtype: int64
```

Кодирование категориальных признаков

```
# One-Hot
pd.get_dummies(df[['ESRB_Rating']])
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	ESRB_Rating_E	ESRB_Rating_E10	ESRB_Rating_M	ESRB_Rating_T
2	1	0	0	0
4	1	0	0	0
6	1	0	0	0
8	1	0	0	0
11	1	0	0	0
...	...	...	...	...
48498	1	0	0	0
51097	0	1	0	0
54537	1	0	0	0
55423	0	1	0	0
55792	1	0	0	0

219 rows × 4 columns

```
from sklearn.preprocessing import LabelEncoder

for col in ('Genre', 'Platform', 'Publisher', 'Developer'):
    le = LabelEncoder()
    col_le = le.fit_transform(df[col])
    print(np.unique(df[col]), np.unique(col_le))
    df[col] = col_le

df.head()
```

```
['Action' 'Action-Adventure' 'Adventure' 'Fighting' 'MMO' 'Misc' 'Music'
 'Platform' 'Puzzle' 'Racing' 'Role-Playing' 'Shooter' 'Simulation'
 'Sports' 'Strategy'] [ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14]
['3DS' 'DS' 'GC' 'N64' 'NS' 'PC' 'PS' 'PS2' 'PS3' 'PS4' 'PSN' 'PSP' 'WW'
 'Wii' 'WiiU' 'X360' 'XB' 'XBL' 'XOne'] [ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18]
['2K Games' 'Activision' 'Aksys Games' 'Atari' 'Atlus'
 'Bethesda Softworks' 'Blizzard Entertainment' 'Capcom' 'DotEmu' 'Drool'
 'EA Sports' 'Electronic Arts' 'FDG Entertainment' 'Feardemic'
 'Hudson Entertainment' 'Ignition Entertainment' 'Invisible Handlebar'
 'Konami' 'LucasArts' 'Mediatonic' 'Microprose' 'Microsoft Game Studios'
 'Midway Games' 'NIS America' 'Namco Bandai' 'Natsume' 'Nicalis'
 'Nintendo' 'Paradox Interactive' 'RedOctane' 'Rockstar Games' 'Sega'
 'Sony Computer Entertainment' 'Sony Interactive Entertainment' 'Square'
 'Square Enix' 'Strategy First' 'THQ' 'Team17 Software' 'Ubisoft'
 'Valve Corporation' 'Warner Bros. Interactive' 'Working Designs'
 'Xseed Games'] [ 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]
['2K Australia / 2K Boston' '2K Boston / 2K Australia / 2K Marin'
 '989 Sports' 'Access Games' 'Alfa System' 'Arc System Works' 'Atlus Co.'
 'Bandai Namco Games' 'BestGameEver.com' 'Bethesda Game Studios'
 'Bethesda Softworks' 'BioWare Edmonton' 'Blizzard Entertainment'
 'Blue Tongue' 'Bright Light Productions' 'Bungie Studios' 'Capcom'
 'Capcom / Dimps Corporation' 'Capcom Production Studio 1'
 'Capcom Production Studio 1 / TOSE Software' 'Capcom Production Studio 4'
 'Cavia Inc.' 'Certain Affinity / Valve Software' 'Choice Provisions'
 'Chris Sawyer' 'Cing' 'Cing Inc. / Town Factory' 'Climax Group'
 'Climax Studios' 'Clover Studio' 'Crytek' 'Digital Eclipse' 'Drool'
 'EA Canada' 'EA Digital Illusions CE' 'EA Montreal' 'EA Vancouver'
 'EXAKT Entertainment' 'Ellipse Studios' 'Ensemble Studios' 'Epic Games'
 'Evolution Studios' 'Firaxis' 'Flagship' 'From Software' 'GRIN'
 'Game Arts' 'Game Atelier' 'Game Freak' 'Game Republic' 'Good-Feel'
 'Grasshopper Manufacture' 'Guerrilla Games' 'HAL Laboratory'
 'High Voltage Software' 'Hudson Entertainment' 'Infinity Ward'
 'Insomniac Games' 'Intelligent Systems' 'IronNos Co., Ltd.'
 'Just Add Water' 'K2 / Kurogane' 'Kojima Productions' 'Konami'
 'Konami / Eighting' 'Konami Computer Entertainment Japan'
 'Konami Computer Entertainment Tokyo' 'Krome Studios' 'Level 5'
 'Lionhead Studios' 'Lizardcube' 'MPS Labs' 'Matrix Software'
 'Media Molecule' 'Mediatonic Ltd.'
 'Mistwalker Corporation / Feelplus Inc.' 'Monolith Productions'
 'Monolith Soft' 'Namco' 'Namco Tales Studio' 'Naughty Dog' 'Neversoft'
 'Nihon Falcom Corporation' 'Nintendo EAD' 'Nintendo EAD Tokyo'
 'Nintendo EPD' 'Nintendo Software Technology Corporation'
 'Nippon Ichi Software' 'Omega Force' 'PlatinumGames' 'Polyphony Digital'
 'Project Sora' 'Quantic Dream' 'Rainbow Studios' 'Rare Ltd.'
 'Raven Software' 'Ready at Dawn' 'Ready at Dawn Studios'
 'Realtime Worlds' 'Retro Studios' 'Rhino Studios' 'Rockstar Leeds/North'
 'Rockstar North' 'Rockstar San Diego' 'SCEA Bend Studio'
 'SCEA Santa Monica Studio' 'SCEI' 'SFB Games' 'Secret Level' 'Sega'
 'Sega WOW Overworks' 'Sonic Team' 'Sonic Team/Dimps Corporation'
 'Spike / Bandai Namco Games' 'Square Enix' 'SquareSoft'
 'Sucker Punch Productions' 'Taleworlds' 'Team Bondi' 'Team ICO'
 'Team Silent' 'Team17 Software' 'The Deep End Games' 'Treyarch'
 'Turn 10 Studios' 'Ubisoft' 'Ubisoft Montreal' 'Ubisoft Paris'
 'Ubisoft Romania' 'Ubisoft San Francisco' 'Ubisoft Shanghai'
 'Ubisoft Toronto' 'Valhalla Game Studios' 'Valve Software' 'Vanillaware'
 'Vicarious Visions' 'Vigil Games' 'Visceral Games'
 "Yuke's Future Media Creators" 'Zipper Interactive' 'id Software'
 'n-Space' 'thatgamecompany' 'tri-Ace'] [  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]
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Rank	Name	Genre	ESRB_Rating	Platform	Publisher	Developer	Critic_Score	User_Score	Year	Sales
2	3	Mario Kart Wii	9	E	13	27	83	8.2	9.1	2008.0	37.14
4	5	Wii Sports Resort	13	E	13	27	83	8.0	8.8	2009.0	33.09
6	7	New Super Mario Bros.	7	E	1	27	83	9.1	8.1	2006.0	30.80
8	9	New Super Mario Bros. Wii	7	E	13	27	83	8.6	9.2	2009.0	30.22
11	12	Wii Play	5	E	13	27	83	5.9	4.5	2007.0	28.02

