Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Отчет по лабораторной работы №2

по дисциплине «Методы машинного обучения»

по теме «Обработка признаков (часть 1)»

Выполнил:
студент группы № ИУ5-21М
Торжков М.С.
подпись, дата

Проверила:
Балашов А.М.
подпись, дата

2023 г.

Задание.

1. 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.

2. Для выбранного датасета (датасетов) на основе материалов лекций решить следующие задачи:

- устранение пропусков в данных;
- кодирование категориальных признаков;
- нормализация числовых признаков.

Импортирование необходимых библиотек

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import scipy.stats as stats
from google.colab import drive

data = pd.read_csv("house_sales.csv")

data = data.drop('Id', 1)
data.head()

<ipython-input-3-c100a8de87ec>:1: FutureWarning: In a future version
of pandas all arguments of DataFrame.drop except for the argument
'labels' will be keyword-only.
  data = data.drop('Id', 1)

   MSSubClass MSZoning  LotFrontage  LotArea Street Alley LotShape  \
0          60       RL         65.0     8450   Pave   NaN      Reg
1          20       RL         80.0     9600   Pave   NaN      Reg
2          60       RL         68.0    11250   Pave   NaN      IR1
3          70       RL         60.0     9550   Pave   NaN      IR1
4          60       RL         84.0    14260   Pave   NaN      IR1

  LandContour Utilities LotConfig  ... PoolArea PoolQC Fence
MiscFeature  \
0         Lvl    AllPub    Inside  ...        0    NaN   NaN
NaN
1         Lvl    AllPub       FR2  ...        0    NaN   NaN
NaN
2         Lvl    AllPub    Inside  ...        0    NaN   NaN
NaN
3         Lvl    AllPub    Corner  ...        0    NaN   NaN
NaN
4         Lvl    AllPub       FR2  ...        0    NaN   NaN
NaN

  MiscVal MoSold  YrSold  SaleType  SaleCondition  SalePrice
0       0      2    2008        WD         Normal     208500
1       0      5    2007        WD         Normal     181500
2       0      9    2008        WD         Normal     223500
3       0      2    2006        WD        Abnorml     140000
4       0     12    2008        WD         Normal     250000

[5 rows x 80 columns]

data_features = list(zip(
[i for i in data.columns], # название признака
zip(
```

```python
    [str(i) for i in data.dtypes], # типы колонок
    [i for i in data.isnull().sum()] # количество пропусков в колонке
)))
data_features # Признаки с типом данных и количеством пропусков
```

```
[('MSSubClass', ('int64', 0)),
 ('MSZoning', ('object', 0)),
 ('LotFrontage', ('float64', 259)),
 ('LotArea', ('int64', 0)),
 ('Street', ('object', 0)),
 ('Alley', ('object', 1369)),
 ('LotShape', ('object', 0)),
 ('LandContour', ('object', 0)),
 ('Utilities', ('object', 0)),
 ('LotConfig', ('object', 0)),
 ('LandSlope', ('object', 0)),
 ('Neighborhood', ('object', 0)),
 ('Condition1', ('object', 0)),
 ('Condition2', ('object', 0)),
 ('BldgType', ('object', 0)),
 ('HouseStyle', ('object', 0)),
 ('OverallQual', ('int64', 0)),
 ('OverallCond', ('int64', 0)),
 ('YearBuilt', ('int64', 0)),
 ('YearRemodAdd', ('int64', 0)),
 ('RoofStyle', ('object', 0)),
 ('RoofMatl', ('object', 0)),
 ('Exterior1st', ('object', 0)),
 ('Exterior2nd', ('object', 0)),
 ('MasVnrType', ('object', 8)),
 ('MasVnrArea', ('float64', 8)),
 ('ExterQual', ('object', 0)),
 ('ExterCond', ('object', 0)),
 ('Foundation', ('object', 0)),
 ('BsmtQual', ('object', 37)),
 ('BsmtCond', ('object', 37)),
 ('BsmtExposure', ('object', 38)),
 ('BsmtFinType1', ('object', 37)),
 ('BsmtFinSF1', ('int64', 0)),
 ('BsmtFinType2', ('object', 38)),
 ('BsmtFinSF2', ('int64', 0)),
 ('BsmtUnfSF', ('int64', 0)),
 ('TotalBsmtSF', ('int64', 0)),
 ('Heating', ('object', 0)),
 ('HeatingQC', ('object', 0)),
 ('CentralAir', ('object', 0)),
 ('Electrical', ('object', 1)),
 ('1stFlrSF', ('int64', 0)),
 ('2ndFlrSF', ('int64', 0)),
 ('LowQualFinSF', ('int64', 0)),
```

