

DATA CLEANING

(It's all about data Clean Up)



<https://www.linkedin.com/in/nibedita-sarkar/>



DATA CLEANING

(It's all about data Clean Up)

Q) What is missing Value ?

The values or data that is not stored (or not present) for some variable/s in the given dataset.

Q) What are the methods to handle missing values?

- 1) Ignore missing values row/ Delete row
- 2) Fill missing value manually
- 3) Global Constant
- 4) Measure of central tendency (Mean, Median, Mode)
- 5) Measure of central tendency for each class
- 6) Most probable value (ML Algorithms)

1) Ignore missing values row/ Delete row :

In this approach, you simply ignore/ delete the rows or observations that contain missing values. This method is suitable when the missing values are minimal and do not significantly impact the analysis. However, if a large proportion of the data is missing, this method may result in the loss of valuable information.

If the dataset contains missing values equal to or less than 5%, we can consider deleting the corresponding rows.

If the columns contain more than 20% or 30% missing values, we can consider deleting those columns.



DATA CLEANING

(It's all about data Clean Up)

2) Fill missing value manually :

For small datasets, you can manually examine the missing values and fill them based on your domain knowledge or by consulting subject matter experts. While this method ensures that the filled values are accurate, it can be time-consuming and subjective.

3) Global Constant :

In this approach, missing values are replaced with a global constant value. For example, you can replace missing numerical values with zero or missing categorical values with a specific category label. However, this method may introduce bias or distort the data if the missing values are not truly missing at random.

4) Measure of central tendency (Mean, Median, Mode) :

Missing values in numerical variables can be replaced by a measure of central tendency such as the mean, median, or mode. The mean is commonly used if the variable follows a normal distribution, while the median is preferred if the variable has outliers. The mode is suitable for categorical variables.

5) Measure of central tendency for each class :

If the data has categorical variables and missing values are present within specific classes or groups, you can replace the missing values with the measure of central tendency (mean, median, or mode) calculated separately for each class. This approach helps preserve the within-class characteristics and reduces bias.



DATA CLEANING

(It's all about data Clean Up)

6) Most probable value (ML Algorithms) :

Machine learning algorithms can be used to predict the missing values based on the available data. You can train a model on the non-missing values and then use it to predict the missing values. This method utilizes the relationships and patterns in the data to estimate the missing values. However, it assumes that the missing values are predictable based on other variables.

Note :

Among the six methods mentioned, I have written a process to handle missing values for the first five methods. The sixth method involves handling missing values using a machine learning algorithm, which I will share at a later time.

Please consider following @Nibedita Sarkar on LinkedIn for more updates and information. You can find the link to her LinkedIn profile here:

[<https://www.linkedin.com/in/nibedita-sarkar/>]



<https://www.linkedin.com/in/nibedita-sarkar/>

tation-by-deleting-row-and-columns

June 11, 2023

1 Data Cleaning :

1.1 Missing value imputation by Deleting Rows and Columns :

Importing necessary libraries

```
[1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

data (Original) :

```
[3]: data = pd.read_csv("train.csv") # Load dataset
```

```
[5]: data.head(5) # Checking first 5 rows from the DataFrame.
```

```
[5]:
```

	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]

```
[4]: data.shape # Checking the shape of the dataset, dataset contains 1460 rows and
      ↪81 columns
```

[4]: (1460, 81)

When we check the shape of the data using the command “data.shape”, we can observe that the DataFrame consists of 1460 rows and 81 columns. However, when we examine the first five rows of the DataFrame using the command “data.head(5)”, it doesn’t display all 81 columns. As a result, we utilize the following code, specifically “pd.set_option()”, to address this issue.

```
[6]: pd.set_option('display.max_columns', None)
      pd.set_option('display.max_rows', None)
```

pd.set_option(‘display.max_columns’, None): This line sets the maximum number of columns to be displayed in the output to None, which means there is no limit. As a result, all columns in a DataFrame will be shown when you print or display it.

pd.set_option(‘display.max_rows’, None): This line sets the maximum number of rows to be displayed in the output to None, removing any limit. As a result, all rows in a DataFrame will be shown when you print or display it.

