Untitled

November 21, 2023

0.0.1 Import the necessary libraries

1.1 Pandas is a Python library for data manipulation and analysis. 1.2 NumPy is a package that contains a multidimensional array object and several derivative ones. 1.3 Matplotlib is a Python visualization package for 2D array plots. 1.4 Seaborn is built on top of Matplotlib. It's used for exploratory data analysis and data visualization.

[24]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
df=pd.read_csv('PEP1.csv')
df.head()
```

[24]:

```
   Id  MSSubClass MSZoning  LotFrontage  LotArea Street Alley LotShape  \
0   1          60       RL         65.0     8450   Pave   NaN      Reg
1   2          20       RL         80.0     9600   Pave   NaN      Reg
2   3          60       RL         68.0    11250   Pave   NaN      IR1
3   4          70       RL         60.0     9550   Pave   NaN      IR1
4   5          60       RL         84.0    14260   Pave   NaN      IR1

  LandContour Utilities  … PoolArea PoolQC Fence MiscFeature MiscVal MoSold  \
0         Lvl    AllPub  …        0    NaN   NaN         NaN       0      2
1         Lvl    AllPub  …        0    NaN   NaN         NaN       0      5
2         Lvl    AllPub  …        0    NaN   NaN         NaN       0      9
3         Lvl    AllPub  …        0    NaN   NaN         NaN       0      2
4         Lvl    AllPub  …        0    NaN   NaN         NaN       0     12

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

[5 rows x 81 columns]
```

0.1 Read the dataset

2.1 Understand the dataset 2.2 Print the name of the columns 2.3 Print the shape of the dataframe 2.4 Check for null values 2.5 Print the unique values 2.6 Select the numerical and categorical variables

[25]:
```
df.columns
```

[25]:
```
Index(['Id', 'MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea', 'Street',
       'Alley', 'LotShape', 'LandContour', 'Utilities', 'LotConfig',
       'LandSlope', 'Neighborhood', 'Condition1', 'Condition2', 'BldgType',
       'HouseStyle', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd',
       'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType',
       'MasVnrArea', 'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual',
       'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinSF1',
       'BsmtFinType2', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'Heating',
       'HeatingQC', 'CentralAir', 'Electrical', '1stFlrSF', '2ndFlrSF',
       'LowQualFinSF', 'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath',
       'HalfBath', 'Bedroom', 'Kitchen', 'KitchenQual', 'TotRmsAbvGrd',
       'Functional', 'Fireplaces', 'FireplaceQu', 'GarageType', 'GarageYrBlt',
       'GarageFinish', 'GarageCars', 'GarageArea', 'GarageQual', 'GarageCond',
       'PavedDrive', 'WoodDeckSF', 'OpenPorchSF', 'EnclosedPorch', '3SsnPorch',
       'ScreenPorch', 'PoolArea', 'PoolQC', 'Fence', 'MiscFeature', 'MiscVal',
       'MoSold', 'YrSold', 'SaleType', 'SaleCondition', 'SalePrice'],
      dtype='object')
```

[8]:
```
df.shape
```

[8]:
```
(1460, 81)
```

[12]:
```
df.isna().sum()
```

[12]:
```
Id                 0
MSSubClass         0
MSZoning           0
LotFrontage      259
LotArea            0
                ...
MoSold             0
YrSold             0
SaleType           0
SaleCondition      0
SalePrice          0
Length: 81, dtype: int64
```

[24]:
```
numerical_feature_columns = list(df._get_numeric_data().columns)
numerical_feature_columns
```

[24]:

```
['Id',
 'MSSubClass',
 'LotFrontage',
 'LotArea',
 'OverallQual',
 'OverallCond',
 'YearBuilt',
 'YearRemodAdd',
 'MasVnrArea',
 'BsmtFinSF1',
 'BsmtFinSF2',
 'BsmtUnfSF',
 'TotalBsmtSF',
 '1stFlrSF',
 '2ndFlrSF',
 'LowQualFinSF',
 'GrLivArea',
 'BsmtFullBath',
 'BsmtHalfBath',
 'FullBath',
 'HalfBath',
 'Bedroom',
 'Kitchen',
 'TotRmsAbvGrd',
 'Fireplaces',
 'GarageYrBlt',
 'GarageCars',
 'GarageArea',
 'WoodDeckSF',
 'OpenPorchSF',
 'EnclosedPorch',
 '3SsnPorch',
 'ScreenPorch',
 'PoolArea',
 'MiscVal',
 'MoSold',
 'YrSold',
 'SalePrice']
```

[28]:

```
categorical_feature_columns = list(set(df.columns) - set(df._get_numeric_data().
 ↪columns))
categorical_feature_columns
```

[28]:

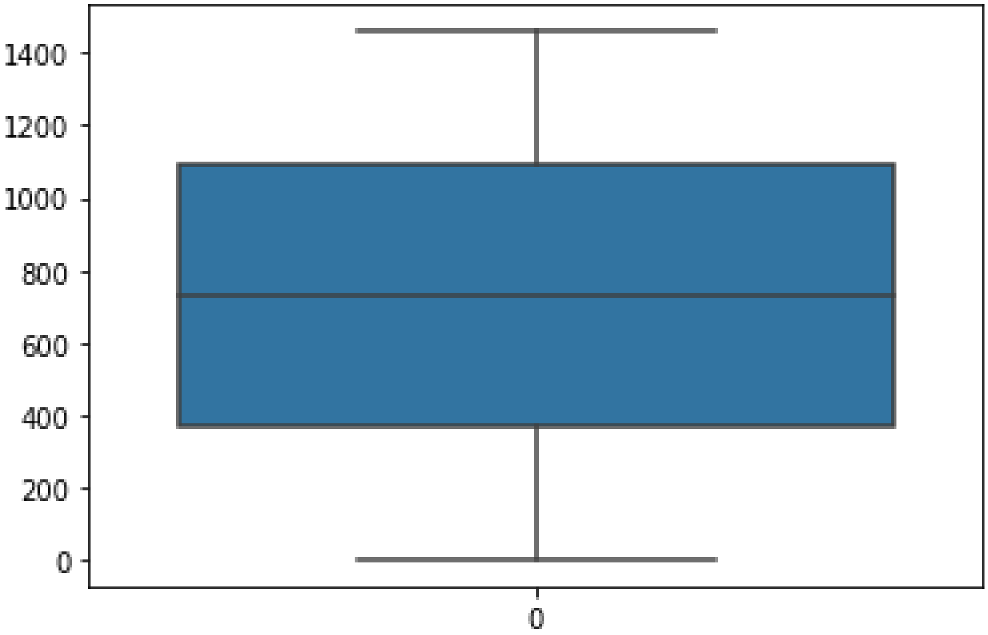
```
['MasVnrType',
 'RoofStyle',
 'MiscFeature',
 'RoofMatl',
 'ExterQual',
 'LotConfig',
 'GarageFinish',
 'LandSlope',
 'BsmtCond',
 'Condition2',
 'Condition1',
 'Exterior1st',
 'KitchenQual',
 'BsmtExposure',
 'PavedDrive',
 'Exterior2nd',
 'Neighborhood',
 'Functional',
 'HouseStyle',
 'CentralAir',
 'GarageType',
 'Fence',
 'Foundation',
 'HeatingQC',
 'BsmtFinType2',
 'Street',
 'Heating',
 'Alley',
 'MSZoning',
 'BldgType',
 'PoolQC',
 'Electrical',
 'FireplaceQu',
 'LotShape',
 'GarageCond',
 'LandContour',
 'BsmtQual',
 'BsmtFinType1',
 'ExterCond',
 'SaleType',
 'GarageQual',
 'Utilities',
 'SaleCondition']
```

0.1.1 Descriptive stats and EDA

3.1 EDA of numerical variables 3.2 Missing value treatment 3.3 Identify the skewness and distribution 3.4 Identify significant variables using a correlation matrix 3.5 Pair plot for distribution and density

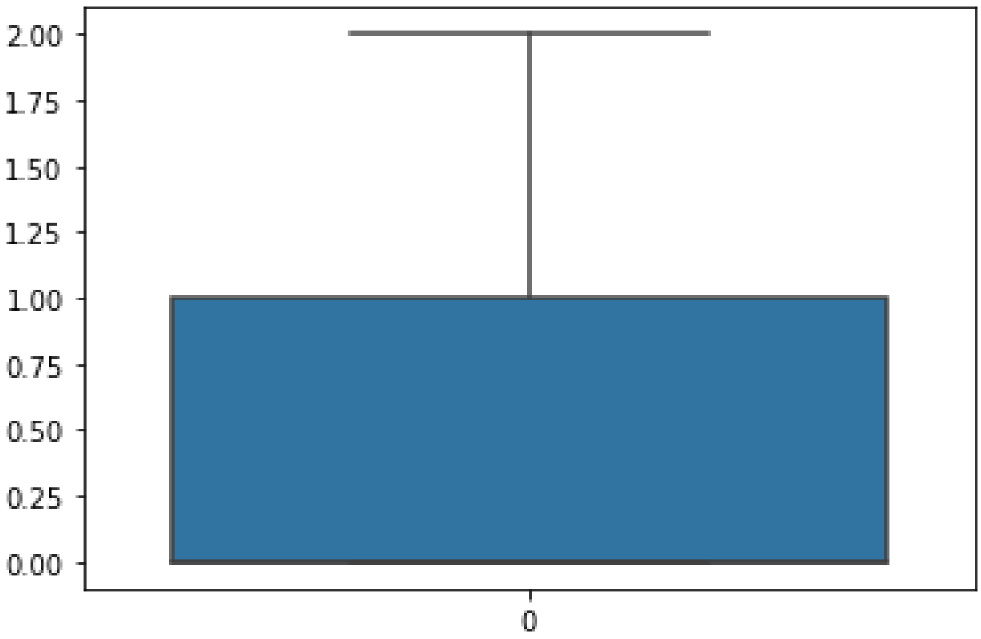
[26]:
```
import seaborn as sns
sns.boxplot(df['Id'])
```

[26]:
```
<AxesSubplot: >
```



[28]:
```
import seaborn as sns
sns.boxplot(df['HalfBath'])
```

[28]:
```
<AxesSubplot: >
```



[29]:
```
df.describe()
```

[29]:
```
                Id    MSSubClass  LotFrontage        LotArea  OverallQual  \
count  1460.000000  1460.000000  1201.000000    1460.000000  1460.000000
mean    730.500000    56.897260    70.049958   10516.828082     6.099315
std     421.610009    42.300571    24.284752    9981.264932     1.382997
min       1.000000    20.000000    21.000000    1300.000000     1.000000
25%     365.750000    20.000000    59.000000    7553.500000     5.000000
50%     730.500000    50.000000    69.000000    9478.500000     6.000000
75%    1095.250000    70.000000    80.000000   11601.500000     7.000000
max    1460.000000   190.000000   313.000000  215245.000000    10.000000