```
 ('GrLivArea', ('int64', 0)),
 ('BsmtFullBath', ('int64', 0)),
 ('BsmtHalfBath', ('int64', 0)),
 ('FullBath', ('int64', 0)),
 ('HalfBath', ('int64', 0)),
 ('BedroomAbvGr', ('int64', 0)),
 ('KitchenAbvGr', ('int64', 0)),
 ('KitchenQual', ('object', 0)),
 ('TotRmsAbvGrd', ('int64', 0)),
 ('Functional', ('object', 0)),
 ('Fireplaces', ('int64', 0)),
 ('FireplaceQu', ('object', 690)),
 ('GarageType', ('object', 81)),
 ('GarageYrBlt', ('float64', 81)),
 ('GarageFinish', ('object', 81)),
 ('GarageCars', ('int64', 0)),
 ('GarageArea', ('int64', 0)),
 ('GarageQual', ('object', 81)),
 ('GarageCond', ('object', 81)),
 ('PavedDrive', ('object', 0)),
 ('WoodDeckSF', ('int64', 0)),
 ('OpenPorchSF', ('int64', 0)),
 ('EnclosedPorch', ('int64', 0)),
 ('3SsnPorch', ('int64', 0)),
 ('ScreenPorch', ('int64', 0)),
 ('PoolArea', ('int64', 0)),
 ('PoolQC', ('object', 1453)),
 ('Fence', ('object', 1179)),
 ('MiscFeature', ('object', 1406)),
 ('MiscVal', ('int64', 0)),
 ('MoSold', ('int64', 0)),
 ('YrSold', ('int64', 0)),
 ('SaleType', ('object', 0)),
 ('SaleCondition', ('object', 0)),
 ('SalePrice', ('int64', 0))]
```

Устранение пропусков

```
# Доля (процент) пропусков для каждого признака
[(c, data[c].isnull().mean()) for c in data.columns]
```

```
[('MSSubClass', 0.0),
 ('MSZoning', 0.0),
 ('LotFrontage', 0.1773972602739726),
 ('LotArea', 0.0),
 ('Street', 0.0),
 ('Alley', 0.9376712328767123),
 ('LotShape', 0.0),
 ('LandContour', 0.0),
 ('Utilities', 0.0),
 ('LotConfig', 0.0),
```

```
('LandSlope', 0.0),
('Neighborhood', 0.0),
('Condition1', 0.0),
('Condition2', 0.0),
('BldgType', 0.0),
('HouseStyle', 0.0),
('OverallQual', 0.0),
('OverallCond', 0.0),
('YearBuilt', 0.0),
('YearRemodAdd', 0.0),
('RoofStyle', 0.0),
('RoofMatl', 0.0),
('Exterior1st', 0.0),
('Exterior2nd', 0.0),
('MasVnrType', 0.005479452054794521),
('MasVnrArea', 0.005479452054794521),
('ExterQual', 0.0),
('ExterCond', 0.0),
('Foundation', 0.0),
('BsmtQual', 0.025342465753424658),
('BsmtCond', 0.025342465753424658),
('BsmtExposure', 0.026027397260273973),
('BsmtFinType1', 0.025342465753424658),
('BsmtFinSF1', 0.0),
('BsmtFinType2', 0.026027397260273973),
('BsmtFinSF2', 0.0),
('BsmtUnfSF', 0.0),
('TotalBsmtSF', 0.0),
('Heating', 0.0),
('HeatingQC', 0.0),
('CentralAir', 0.0),
('Electrical', 0.0006849315068493151),
('1stFlrSF', 0.0),
('2ndFlrSF', 0.0),
('LowQualFinSF', 0.0),
('GrLivArea', 0.0),
('BsmtFullBath', 0.0),
('BsmtHalfBath', 0.0),
('FullBath', 0.0),
('HalfBath', 0.0),
('BedroomAbvGr', 0.0),
('KitchenAbvGr', 0.0),
('KitchenQual', 0.0),
('TotRmsAbvGrd', 0.0),
('Functional', 0.0),
('Fireplaces', 0.0),
('FireplaceQu', 0.4726027397260274),
('GarageType', 0.05547945205479452),
('GarageYrBlt', 0.05547945205479452),
('GarageFinish', 0.05547945205479452),
```

```
 ('GarageCars', 0.0),
 ('GarageArea', 0.0),
 ('GarageQual', 0.05547945205479452),
 ('GarageCond', 0.05547945205479452),
 ('PavedDrive', 0.0),
 ('WoodDeckSF', 0.0),
 ('OpenPorchSF', 0.0),
 ('EnclosedPorch', 0.0),
 ('3SsnPorch', 0.0),
 ('ScreenPorch', 0.0),
 ('PoolArea', 0.0),
 ('PoolQC', 0.9952054794520548),
 ('Fence', 0.8075342465753425),
 ('MiscFeature', 0.963013698630137),
 ('MiscVal', 0.0),
 ('MoSold', 0.0),
 ('YrSold', 0.0),
 ('SaleType', 0.0),
 ('SaleCondition', 0.0),
 ('SalePrice', 0.0)]
```

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

```
      MSSubClass MSZoning  LotArea Street LotShape LandContour
Utilities  \
0             60       RL     8450   Pave      Reg         Lvl
AllPub
1             20       RL     9600   Pave      Reg         Lvl
AllPub
2             60       RL    11250   Pave      IR1         Lvl
AllPub
3             70       RL     9550   Pave      IR1         Lvl
AllPub
4             60       RL    14260   Pave      IR1         Lvl
AllPub
...          ...      ...      ...    ...      ...         ...        .
..
1455          60       RL     7917   Pave      Reg         Lvl
AllPub
1456          20       RL    13175   Pave      Reg         Lvl
AllPub
1457          70       RL     9042   Pave      Reg         Lvl
AllPub
1458          20       RL     9717   Pave      Reg         Lvl
AllPub
1459          20       RL     9937   Pave      Reg         Lvl
AllPub

