AED Pandas Matplotlib

June 14, 2020

1 Análise Exploratória de Dados

utilizando as bibliotecas Numpy, Pandas, Matplotlib e Seaborn

```
In [1]: # Importando os módulos
        import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import seaborn as sns
        %matplotlib inline
```

```
In [2]: # Carregando o dataset
        casas = pd.read_csv('Casas.csv')
        casas.head()
```

```
Out[2]:    id  Valor  Tamanho  Lote  Banheiros  Quartos    BQ   Ano  Tempo  \
        0   1  388.0    2.180     4        3.0        4  12.0  1940   -3.0
        1   2  450.0    2.054     5        3.0        4  12.0  1957   -1.3
        2   3  386.0    2.112     5        2.0        4   8.0  1955   -1.5
        3   4  350.0    1.442     6        1.0        2   2.0  1956   -1.4
        4   5  155.5    1.800     1        2.0        4   8.0  1994    2.4

           Tempo_Quad  Garagem_Tamanho Status  D7  escola  D8  D9  D10  D11  D12
        0        9.00                0    sld   0  edison   1   0    0    0    0
        1        1.69                2    sld   0  edison   1   0    0    0    0
        2        2.25                2    sld   0  edison   1   0    0    0    0
        3        1.96                1    act   1   adams   0   0    1    0    0
        4        5.76                1    sld   0   adams   0   0    1    0    0
```

```
In [3]: # Verificando o número de observações e variáveis
        casas.shape
```

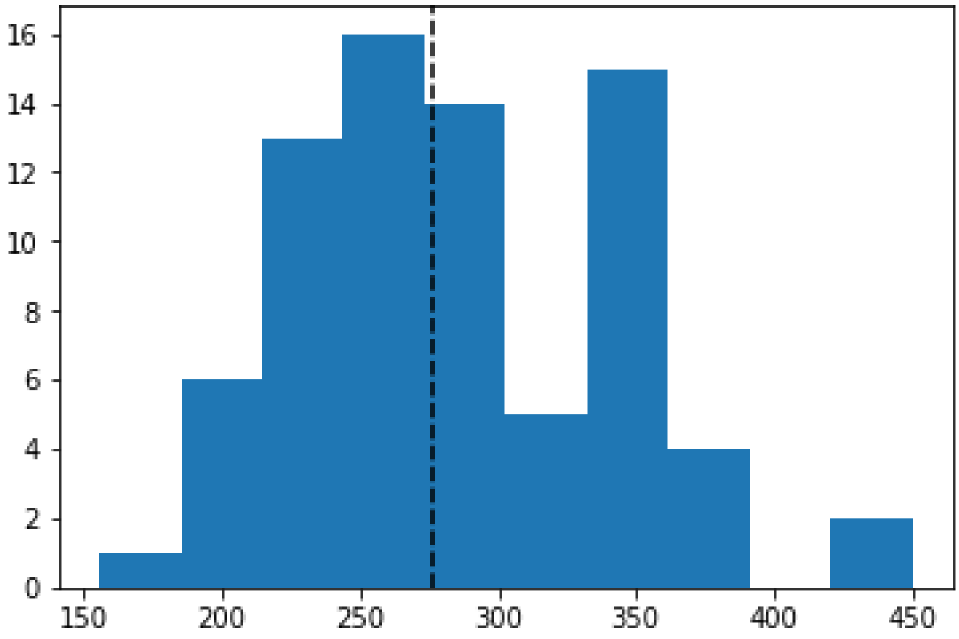
```
Out[3]: (76, 19)
```

```
In [4]: # Medidas de tendência central do atributo "Valor"
        casas['Valor'].describe()
```

```
Out[4]: count     76.000000
        mean     285.795395
        std       60.332686
        min      155.500000
        25%      242.750000
        50%      276.000000
        75%      336.750000
        max      450.000000
        Name: Valor, dtype: float64
```

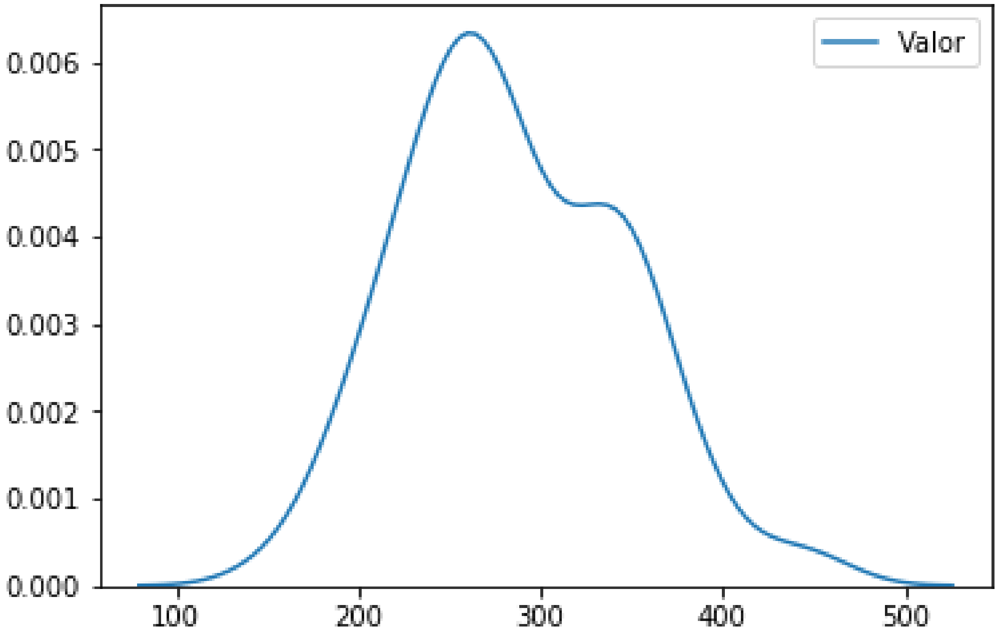
```
In [5]: # Boxplot do atributo "Valor"
        sns.boxplot(y=casas['Valor'])
```

```
Out[5]: <matplotlib.axes._subplots.AxesSubplot at 0x109e52c2b0>
```