```
# Static label encoding
di = {
    'RP':  0,
    'EC':  1,
    'KA':  2,
    'E':   3,
    'E10': 4,
    'T':   5,
    'M':   6,
    'AO':  7,
}

df.replace({'ESRB_Rating': di}, inplace=True)
df.head()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Rank	Name	Genre	ESRB_Rating	Platform	Publisher	Developer	Critic_Score	User_Score	Year	Sales
2	3	Mario Kart Wii	9	3	13	27	83	8.2	9.1	2008.0	37.14
4	5	Wii Sports Resort	13	3	13	27	83	8.0	8.8	2009.0	33.09
6	7	New Super Mario Bros.	7	3	1	27	83	9.1	8.1	2006.0	30.80
8	9	New Super Mario Bros. Wii	7	3	13	27	83	8.6	9.2	2009.0	30.22
11	12	Wii Play	5	3	13	27	83	5.9	4.5	2007.0	28.02

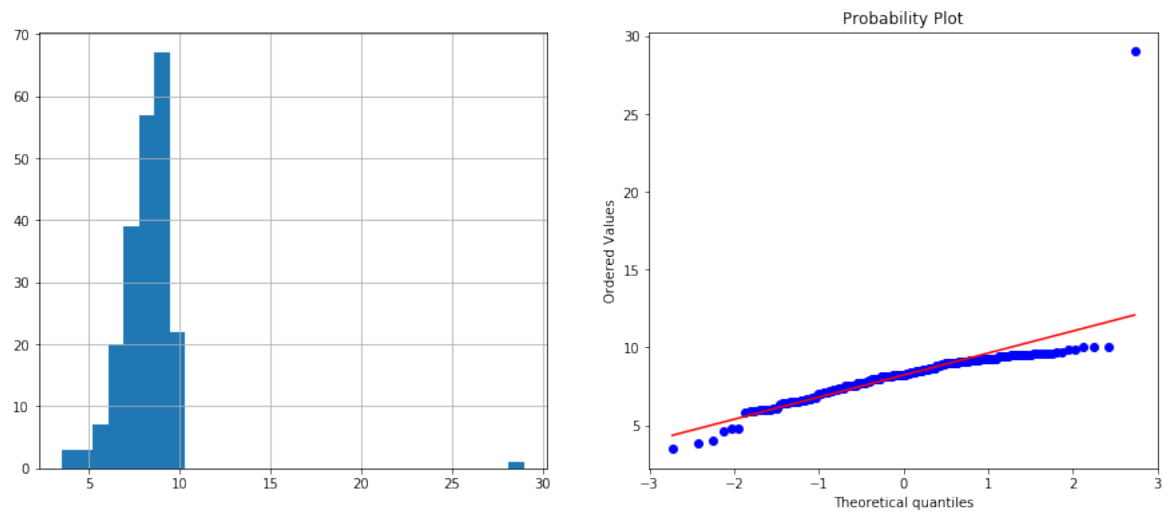
Нормализация числовых признаков

```
import scipy.stats as stats
```