     LotConfig LandSlope Neighborhood  ... EnclosedPorch 3SsnPorch  \
```

```
0       Inside       Gtl      CollgCr  ...           0        0
1          FR2       Gtl      Veenker  ...           0        0
2       Inside       Gtl      CollgCr  ...           0        0
3       Corner       Gtl      Crawfor  ...         272        0
4          FR2       Gtl      NoRidge  ...           0        0
...        ...       ...          ...  ...         ...      ...
1455    Inside       Gtl      Gilbert  ...           0        0
1456    Inside       Gtl       NWAmes  ...           0        0
1457    Inside       Gtl      Crawfor  ...           0        0
1458    Inside       Gtl        NAmes  ...         112        0
1459    Inside       Gtl      Edwards  ...           0        0

     ScreenPorch PoolArea  MiscVal  MoSold  YrSold  SaleType
SaleCondition  \
0              0        0        0       2    2008        WD
Normal
1              0        0        0       5    2007        WD
Normal
2              0        0        0       9    2008        WD
Normal
3              0        0        0       2    2006        WD
Abnorml
4              0        0        0      12    2008        WD
Normal
...          ...      ...      ...     ...     ...       ...
...
1455           0        0        0       8    2007        WD
Normal
1456           0        0        0       2    2010        WD
Normal
1457           0        0     2500       5    2010        WD
Normal
1458           0        0        0       4    2010        WD
Normal
1459           0        0        0       6    2008        WD
Normal

     SalePrice
0       208500
1       181500
2       223500
3       140000
4       250000
...        ...
1455    175000
1456    210000
1457    266500
1458    142125
1459    147500
```

```
[1460 rows x 61 columns]

# Удаление колонок с высоким процентом пропусков (более 50%)
data.dropna(axis=1, thresh=730)

      MSSubClass MSZoning  LotFrontage  LotArea Street LotShape
LandContour  \
0             60       RL         65.0     8450   Pave      Reg
Lvl
1             20       RL         80.0     9600   Pave      Reg
Lvl
2             60       RL         68.0    11250   Pave      IR1
Lvl
3             70       RL         60.0     9550   Pave      IR1
Lvl
4             60       RL         84.0    14260   Pave      IR1
Lvl
...          ...      ...          ...      ...    ...      ...
...
1455          60       RL         62.0     7917   Pave      Reg
Lvl
1456          20       RL         85.0    13175   Pave      Reg
Lvl
1457          70       RL         66.0     9042   Pave      Reg
Lvl
1458          20       RL         68.0     9717   Pave      Reg
Lvl
1459          20       RL         75.0     9937   Pave      Reg
Lvl

     Utilities LotConfig LandSlope  ... EnclosedPorch 3SsnPorch
ScreenPorch  \
0       AllPub    Inside       Gtl  ...             0         0
0
1       AllPub       FR2       Gtl  ...             0         0
0
2       AllPub    Inside       Gtl  ...             0         0
0
3       AllPub    Corner       Gtl  ...           272         0
0
4       AllPub       FR2       Gtl  ...             0         0
0
...        ...       ...       ...  ...           ...       ...
...
1455    AllPub    Inside       Gtl  ...             0         0
0
1456    AllPub    Inside       Gtl  ...             0         0
0
1457    AllPub    Inside       Gtl  ...             0         0
```

```
0
1458    AllPub    Inside       Gtl  ...           112         0
0
1459    AllPub    Inside       Gtl  ...             0         0
0

     PoolArea MiscVal  MoSold  YrSold  SaleType  SaleCondition
SalePrice
0           0       0       2    2008        WD         Normal
208500
1           0       0       5    2007        WD         Normal
181500
2           0       0       9    2008        WD         Normal
223500
3           0       0       2    2006        WD        Abnorml
140000
4           0       0      12    2008        WD         Normal
250000
...       ...     ...     ...     ...       ...            ...        .
..
1455        0       0       8    2007        WD         Normal
175000
1456        0       0       2    2010        WD         Normal
210000
1457        0    2500       5    2010        WD         Normal
266500
1458        0       0       4    2010        WD         Normal
142125
1459        0       0       6    2008        WD         Normal
147500

[1460 rows x 76 columns]
```

```python
# Заполним пропуски возраста средними значениями
def impute_na(df, variable, value):
    df[variable].fillna(value, inplace=True)
impute_na(data, 'LotFrontage', data['LotFrontage'].mean())

# Убедимся, что признак LotFrontage не имеет пустых значений
data.isnull().sum()
```

```
MSSubClass       0
MSZoning         0
LotFrontage      0
LotArea          0
Street           0
                ..
MoSold           0
YrSold           0
SaleType         0
SaleCondition    0
```

```
SalePrice        0
Length: 80, dtype: int64
```

Кодирование категориальных признаков

```
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
cat_enc_le = le.fit_transform(data['SaleCondition'])

data['SaleCondition'].unique()

array(['Normal', 'Abnorml', 'Partial', 'AdjLand', 'Alloca', 'Family'],
      dtype=object)

np.unique(cat_enc_le)

array([0, 1, 2, 3, 4, 5])

le.inverse_transform([0, 1, 2, 3, 4, 5])

array(['Abnorml', 'AdjLand', 'Alloca', 'Family', 'Normal', 'Partial'],
      dtype=object)

data['LotConfig'].unique()

array(['Inside', 'FR2', 'Corner', 'CulDSac', 'FR3'], dtype=object)