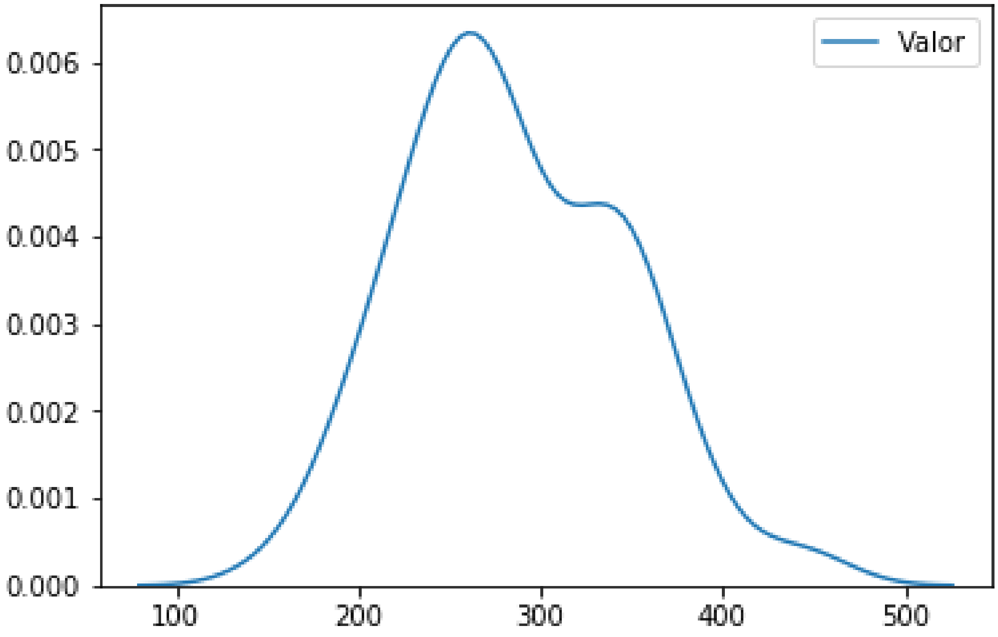


```
In [6]: plt.hist(casas['Valor'])
        plt.axvline(casas['Valor'].median(), color='k', linestyle='dashed')
```

```
Out[6]: <matplotlib.lines.Line2D at 0x109e593400>
```

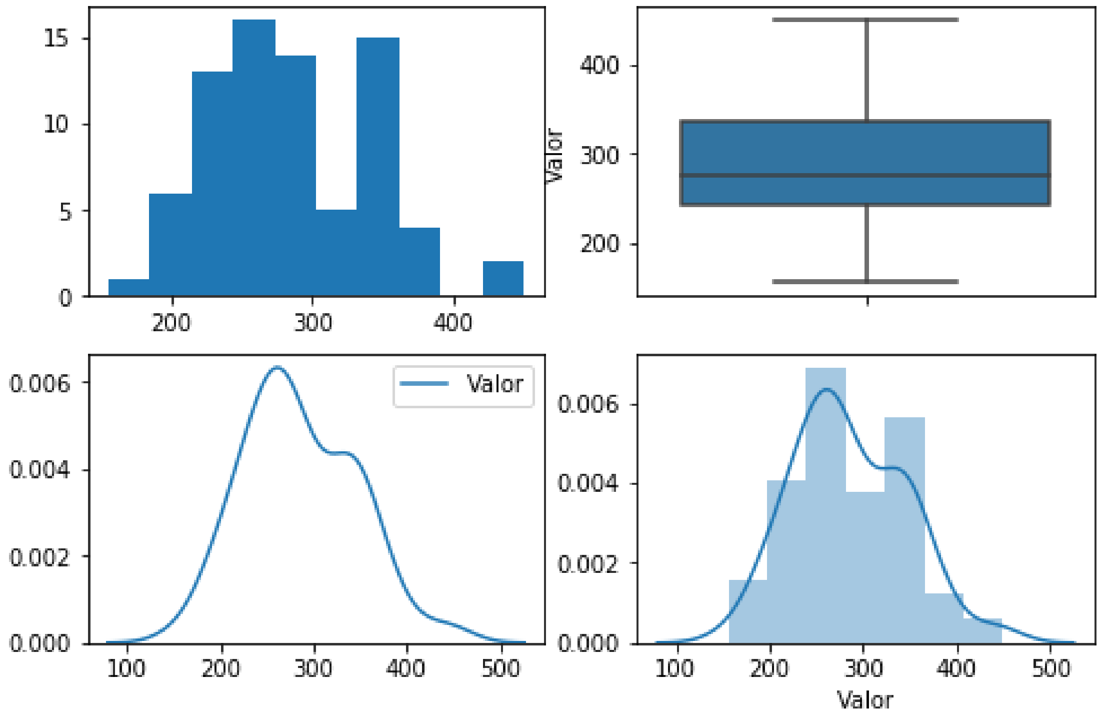


```
In [7]: # Gráfico de densidade
        sns.kdeplot(casas['Valor'])

Out[7]: <matplotlib.axes._subplots.AxesSubplot at 0x109e6624e0>
```