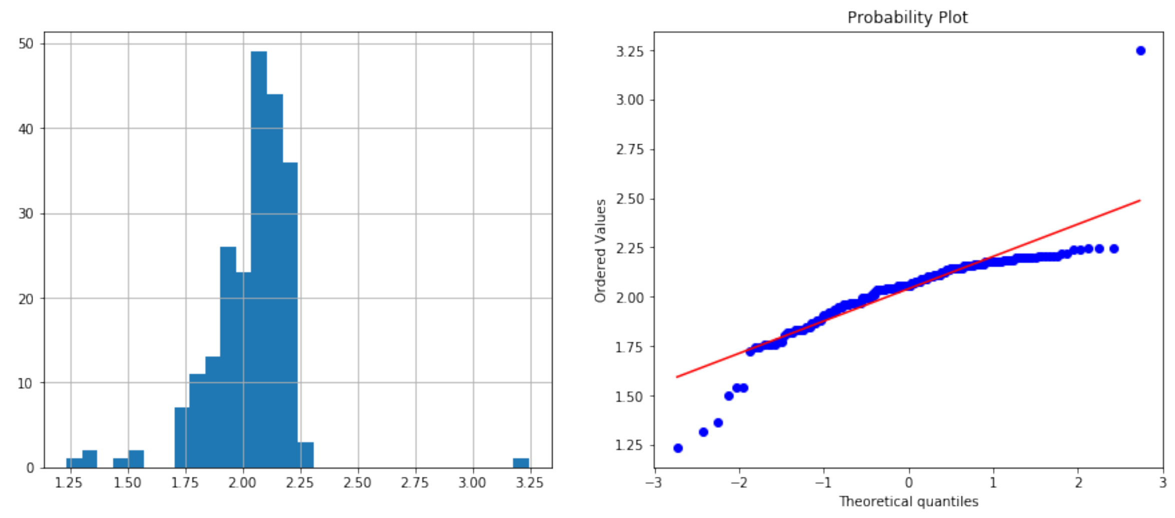
```
def diagnostic_plots(df, variable):
    plt.figure(figsize=(15,6))
    # гистограмма
    plt.subplot(1, 2, 1)
    df[variable].hist(bins=30)
    ## Q-Q plot
    plt.subplot(1, 2, 2)
    stats.probplot(df[variable], dist="norm", plot=plt)
    plt.show()
```

```
def boxcox(data, col):
    diagnostic_plots(data, col)
    data[f'{col}_boxcox'], param = stats.boxcox(data[col])
    print('Оптимальное значение λ = {}'.format(param))
    diagnostic_plots(data, f'{col}_boxcox')
```