pip install category_encoders

Looking in indexes: https://pypi.org/simple, https://us-
python.pkg.dev/colab-wheels/public/simple/
Collecting category_encoders
  Downloading category_encoders-2.6.0-py2.py3-none-any.whl (81 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 81.2/81.2 KB 4.6 MB/s eta
0:00:00
ent already satisfied: scipy>=1.0.0 in /usr/local/lib/python3.9/dist-
packages (from category_encoders) (1.10.1)
Requirement already satisfied: scikit-learn>=0.20.0 in
/usr/local/lib/python3.9/dist-packages (from category_encoders)
(1.2.2)
Requirement already satisfied: statsmodels>=0.9.0 in
/usr/local/lib/python3.9/dist-packages (from category_encoders)
(0.13.5)
Requirement already satisfied: pandas>=1.0.5 in
/usr/local/lib/python3.9/dist-packages (from category_encoders)
(1.4.4)
Requirement already satisfied: patsy>=0.5.1 in
/usr/local/lib/python3.9/dist-packages (from category_encoders)
(0.5.3)
Requirement already satisfied: numpy>=1.14.0 in
/usr/local/lib/python3.9/dist-packages (from category_encoders)
(1.22.4)
```

```
Requirement already satisfied: python-dateutil>=2.8.1 in
/usr/local/lib/python3.9/dist-packages (from pandas>=1.0.5-
>category_encoders) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in
/usr/local/lib/python3.9/dist-packages (from pandas>=1.0.5-
>category_encoders) (2022.7.1)
Requirement already satisfied: six in /usr/local/lib/python3.9/dist-
packages (from patsy>=0.5.1->category_encoders) (1.16.0)
Requirement already satisfied: joblib>=1.1.1 in
/usr/local/lib/python3.9/dist-packages (from scikit-learn>=0.20.0-
>category_encoders) (1.1.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in
/usr/local/lib/python3.9/dist-packages (from scikit-learn>=0.20.0-
>category_encoders) (3.1.0)
Requirement already satisfied: packaging>=21.3 in
/usr/local/lib/python3.9/dist-packages (from statsmodels>=0.9.0-
>category_encoders) (23.0)
Installing collected packages: category_encoders
Successfully installed category_encoders-2.6.0
```

```python
#CountEncoder
from category_encoders.count import CountEncoder as ce_CountEncoder

ce_CountEncoder1 = ce_CountEncoder()
data_COUNT_ENC =
ce_CountEncoder1.fit_transform(data[data.columns.difference(['SaleType
'])])

data_COUNT_ENC.head()
```

```
   1stFlrSF  2ndFlrSF  3SsnPorch  Alley  BedroomAbvGr  BldgType
BsmtCond  \
0       856       854          0   1369             3      1220
1311
1      1262         0          0   1369             3      1220
1311
2       920       866          0   1369             3      1220
1311
3       961       756          0   1369             3      1220
65
4      1145      1053          0   1369             4      1220
1311

   BsmtExposure  BsmtFinSF1  BsmtFinSF2  ...  SalePrice  ScreenPorch
Street  \
0           953         706           0  ...     208500            0
1454
1           134         978           0  ...     181500            0
1454
2           114         486           0  ...     223500            0
1454
```

```
3           953         216           0  ...     140000            0
1454
4           221         655           0  ...     250000            0
1454

   TotRmsAbvGrd  TotalBsmtSF  Utilities  WoodDeckSF  YearBuilt
YearRemodAdd  \
0             8          856       1459           0       2003
2003
1             6         1262       1459         298       1976
1976
2             6          920       1459           0       2001
2002
3             7          756       1459           0       1915
1970
4             9         1145       1459         192       2000
2000

   YrSold
0    2008
1    2007
2    2008
3    2006
4    2008

[5 rows x 79 columns]
```

```
data['MSZoning'].unique()
```

```
array(['RL', 'RM', 'C (all)', 'FV', 'RH'], dtype=object)
```

```
data_COUNT_ENC['MSZoning'].unique()
```

```
array([1151,  218,   10,   65,   16])
```

```
ce_CountEncoder2 = ce_CountEncoder(normalize=True)
data_FREQ_ENC =
ce_CountEncoder2.fit_transform(data[data.columns.difference(['SaleType
'])])
```

```
data_FREQ_ENC['MSZoning'].unique()
```

```
array([0.78835616, 0.14931507, 0.00684932, 0.04452055, 0.0109589 ])
```

```
from category_encoders.helmert import HelmertEncoder as
ce_HelmertEncoder
```

```
#HelmetEncoder
ce_HelmertEncoder1 = ce_HelmertEncoder()
data_HELM_ENC =
ce_HelmertEncoder1.fit_transform(data[data.columns.difference(['SaleTy
pe'])], data['SaleType'])
```