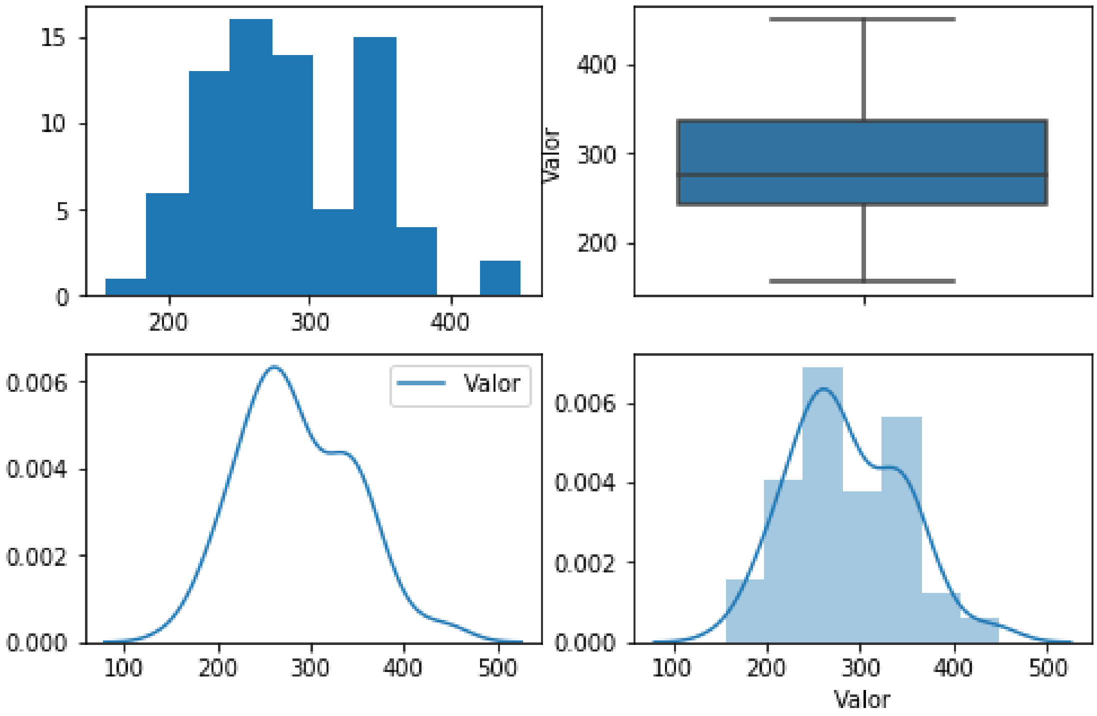


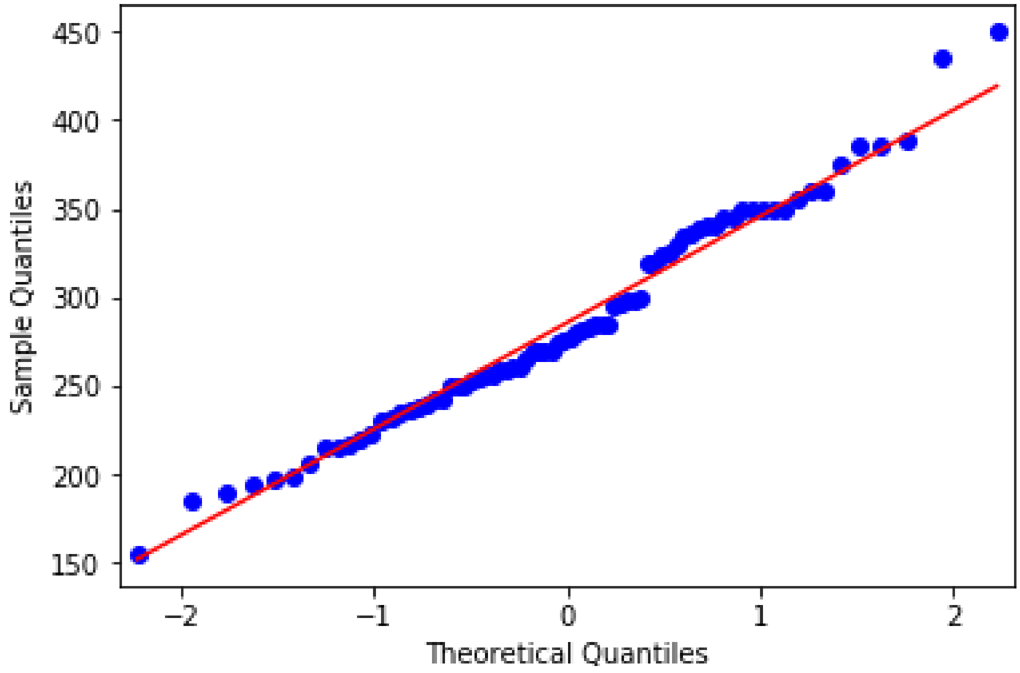
```
In [8]: # QQ-plot (verificando a distribuição dos dados)
        from statsmodels.graphics.gofplots import qqplot
```

```
In [9]: qqplot(casas['Valor'], line='s')
        plt.show()
```



```
In [14]: # Gráficos em uma mesma área de plotagem
         plt.figure(figsize=(8,8))
         plt.subplot(3,2,1)
         plt.hist(casas['Valor'])
         plt.subplot(3,2,2)
         sns.boxplot(y=casas['Valor'])
         plt.subplot(3,2,3)
         sns.kdeplot(casas['Valor'])
         plt.subplot(3,2,4)
         sns.distplot(casas['Valor'])

         qqplot(casas['Valor'], line='s')
         plt.show()
```