```
[7]: data.head(2) # The code `data.head(2)` displays the first two rows of the
      ↪dataset, showing all 81 columns.
```

```
[7]:   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

      LandContour  Utilities  LotConfig  LandSlope  Neighborhood  Condition1  \
0             Lvl     AllPub     Inside         Gtl         CollgCr         Norm
1             Lvl     AllPub         FR2         Gtl         Veenker         Feedr

      Condition2  BldgType  HouseStyle  OverallQual  OverallCond  YearBuilt  \
0             Norm     1Fam     2Story             7             5         2003
1             Norm     1Fam     1Story             6             8         1976

      YearRemodAdd  RoofStyle  RoofMatl  Exterior1st  Exterior2nd  MasVnrType  \
0             2003     Gable  CompShg     VinylSd     VinylSd     BrkFace
1             1976     Gable  CompShg     MetalSd     MetalSd         None

      MasVnrArea  ExterQual  ExterCond  Foundation  BsmtQual  BsmtCond  BsmtExposure  \
0             196.0         Gd         TA         PConc         Gd         TA         No
1              0.0         TA         TA         CBlock         Gd         TA         Gd

      BsmtFinType1  BsmtFinSF1  BsmtFinType2  BsmtFinSF2  BsmtUnfSF  TotalBsmtSF  \
```

```

0      GLQ      706      Unf      0      150      856
1      ALQ      978      Unf      0      284      1262

  Heating HeatingQC CentralAir Electrical 1stFlrSF 2ndFlrSF LowQualFinSF \
0  GasA      Ex      Y      SBrkr      856      854      0
1  GasA      Ex      Y      SBrkr      1262      0      0

  GrLivArea BsmtFullBath BsmtHalfBath FullBath HalfBath BedroomAbvGr \
0      1710      1      0      2      1      3
1      1262      0      1      2      0      3

  KitchenAbvGr KitchenQual TotRmsAbvGrd Functional Fireplaces FireplaceQu \
0      1      Gd      8      Typ      0      NaN
1      1      TA      6      Typ      1      TA

  GarageType GarageYrBlt GarageFinish GarageCars GarageArea GarageQual \
0  Attchd      2003.0      RFn      2      548      TA
1  Attchd      1976.0      RFn      2      460      TA

  GarageCond PavedDrive WoodDeckSF OpenPorchSF EnclosedPorch 3SsnPorch \
0      TA      Y      0      61      0      0
1      TA      Y      298      0      0      0

  ScreenPorch PoolArea PoolQC Fence MiscFeature MiscVal MoSold YrSold \
0      0      0      NaN      NaN      NaN      0      2      2008
1      0      0      NaN      NaN      NaN      0      5      2007

  SaleType SaleCondition SalePrice
0      WD      Normal      208500
1      WD      Normal      181500

```

```
[8]: data.tail(2) # The code `data.tail(2)` displays the last two rows of the
↳ dataset.
```

```
[8]:      Id MSSubClass MSZoning LotFrontage LotArea Street Alley LotShape \
1458 1459      20      RL      68.0      9717      Pave      NaN      Reg
1459 1460      20      RL      75.0      9937      Pave      NaN      Reg

  LandContour Utilities LotConfig LandSlope Neighborhood Condition1 \
1458      Lvl      AllPub      Inside      Gtl      NNames      Norm
1459      Lvl      AllPub      Inside      Gtl      Edwards      Norm

  Condition2 BldgType HouseStyle OverallQual OverallCond YearBuilt \
1458      Norm      1Fam      1Story      5      6      1950
1459      Norm      1Fam      1Story      5      6      1965

  YearRemodAdd RoofStyle RoofMatl Exterior1st Exterior2nd MasVnrType \

```

1458	1996	Hip	CompShg	MetalSd	MetalSd	None			
1459	1965	Gable	CompShg	HdBoard	HdBoard	None			
	MasVnrArea	ExterQual	ExterCond	Foundation	BsmtQual	BsmtCond	\		
1458	0.0	TA	TA	CBlock	TA	TA			
1459	0.0	Gd	TA	CBlock	TA	TA			
	BsmtExposure	BsmtFinType1	BsmtFinSF1	BsmtFinType2	BsmtFinSF2	\			
1458	Mn	GLQ	49	Rec	1029				
1459	No	BLQ	830	LwQ	290				
	BsmtUnfSF	TotalBsmtSF	Heating	HeatingQC	CentralAir	Electrical	\		
1458	0	1078	GasA	Gd	Y	FuseA			
1459	136	1256	GasA	Gd	Y	SBrkr			
	1stFlrSF	2ndFlrSF	LowQualFinSF	GrLivArea	BsmtFullBath	BsmtHalfBath	\		
1458	1078	0	0	1078	1	0			
1459	1256	0	0	1256	1	0			
	FullBath	HalfBath	BedroomAbvGr	KitchenAbvGr	KitchenQual	\			
1458	1	0	2	1	Gd				
1459	1	1	3	1	TA				
	TotRmsAbvGrd	Functional	Fireplaces	FireplaceQu	GarageType	GarageYrBlt	\		
1458	5	Typ	0	NaN	Attchd	1950.0			
1459	6	Typ	0	NaN	Attchd	1965.0			
	GarageFinish	GarageCars	GarageArea	GarageQual	GarageCond	PavedDrive	\		
1458	Unf	1	240	TA	TA	Y			
1459	Fin	1	276	TA	TA	Y			
	WoodDeckSF	OpenPorchSF	EnclosedPorch	3SsnPorch	ScreenPorch	\			
1458	366	0	112	0	0				
1459	736	68	0	0	0				
	PoolArea	PoolQC	Fence	MiscFeature	MiscVal	MoSold	YrSold	SaleType	\
1458	0	NaN	NaN	NaN	0	4	2010	WD	
1459	0	NaN	NaN	NaN	0	6	2008	WD	
	SaleCondition	SalePrice							
1458	Normal	142125							
1459	Normal	147500							