       OverallCond    YearBuilt  YearRemodAdd   MasVnrArea   BsmtFinSF1  …  \
count  1460.000000  1460.000000   1460.000000  1452.000000  1460.000000  …
mean      5.575342  1971.267808   1984.865753   103.685262   443.639726  …
std       1.112799    30.202904     20.645407   181.066207   456.098091  …
min       1.000000  1872.000000   1950.000000     0.000000     0.000000  …
25%       5.000000  1954.000000   1967.000000     0.000000     0.000000  …
50%       5.000000  1973.000000   1994.000000     0.000000   383.500000  …
75%       6.000000  2000.000000   2004.000000   166.000000   712.250000  …
max       9.000000  2010.000000   2010.000000  1600.000000  5644.000000  …

        WoodDeckSF  OpenPorchSF  EnclosedPorch    3SsnPorch  ScreenPorch  \
count  1460.000000  1460.000000    1460.000000  1460.000000  1460.000000
mean     94.244521    46.660274      21.954110     3.409589    15.060959
std     125.338794    66.256028      61.119149    29.317331    55.757415
```

```
min       0.000000    0.000000      0.000000    0.000000    0.000000
25%       0.000000    0.000000      0.000000    0.000000    0.000000
50%       0.000000   25.000000      0.000000    0.000000    0.000000
75%     168.000000   68.000000      0.000000    0.000000    0.000000
max     857.000000  547.000000    552.000000  508.000000  480.000000

          PoolArea      MiscVal       MoSold       YrSold      SalePrice
count  1460.000000  1460.000000  1460.000000  1460.000000    1460.000000
mean      2.758904    43.489041     6.321918  2007.815753  180921.195890
std      40.177307   496.123024     2.703626     1.328095   79442.502883
min       0.000000     0.000000     1.000000  2006.000000   34900.000000
25%       0.000000     0.000000     5.000000  2007.000000  129975.000000
50%       0.000000     0.000000     6.000000  2008.000000  163000.000000
75%       0.000000     0.000000     8.000000  2009.000000  214000.000000
max     738.000000 15500.000000    12.000000  2010.000000  755000.000000