```
/usr/local/lib/python3.9/dist-packages/category_encoders/
base_contrast_encoder.py:126: FutureWarning: Intercept column might
not be added anymore in future releases (c.f. issue #370)
  warnings.warn("Intercept column might not be added anymore in future
releases (c.f. issue #370)",
/usr/local/lib/python3.9/dist-packages/category_encoders/base_contrast
_encoder.py:126: FutureWarning: Intercept column might not be added
anymore in future releases (c.f. issue #370)
  warnings.warn("Intercept column might not be added anymore in future
releases (c.f. issue #370)",
```

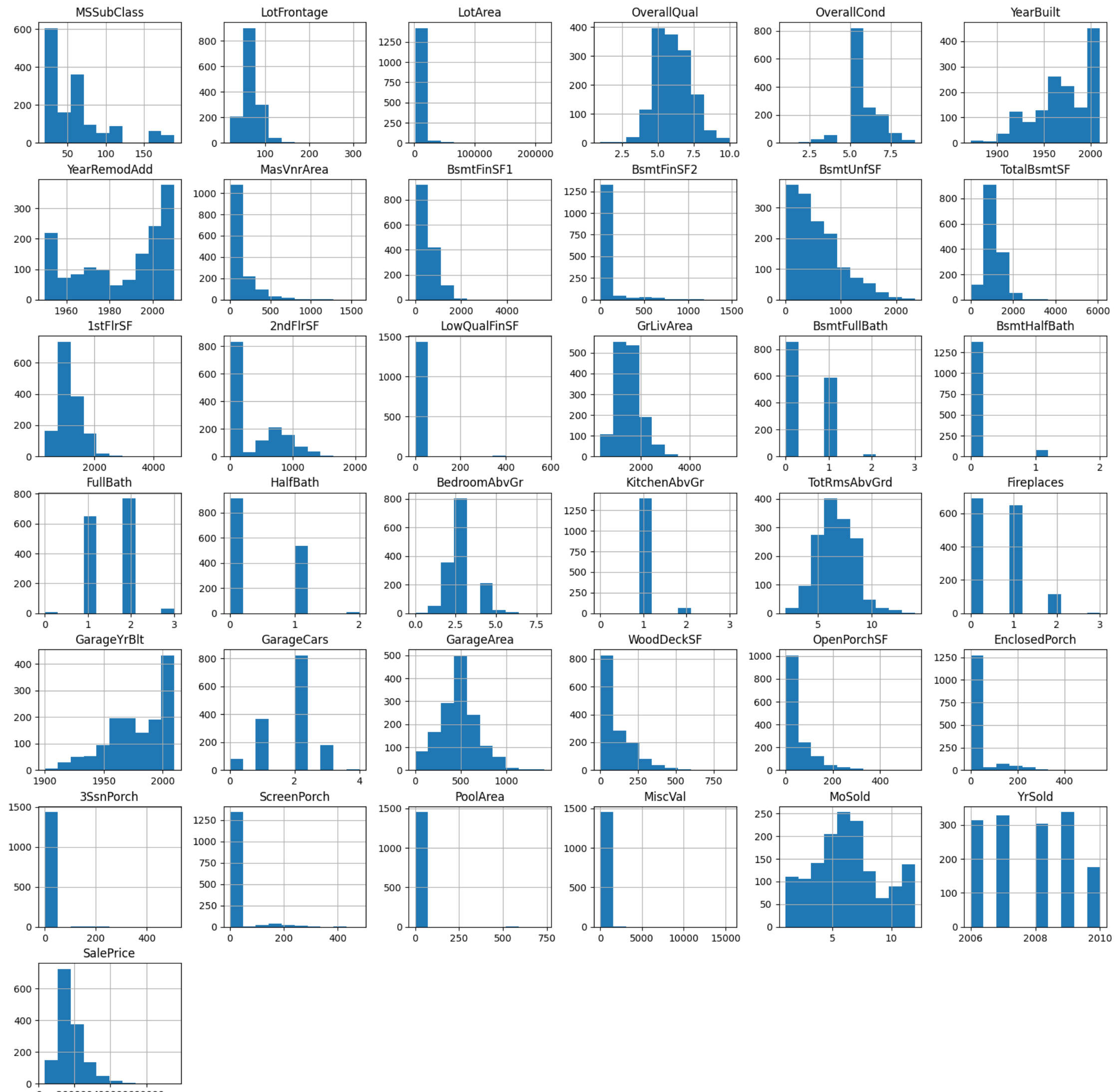
```
data_HELM_ENC.head()
```

```
   intercept  1stFlrSF  2ndFlrSF  3SsnPorch  Alley_0  Alley_1
BedroomAbvGr  \
0          1       856       854          0     -1.0     -1.0
3
1          1      1262         0          0     -1.0     -1.0
3
2          1       920       866          0     -1.0     -1.0
3
3          1       961       756          0     -1.0     -1.0
3
4          1      1145      1053          0     -1.0     -1.0
4

   BldgType_0  BldgType_1  BldgType_2  ...  SalePrice  ScreenPorch
Street_0  \
0        -1.0        -1.0        -1.0  ...     208500            0
-1.0
1        -1.0        -1.0        -1.0  ...     181500            0
-1.0
2        -1.0        -1.0        -1.0  ...     223500            0
-1.0
3        -1.0        -1.0        -1.0  ...     140000            0
-1.0
4        -1.0        -1.0        -1.0  ...     250000            0
-1.0

   TotRmsAbvGrd  TotalBsmtSF  Utilities_0  WoodDeckSF  YearBuilt  \
0             8          856         -1.0           0       2003
1             6         1262         -1.0         298       1976
2             6          920         -1.0           0       2001
3             7          756         -1.0           0       1915
4             9         1145         -1.0         192       2000