15
10
5
0
200
300
400
Valor
400
300
200
0.006
0.004
0.002
0.000
100
200
300
400
500
Valor
0.006
0.004
0.002
0.000
100
200
300
400
500
Valor


450
400
350
300
250
200
150
Sample Quantiles
−2
−1
0
1
2
Theoretical Quantiles

```
In [15]: # Ordenando o dataset pelo "Valor" em ordem decrescente
         casas.sort_values(by='Valor', ascending=False).head()
```

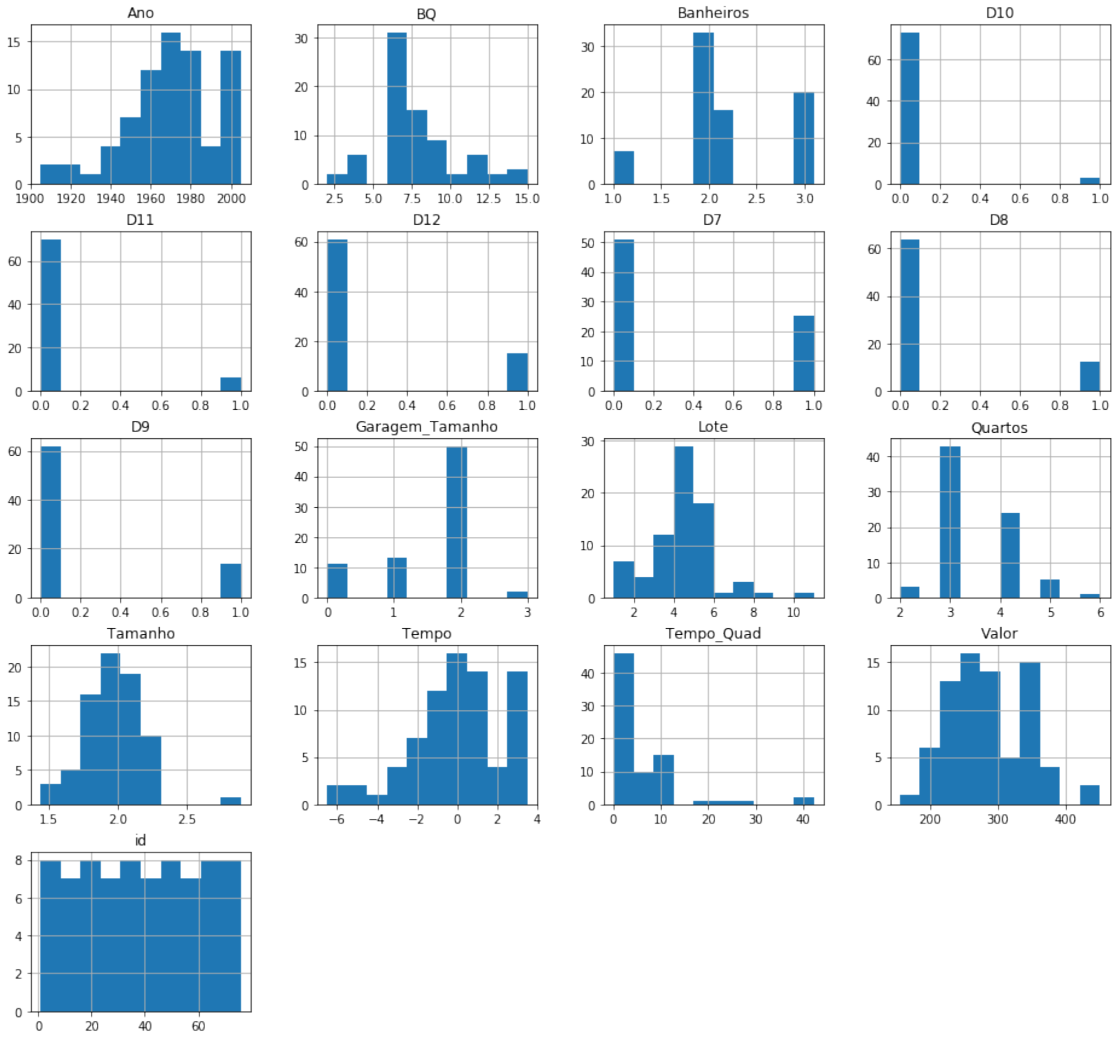
```
Out[15]:     id  Valor  Tamanho  Lote  Banheiros  Quartos    BQ   Ano  Tempo  \
         1    2  450.0    2.054     5        3.0        4  12.0  1957   -1.3
         73  74  435.0    2.253    11        2.0        3   6.0  1979    0.9
         0    1  388.0    2.180     4        3.0        4  12.0  1940   -3.0
         2    3  386.0    2.112     5        2.0        4   8.0  1955   -1.5
         51  52  385.5    1.904     4        1.1        3   3.3  1919   -5.1

             Tempo_Quad  Garagem_Tamanho Status  D7  escola  D8  D9  D10  D11  D12
         1         1.69                2    sld   0  edison   1   0    0    0    0
         73        0.81                2    sld   0    edge   0   0    0    0    0
         0         9.00                0    sld   0  edison   1   0    0    0    0
         2         2.25                2    sld   0  edison   1   0    0    0    0
         51       26.01                1    sld   0  edison   1   0    0    0    0
```

```
In [16]: # Resumo estatístico do dataset
         casas.describe()
```

```
Out[16]:              id       Valor    Tamanho       Lote  Banheiros    Quartos  \
         count  76.000000   76.000000  76.000000  76.000000  76.000000  76.000000
         mean   38.500000  285.795395   1.970395   3.986842   2.207895   3.447368
         std    22.083176   60.332686   0.212420   1.653227   0.570325   0.737468
         min     1.000000  155.500000   1.440000   1.000000   1.000000   2.000000
         25%    19.750000  242.750000   1.860750   3.000000   2.000000   3.000000
         50%    38.500000  276.000000   1.966500   4.000000   2.000000   3.000000
         75%    57.250000  336.750000   2.107500   5.000000   3.000000   4.000000
         max    76.000000  450.000000   2.896000  11.000000   3.100000   6.000000

                       BQ          Ano      Tempo  Tempo_Quad  Garagem_Tamanho  \
         count  76.000000    76.000000  76.000000   76.000000        76.000000
         mean    7.672368  1969.407895  -0.059211    5.449868         1.565789
         std     2.764663    23.492511   2.349251    8.206546         0.771760
         min     2.000000  1905.000000  -6.500000    0.000000         0.000000
         25%     6.000000  1957.750000  -1.225000    0.250000         1.000000
         50%     6.300000  1969.500000  -0.050000    1.220000         2.000000
         75%     9.000000  1980.000000   1.000000    9.000000         2.000000
         max    15.000000  2005.000000   3.500000   42.250000         3.000000

                       D7         D8         D9        D10        D11        D12
         count  76.000000  76.000000  76.000000  76.000000  76.000000  76.000000
         mean    0.328947   0.157895   0.184211   0.039474   0.078947   0.197368
         std     0.472953   0.367065   0.390232   0.196013   0.271448   0.400657
         min     0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
         25%     0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
         50%     0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
         75%     1.000000   0.000000   0.000000   0.000000   0.000000   0.000000
         max     1.000000   1.000000   1.000000   1.000000   1.000000   1.000000
```

```
In [17]: # Histogramas de todoos atributos
         casas.hist(figsize=(16,15))
```

```
Out[17]: array([[<matplotlib.axes._subplots.AxesSubplot object at 0x00000010B4D6C588>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000010B27ACA20>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000010B4E9EB38>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000010B4ECE0F0>],
                [<matplotlib.axes._subplots.AxesSubplot object at 0x00000010B4EF5668>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000010B4F1BBE0>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000010B4F4D198>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000010B4F746D8>],
                [<matplotlib.axes._subplots.AxesSubplot object at 0x00000010B4F74710>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000010B4FCC1D0>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000010B4FF4748>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000010B501ACC0>],
                [<matplotlib.axes._subplots.AxesSubplot object at 0x00000010B504E278>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000010B50757F0>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000010B509DD68>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000010B50CF320>],
                [<matplotlib.axes._subplots.AxesSubplot object at 0x00000010B50F6898>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000010B511EE10>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000010B514E3C8>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000010B5176940>]],
               dtype=object)
```