[8 rows x 38 columns]
```

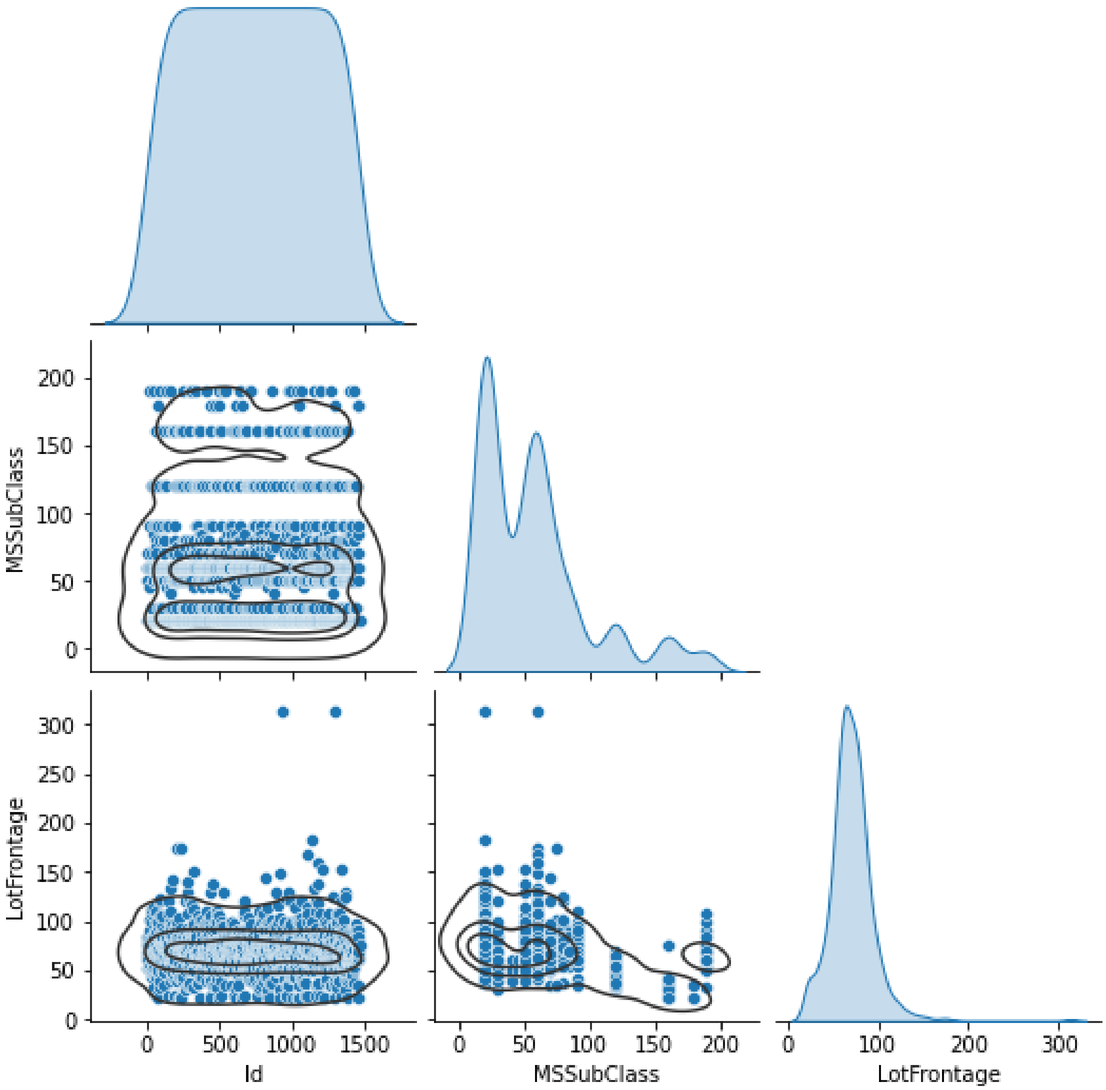
[33]:
```
df.skew(axis=0,skipna=True)
```

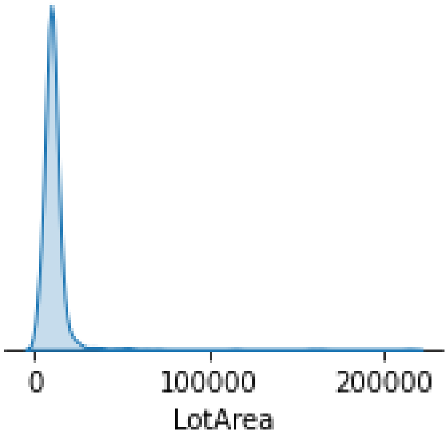
```
/tmp/ipykernel_71/4266299306.py:1: FutureWarning: The default value of
numeric_only in DataFrame.skew is deprecated. In a future version, it will
default to False. In addition, specifying 'numeric_only=None' is deprecated.
Select only valid columns or specify the value of numeric_only to silence this
warning.
  df.skew(axis=0,skipna=True)
```

[33]:
```
Id                0.000000
MSSubClass        1.407657
LotFrontage       2.163569
LotArea          12.207688
OverallQual       0.216944
OverallCond       0.693067
YearBuilt        -0.613461
YearRemodAdd     -0.503562
MasVnrArea        2.669084
BsmtFinSF1        1.685503
BsmtFinSF2        4.255261
BsmtUnfSF         0.920268
TotalBsmtSF       1.524255
1stFlrSF          1.376757
2ndFlrSF          0.813030
LowQualFinSF      9.011341
GrLivArea         1.366560
BsmtFullBath      0.596067
BsmtHalfBath      4.103403
FullBath          0.036562
```