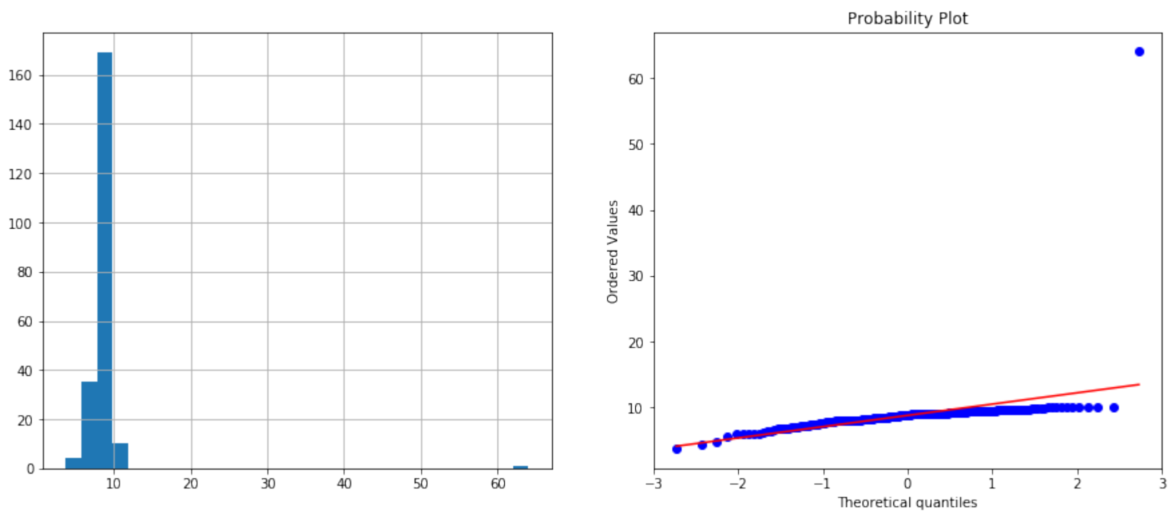
```
boxcox(df, 'Critic_Score')
```



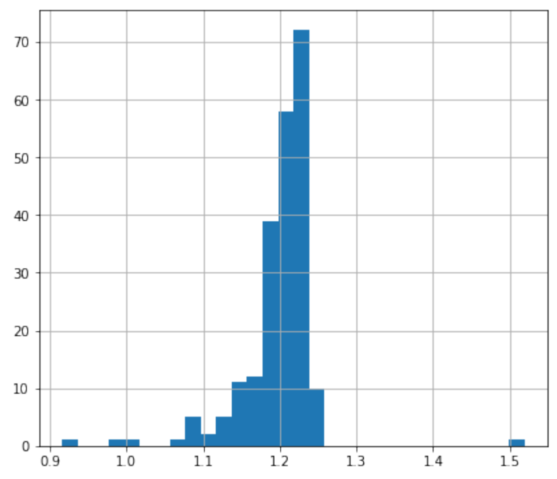
```
Оптимальное значение λ = -0.021821216576747314
```

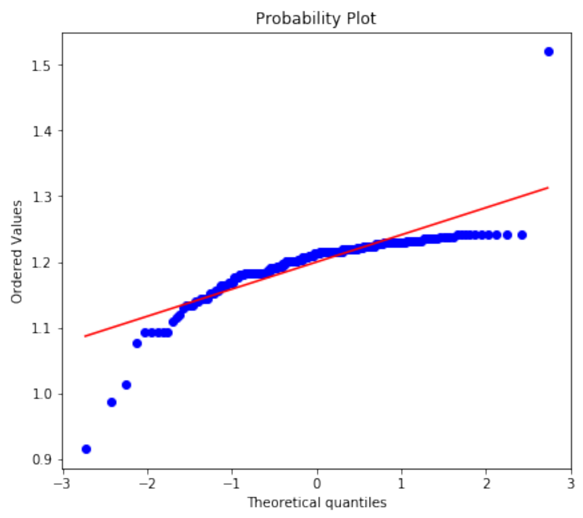


```
boxcox(df, 'User_Score')
```



```
Оптимальное значение λ = -0.6048001579673186
```





```
df.head()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Rank	Name	Genre	ESRB_Rating	Platform	Publisher	Developer	Critic_Score	User_Score	Year	Sales	Critic_Score_boxcox	Us
2	3	Mario Kart Wii	9	3	13	27	83	8.2	9.1	2008.0	37.14	2.056560	1.2
4	5	Wii Sports Resort	13	3	13	27	83	8.0	8.8	2009.0	33.09	2.032969	1.2
6	7	New Super Mario Bros.	7	3	1	27	83	9.1	8.1	2006.0	30.80	2.155914	1.1
8	9	New Super Mario Bros. Wii	7	3	13	27	83	8.6	9.2	2009.0	30.22	2.102027	1.2
11	12	Wii Play	5	3	13	27	83	5.9	4.5	2007.0	28.02	1.741018	0.9

```
df.tail(1)
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Rank	Name	Genre	ESRB_Rating	Platform	Publisher	Developer	Critic_Score	User_Score	Year	Sales	Critic_Score_boxco
55792	66666	FakeGame	6	3	4	9	32	29.0	64.0	2018.0	0.0	3.246559

Сохраняем

```
save_path = 'video_games_s2.csv'
df.to_csv(save_path, index=False)
```

```
check_df = pd.read_csv(save_path)
check_df.head()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Rank	Name	Genre	ESRB_Rating	Platform	Publisher	Developer	Critic_Score	User_Score	Year	Sales	Critic_Score_boxcox	Use
0	3	Mario Kart Wii	9	3	13	27	83	8.2	9.1	2008.0	37.14	2.056560	1.21
1	5	Wii Sports Resort	13	3	13	27	83	8.0	8.8	2009.0	33.09	2.032969	1.20
2	7	New Super Mario Bros.	7	3	1	27	83	9.1	8.1	2006.0	30.80	2.155914	1.18
3	9	New Super Mario Bros. Wii	7	3	13	27	83	8.6	9.2	2009.0	30.22	2.102027	1.22
4	12	Wii Play	5	3	13	27	83	5.9	4.5	2007.0	28.02	1.741018	0.98