```
[9]: data.info()
```

```
....
```


The `data.info()` function provides a concise summary of the dataset, including information about the number of rows, columns, data types, memory usage and missing values.

'''

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 81 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Id                     1460 non-null   int64
1   MSSubClass             1460 non-null   int64
2   MSZoning               1460 non-null   object
3   LotFrontage           1201 non-null   float64
4   LotArea               1460 non-null   int64
5   Street                1460 non-null   object
6   Alley                 91 non-null     object
7   LotShape              1460 non-null   object
8   LandContour           1460 non-null   object
9   Utilities             1460 non-null   object
10  LotConfig             1460 non-null   object
11  LandSlope             1460 non-null   object
12  Neighborhood          1460 non-null   object
13  Condition1            1460 non-null   object
14  Condition2            1460 non-null   object
15  BldgType              1460 non-null   object
16  HouseStyle            1460 non-null   object
17  OverallQual           1460 non-null   int64
18  OverallCond           1460 non-null   int64
19  YearBuilt             1460 non-null   int64
20  YearRemodAdd          1460 non-null   int64
21  RoofStyle             1460 non-null   object
22  RoofMatl              1460 non-null   object
23  Exterior1st           1460 non-null   object
24  Exterior2nd           1460 non-null   object
25  MasVnrType            1452 non-null   object
26  MasVnrArea            1452 non-null   float64
27  ExterQual             1460 non-null   object
28  ExterCond             1460 non-null   object
29  Foundation            1460 non-null   object
30  BsmtQual              1423 non-null   object
31  BsmtCond              1423 non-null   object
32  BsmtExposure          1422 non-null   object
33  BsmtFinType1          1423 non-null   object
34  BsmtFinSF1            1460 non-null   int64
35  BsmtFinType2          1422 non-null   object
```

36	BsmtFinSF2	1460	non-null	int64
37	BsmtUnfSF	1460	non-null	int64
38	TotalBsmtSF	1460	non-null	int64
39	Heating	1460	non-null	object
40	HeatingQC	1460	non-null	object
41	CentralAir	1460	non-null	object
42	Electrical	1459	non-null	object
43	1stFlrSF	1460	non-null	int64
44	2ndFlrSF	1460	non-null	int64
45	LowQualFinSF	1460	non-null	int64
46	GrLivArea	1460	non-null	int64
47	BsmtFullBath	1460	non-null	int64
48	BsmtHalfBath	1460	non-null	int64
49	FullBath	1460	non-null	int64
50	HalfBath	1460	non-null	int64
51	BedroomAbvGr	1460	non-null	int64
52	KitchenAbvGr	1460	non-null	int64
53	KitchenQual	1460	non-null	object
54	TotRmsAbvGrd	1460	non-null	int64
55	Functional	1460	non-null	object
56	Fireplaces	1460	non-null	int64
57	FireplaceQu	770	non-null	object
58	GarageType	1379	non-null	object
59	GarageYrBlt	1379	non-null	float64
60	GarageFinish	1379	non-null	object
61	GarageCars	1460	non-null	int64
62	GarageArea	1460	non-null	int64
63	GarageQual	1379	non-null	object
64	GarageCond	1379	non-null	object
65	PavedDrive	1460	non-null	object
66	WoodDeckSF	1460	non-null	int64
67	OpenPorchSF	1460	non-null	int64
68	EnclosedPorch	1460	non-null	int64
69	3SsnPorch	1460	non-null	int64
70	ScreenPorch	1460	non-null	int64
71	PoolArea	1460	non-null	int64
72	PoolQC	7	non-null	object
73	Fence	281	non-null	object
74	MiscFeature	54	non-null	object
75	MiscVal	1460	non-null	int64
76	MoSold	1460	non-null	int64
77	YrSold	1460	non-null	int64
78	SaleType	1460	non-null	object
79	SaleCondition	1460	non-null	object
80	SalePrice	1460	non-null	int64