```
HalfBath          0.675897
Bedroom           0.211790
Kitchen           4.488397
TotRmsAbvGrd      0.676341
Fireplaces        0.649565
GarageYrBlt      -0.649415
GarageCars       -0.342549
GarageArea        0.179981
WoodDeckSF        1.541376
OpenPorchSF       2.364342
EnclosedPorch     3.089872
3SsnPorch        10.304342
ScreenPorch       4.122214
PoolArea         14.828374
MiscVal          24.476794
MoSold            0.212053
YrSold            0.096269
SalePrice         1.882876
dtype: float64
```

```
[69]: for i in range(0,len(df.columns),4):
          g=sns.pairplot(df.iloc[:,i:i+4],diag_kind="kde", corner=True)
          g.map_lower(sns.kdeplot, levels=4, color=".2")
          plt.show()
```


MSSubClass
0
50
100
150
200
LotFrontage
0
50
100
150
200
250
300
0
500
1000
1500
Id
0
50
100
150
200
MSSubClass
0
100
200
300
LotFrontage


0
100000
200000
LotArea

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
/tmp/ipykernel_72/1145027026.py in <cell line: 1>()
      1 for i in range(0,len(df.columns),4):
----> 2     g=sns.pairplot(df.iloc[:,i:i+4],diag_kind="kde", corner=True)
      3     g.map_lower(sns.kdeplot, levels=4, color=".2")
      4     plt.show()

/usr/local/lib/python3.10/site-packages/seaborn/axisgrid.py in pairplot(data, hue, hue_order, palette, vars, x_vars, y_vars, kind, diag_kind, markers, height, aspect, corner, dropna, plot_kws, diag_kws, grid_kws, size)
   2112     # Set up the PairGrid
   2113     grid_kws.setdefault("diag_sharey", diag_kind == "hist")
-> 2114     grid = PairGrid(data, vars=vars, x_vars=x_vars, y_vars=y_vars, hue=hue,
   2115                     hue_order=hue_order, palette=palette, corner=corner
   2116                     height=height, aspect=aspect, dropna=dropna, **grid_kws)

/usr/local/lib/python3.10/site-packages/seaborn/axisgrid.py in __init__(self, data, hue, vars, x_vars, y_vars, hue_order, palette, hue_kws, corner, diag_sharey, height, aspect, layout_pad, despine, dropna)
   1264
   1265         if not x_vars:
-> 1266             raise ValueError("No variables found for grid columns.")
   1267         if not y_vars:
   1268             raise ValueError("No variables found for grid rows.")

ValueError: No variables found for grid columns.
```

[65]:

```
corr = df.corr()
corr.style.background_gradient(cmap='coolwarm').set_precision(2)
```

```
/tmp/ipykernel_72/2001914525.py:1: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it will
default to False. Select only valid columns or specify the value of numeric_only
to silence this warning.
  corr = df.corr()
/tmp/ipykernel_72/2001914525.py:2: FutureWarning: this method is deprecated in
favour of `Styler.format(precision=..)`
  corr.style.background_gradient(cmap='coolwarm').set_precision(2)
```

[65]:

```
<pandas.io.formats.style.Styler at 0x7f8a1d516f80>
```

[ ]:

```

```