   YearRemodAdd  YrSold
0          2003    2008
1          1976    2007
2          2002    2008
```

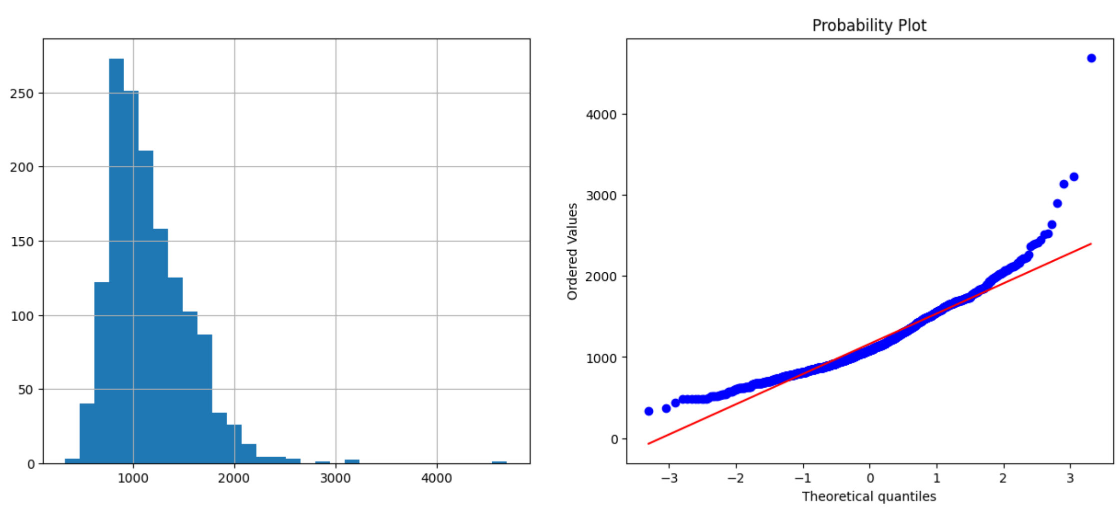
```
3           1970    2006
4           2000    2008

[5 rows x 255 columns]
```

Нормализация числовых признаков

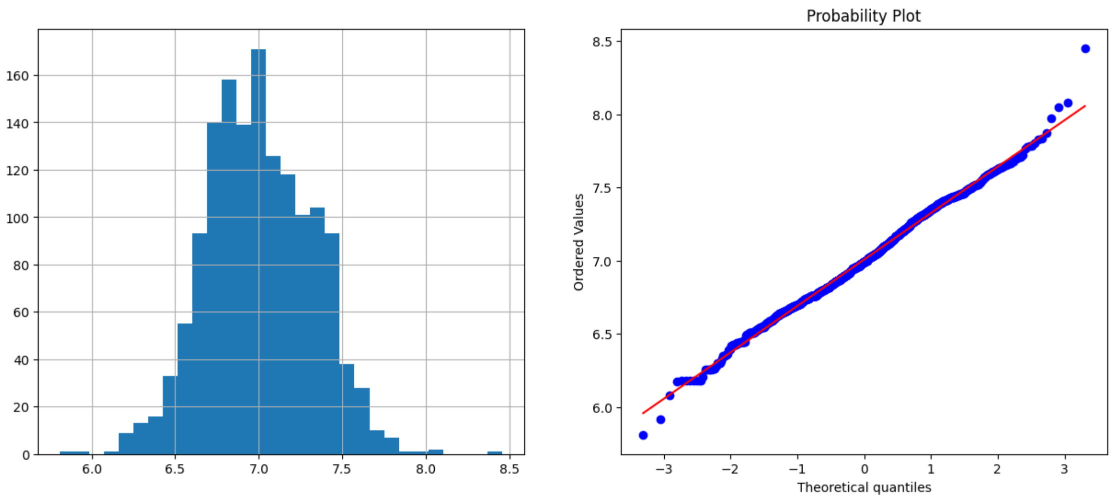
```
def diagnostic_plots(df, variable):
    plt.figure(figsize=(15,6))
    # гистограмма
    plt.subplot(1, 2, 1)
    df[variable].hist(bins=30)
    ## Q-Q plot
    plt.subplot(1, 2, 2)
    stats.probplot(df[variable], dist="norm", plot=plt)
    plt.show()

data.hist(figsize=(20,20))
plt.show()
```



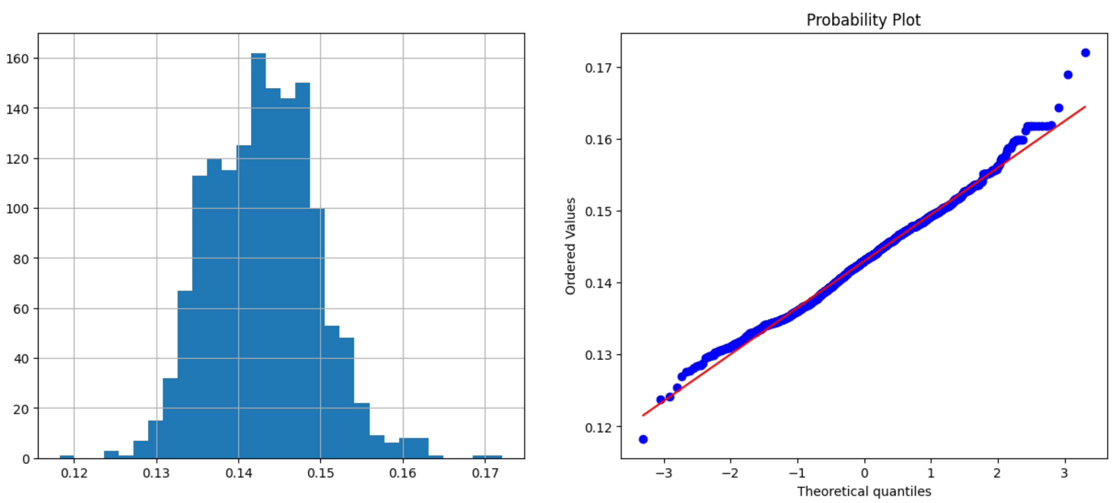