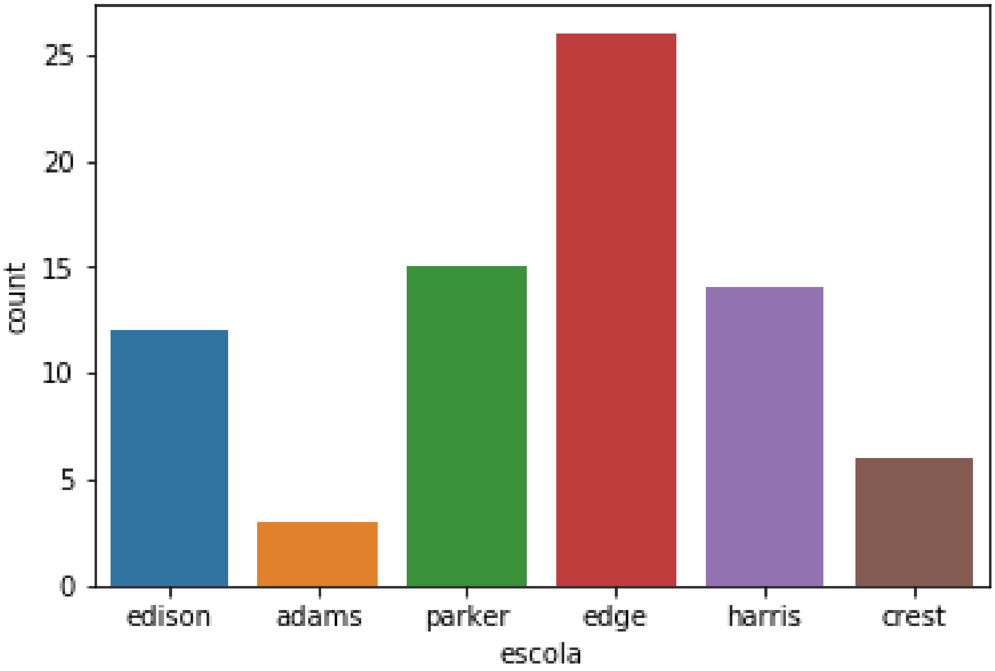
```
In [18]: # Verificando os valores NaN (Not a Number)
         casas.isnull().sum()

Out[18]: id                 0
         Valor              0
         Tamanho            0
         Lote               0
         Banheiros          0
         Quartos            0
         BQ                 0
         Ano                0
         Tempo              0
         Tempo_Quad         0
         Garagem_Tamanho    0
         Status             0
         D7                 0
         escola             0
         D8                 0
         D9                 0
         D10                0
         D11                0
         D12                0
         dtype: int64
```

```
In [19]: # Variáveis categóricas - quantidade de observações por categoria
         casas['escola'].value_counts()
```

```
Out[19]: edge      26
         parker    15
         harris    14
         edison    12
         crest      6
         adams      3
         Name: escola, dtype: int64
```