dtypes: float64(3), int64(35), object(43)
memory usage: 924.0+ KB

[9]: '\n\nThe `data.info()` function provides a concise summary of the dataset, including information about the number of rows, \nncolumns, data types, memory usage and missing values.\n\n'

```
[10]: data.isnull().sum()

# The code `data.isnull().sum()` calculates the sum of missing values in each
↳column of the dataset.
```

```
[10]: Id                0
      MSSubClass         0
      MSZoning           0
      LotFrontage       259
      LotArea           0
      Street            0
      Alley             1369
      LotShape          0
      LandContour       0
      Utilities         0
      LotConfig         0
      LandSlope         0
      Neighborhood     0
      Condition1        0
      Condition2        0
      BldgType          0
      HouseStyle        0
      OverallQual       0
      OverallCond       0
      YearBuilt         0
      YearRemodAdd      0
      RoofStyle         0
      RoofMatl          0
      Exterior1st       0
      Exterior2nd       0
      MasVnrType        8
      MasVnrArea        8
      ExterQual         0
      ExterCond         0
      Foundation        0
      BsmtQual          37
      BsmtCond          37
      BsmtExposure     38
      BsmtFinType1     37
      BsmtFinSF1        0
      BsmtFinType2     38
      BsmtFinSF2        0
      BsmtUnfSF         0
```

```
TotalBsmtSF      0
Heating           0
HeatingQC        0
CentralAir       0
Electrical       1
1stFlrSF         0
2ndFlrSF         0
LowQualFinSF     0
GrLivArea        0
BsmtFullBath     0
BsmtHalfBath     0
FullBath         0
HalfBath         0
BedroomAbvGr    0
KitchenAbvGr    0
KitchenQual     0
TotRmsAbvGrd    0
Functional       0
Fireplaces       0
FireplaceQu     690
GarageType       81
GarageYrBlt     81
GarageFinish     81
GarageCars       0
GarageArea       0
GarageQual      81
GarageCond      81
PavedDrive      0
WoodDeckSF      0
OpenPorchSF     0
EnclosedPorch   0
3SsnPorch       0
ScreenPorch     0
PoolArea        0
PoolQC         1453
Fence           1179
MiscFeature     1406
MiscVal         0
MoSold         0
YrSold         0
SaleType        0
SaleCondition   0
SalePrice       0
dtype: int64
```

```
[11]: plt.figure(figsize=(25,25))
      sns.heatmap(data.isnull())
```