```
diagnostic_plots(data, '1stFlrSF')
```



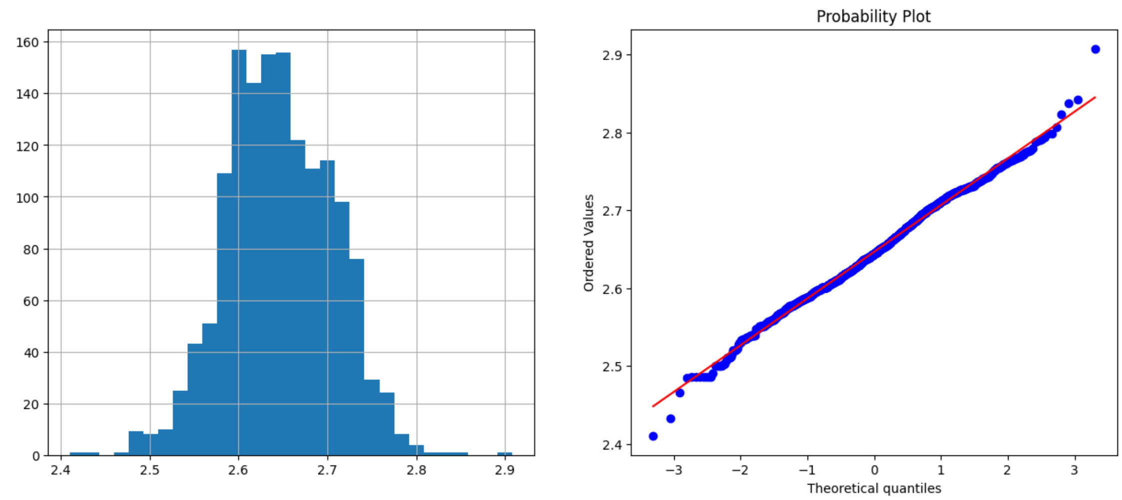
```
#Логарифмическое преобразование
data['1stFlrSF'] = np.log(data['1stFlrSF'])
diagnostic_plots(data, '1stFlrSF')
```



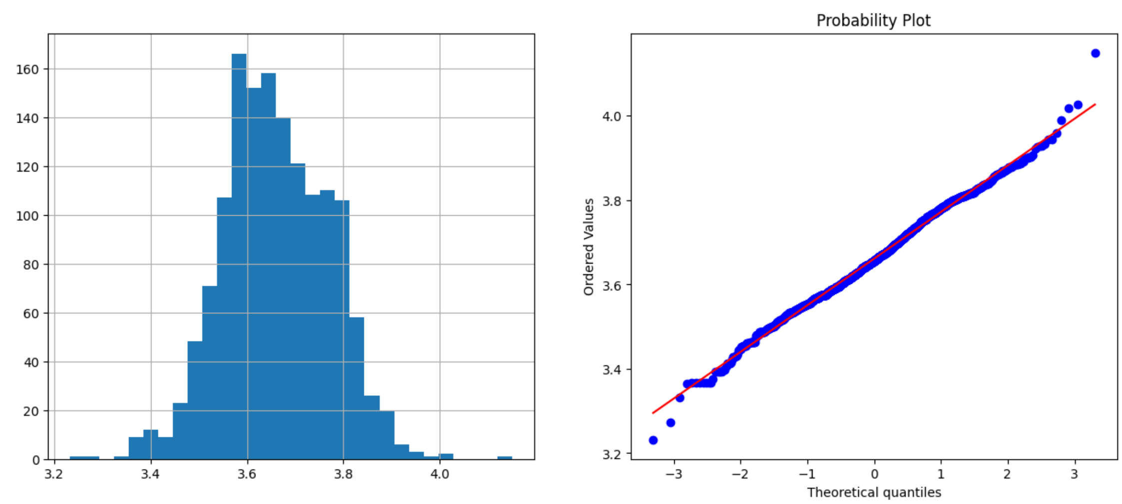

```
#Обратное преобразование
data['1stFlrSF_reciprocal'] = 1 / (data['1stFlrSF'])
diagnostic_plots(data, '1stFlrSF_reciprocal')
```



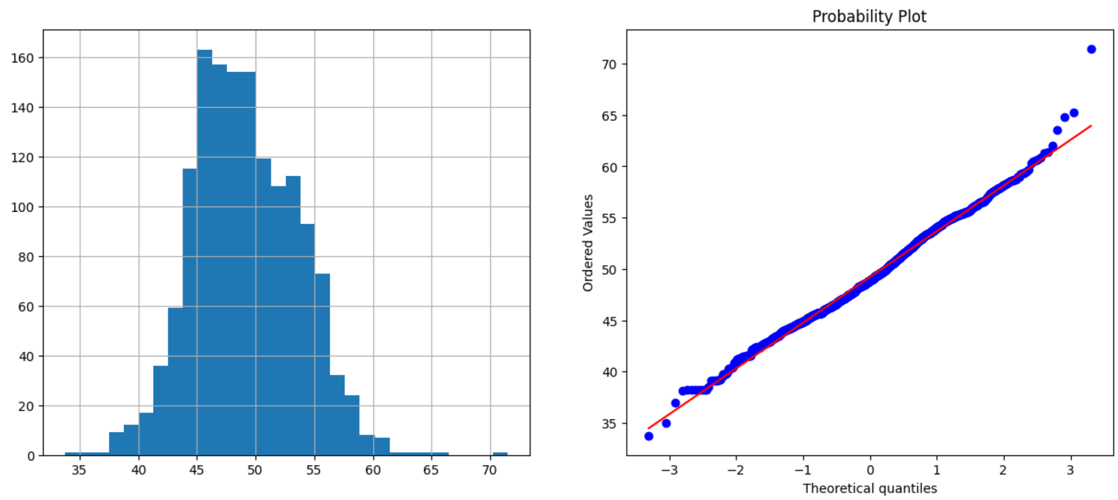