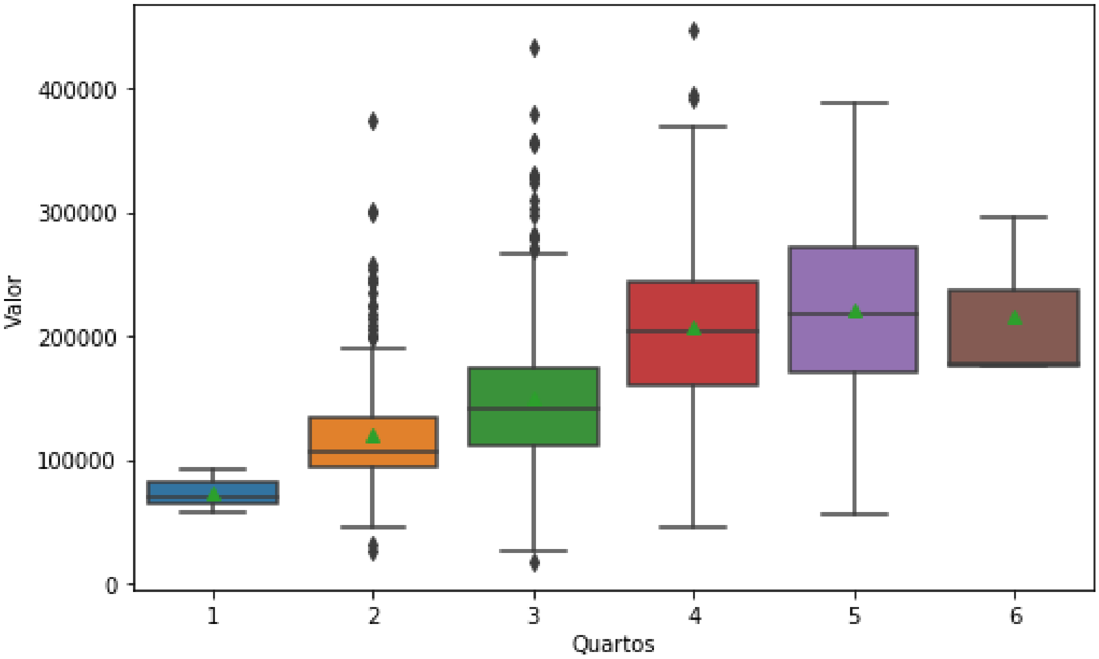
```
In [20]: # Gráfico de distribuição por categorias
         sns.countplot(x='escola', data=casas)
```

```
Out[20]: <matplotlib.axes._subplots.AxesSubplot at 0x10b58d0e10>
```



```
In [21]: # Verificando a média dos valores em relação a localização (escola)
         casas.groupby('escola')['Valor'].agg(np.mean)

Out[21]: escola
         adams     241.833333
         crest     287.816667
         edge      269.757692
         edison    327.100000
         harris    319.103571
         parker    257.446667
         Name: Valor, dtype: float64
```

```
In [22]: # Bloxplot por grupos
         sns.boxplot(x=casas['escola'], y=casas['Valor'], showmeans=True)

Out[22]: <matplotlib.axes._subplots.AxesSubplot at 0x10b4b7be80>
```



```
In [23]: # Variáveis categóricas - valores proporcionais (percentuais)
         casas['Status'].value_counts(normalize=True)

Out[23]: sld    0.500000
         act    0.328947
         pen    0.171053
         Name: Status, dtype: float64
```

```
In [24]: # Verificando a média dos valores das casas vendidas
         casas.loc[casas['Status']=='sld', 'Valor'].mean()
```

```
Out[24]: 269.21315789473687
```

```
In [25]: # Analisando a distribuição das observações entre duas variáveis (tabela de contingên
         pd.crosstab(casas['escola'], casas['Status'])
```

```
Out[25]: Status  act  pen  sld
         escola
         adams     1    0    2
         crest     5    0    1
         edge      6    6   14
         edison    1    2    9
         harris    6    3    5
         parker    6    2    7
```

```
In [45]: # Tabela de contingência com valores percentuais (por linha)
         pd.crosstab(casas['escola'], casas['Status'], normalize='index', ).mul(100)
```

```
Out[45]: Status        act        pen        sld
         escola
         adams   33.333333   0.000000  66.666667
         crest   83.333333   0.000000  16.666667
         edge    23.076923  23.076923  53.846154
         edison   8.333333  16.666667  75.000000
         harris  42.857143  21.428571  35.714286
         parker  40.000000  13.333333  46.666667
```

```
In [ ]: sns.countplot(x='escola', hue='Status', data=casas)
```

1.1 Transformando Dados

```
In [27]: df = pd.read_csv('AmesHousing.csv')
         df.head()
```

```
Out[27]:    Order        PID  MS SubClass MS Zoning  Lot Frontage  Lot Area Street  \
         0      1  526301100           20        RL         141.0     31770   Pave
         1      2  526350040           20        RH          80.0     11622   Pave
         2      3  526351010           20        RL          81.0     14267   Pave
         3      4  526353030           20        RL          93.0     11160   Pave
         4      5  527105010           60        RL          74.0     13830   Pave

           Alley Lot Shape Land Contour  ... Pool Area Pool QC  Fence Misc Feature  \
         0   NaN       IR1          Lvl  ...         0     NaN    NaN          NaN
         1   NaN       Reg          Lvl  ...         0     NaN  MnPrv          NaN
         2   NaN       IR1          Lvl  ...         0     NaN    NaN         Gar2
         3   NaN       Reg          Lvl  ...         0     NaN    NaN          NaN
         4   NaN       IR1          Lvl  ...         0     NaN  MnPrv          NaN

           Misc Val Mo Sold Yr Sold Sale Type  Sale Condition  SalePrice
         0        0       5    2010       WD           Normal     215000
```

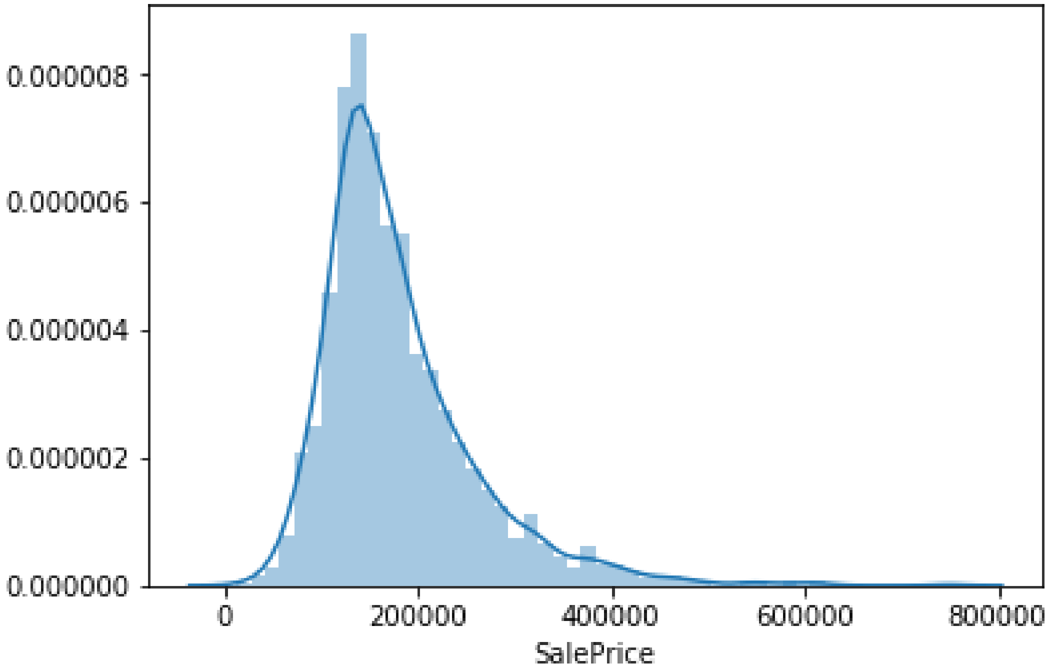
```
1         0        6    2010        WD            Normal      105000
2     12500        6    2010        WD            Normal      172000
3         0        4    2010        WD            Normal      244000
4         0        3    2010        WD            Normal      189900

[5 rows x 82 columns]
```

```
In [28]: # Verificando o número de observações e atributos do dataframe
         df.shape
```