```
#Квадратный корень
data['1stFlrSF_sqr'] = data['1stFlrSF']**(1/2)
diagnostic_plots(data, '1stFlrSF_sqr')
```



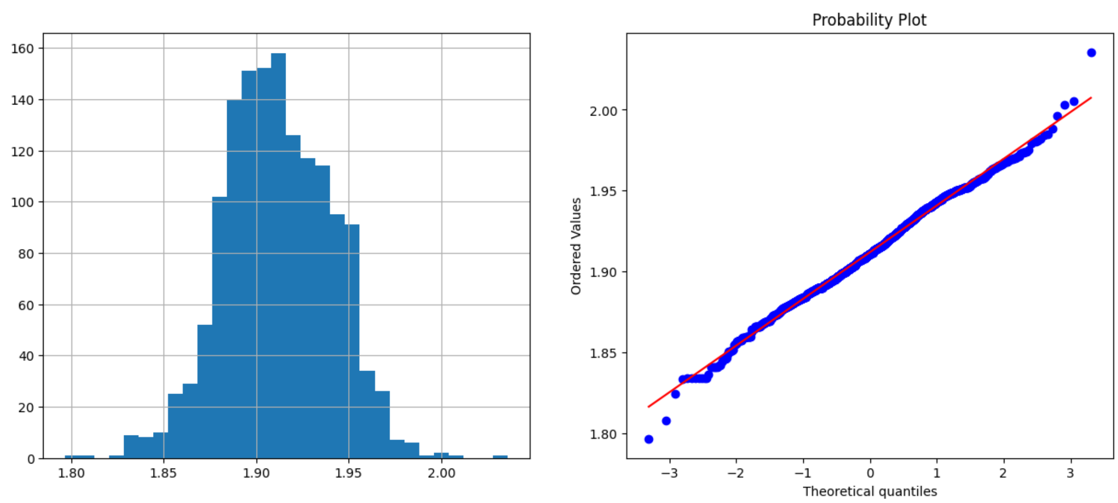

```python
#Возведение в степень
data['1stFlrSF_exp1'] = data['1stFlrSF']**(1/1.5)
diagnostic_plots(data, '1stFlrSF_exp1')
```




```python
data['1stFlrSF_exp2'] = data['1stFlrSF']**(2)
diagnostic_plots(data, '1stFlrSF_exp2')
```

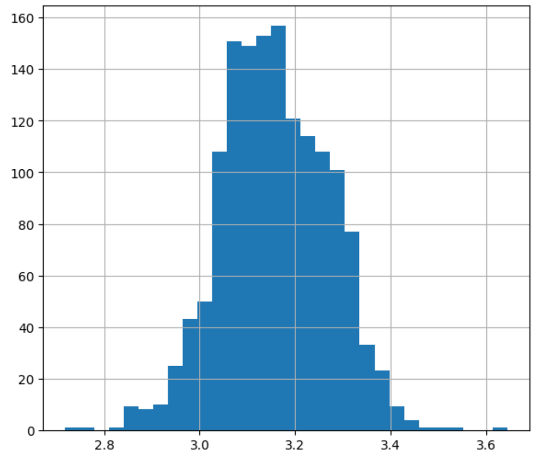



```
data['1stFlrSF_exp3'] = data['1stFlrSF']**(0.333)
diagnostic_plots(data, '1stFlrSF_exp3')
```

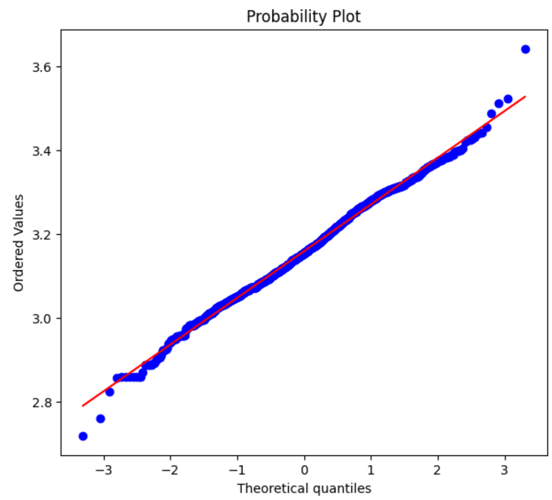



```
#Преобразованиея Бокса-Кокса
data['1stFlrSF_boxcox'], param = stats.boxcox(data['1stFlrSF'])
print('Оптимальное значение λ = {}'.format(param))
diagnostic_plots(data, '1stFlrSF_boxcox')
```

```
Оптимальное значение λ = 0.46304765872484194
```



160
140
120
100
80
60
40
20
0
2.8
3.0
3.2
3.4
3.6



Probability Plot
Ordered Values
3.6
3.4
3.2
3.0
2.8
−3
−2
−1
0
1
2
3
Theoretical quantiles