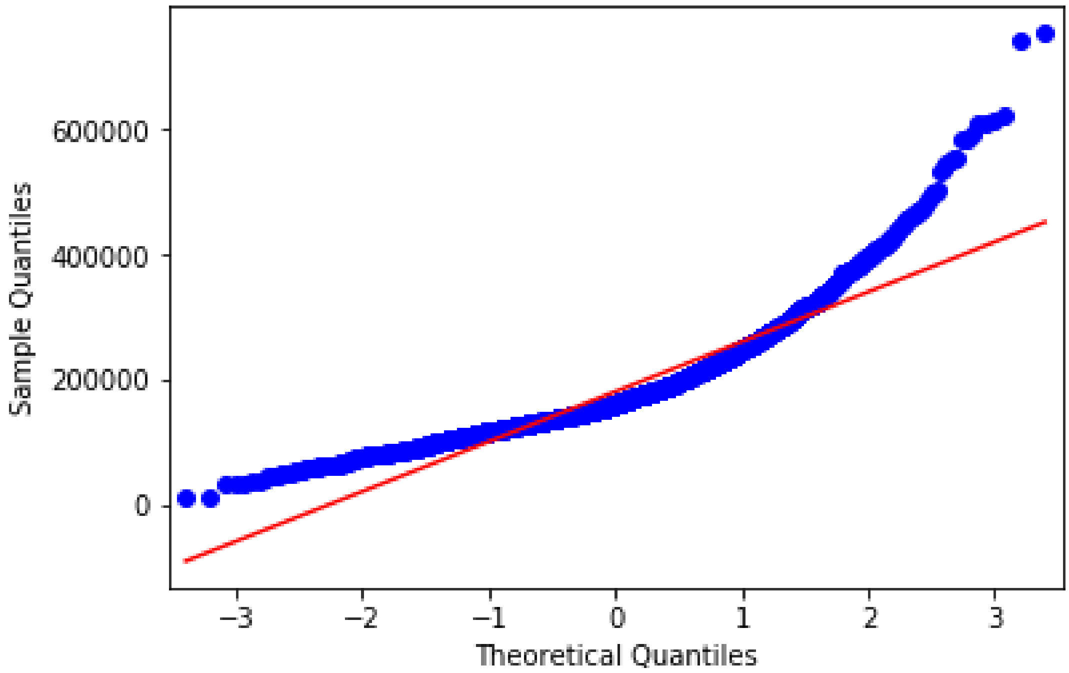
```
Out[28]: (2930, 82)
```

```
In [29]: # Gráfico de densidade e histograma do atributo "SalePrice"
         sns.distplot(df['SalePrice'])
```

```
Out[29]: <matplotlib.axes._subplots.AxesSubplot at 0x10b4b49048>
```



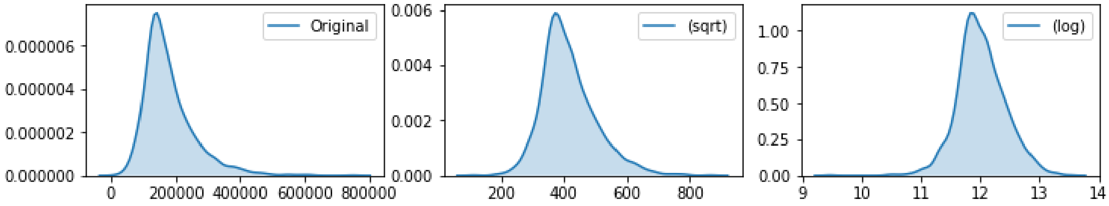
```
In [30]: # Análise de distribuição dos valores de "SalePrice"
         qqplot(df['SalePrice'], line='s')
         plt.show()
```



```
In [31]: # Transformando os valores usando a raiz quadrada e o logaritmo natural
         valor_r = np.sqrt(df['SalePrice'])
         valor_l = np.log(df['SalePrice'])
```

```
In [32]: # Gráficos de densidade
         plt.figure(figsize=(12,12))
         plt.subplot(5,3,1)
         sns.kdeplot(df['SalePrice'], shade=True, label="Original")
         plt.subplot(5,3,2)
         sns.kdeplot(valor_r, shade=True, label="(sqrt)")
         plt.subplot(5,3,3)
         sns.kdeplot(valor_l, shade=True, label="(log)" )
```

```
Out[32]: <matplotlib.axes._subplots.AxesSubplot at 0x10b4c32eb8>
```



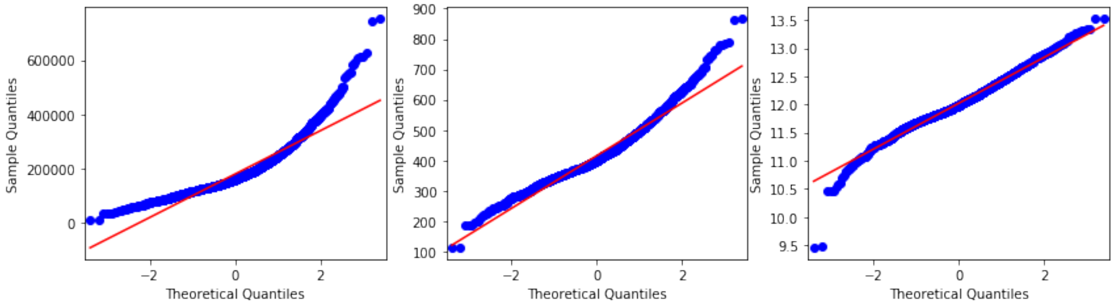
```
In [33]: print(df['SalePrice'].median())
         print(valor_r.median())
         print(valor_l.median())
```

```
160000.0
400.0
11.982929094215963
```

```
In [34]: print(df['SalePrice'].skew())
         print(valor_r.skew())
         print(valor_l.skew())
```

```
1.7435000757376466
0.8847697873897288
-0.014793439509736364
```

```
In [35]: # Gráfico qqplot
         fig = plt.figure(figsize=(14,12))
         ax = fig.add_subplot(3,3,1)
         qqplot(df['SalePrice'], line='s', ax=ax)
         ax = fig.add_subplot(3,3,2)
         qqplot(valor_r, line='s', ax=ax)
         ax = fig.add_subplot(3,3,3)
         qqplot(valor_l, line='s', ax=ax)
         plt.show()
```



```
In [ ]:
```