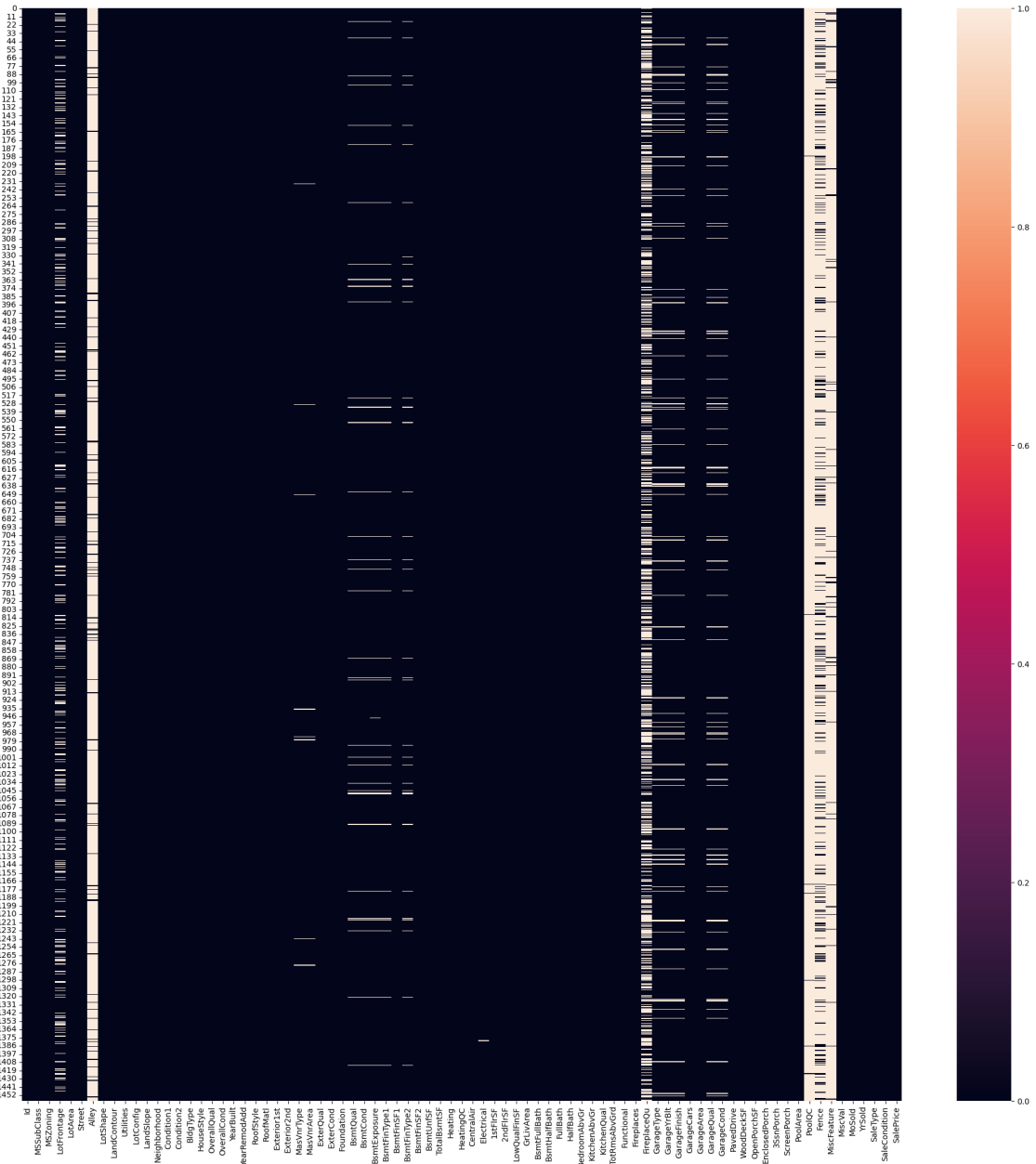
```
plt.show()
```

```
'''
```

The code `plt.figure(figsize=(25,25))` sets the figure size for the plot.

The next line `sns.heatmap(data.isnull())` creates a heatmap visualization of
↳ the missing values in the dataset
using the seaborn library.

```
'''
```



```
[11]: '\n\nThe code `plt.figure(figsize=(25,25))` sets the figure size for the plot.\n\n\nThe next line `sns.heatmap(data.isnull())` creates a heatmap visualization of the missing values in the dataset \nusing the seaborn library.\n\n'
```

```
[12]: missing_value_percent = data.isnull().sum() / data.shape[0] * 100
print(missing_value_percent)

# The code calculates the percentage of missing values in each column of the
↪ dataset and then prints the resulting percentages.
```

Id	0.000000
MSSubClass	0.000000
MSZoning	0.000000
LotFrontage	17.739726
LotArea	0.000000
Street	0.000000
Alley	93.767123
LotShape	0.000000
LandContour	0.000000
Utilities	0.000000
LotConfig	0.000000
LandSlope	0.000000
Neighborhood	0.000000
Condition1	0.000000
Condition2	0.000000
BldgType	0.000000
HouseStyle	0.000000
OverallQual	0.000000
OverallCond	0.000000
YearBuilt	0.000000
YearRemodAdd	0.000000
RoofStyle	0.000000
RoofMatl	0.000000
Exterior1st	0.000000
Exterior2nd	0.000000
MasVnrType	0.547945
MasVnrArea	0.547945
ExterQual	0.000000
ExterCond	0.000000
Foundation	0.000000
BsmtQual	2.534247
BsmtCond	2.534247
BsmtExposure	2.602740
BsmtFinType1	2.534247
BsmtFinSF1	0.000000
BsmtFinType2	2.602740

BsmtFinSF2	0.000000
BsmtUnfSF	0.000000
TotalBsmtSF	0.000000
Heating	0.000000
HeatingQC	0.000000
CentralAir	0.000000
Electrical	0.068493
1stFlrSF	0.000000
2ndFlrSF	0.000000
LowQualFinSF	0.000000
GrLivArea	0.000000
BsmtFullBath	0.000000
BsmtHalfBath	0.000000
FullBath	0.000000
HalfBath	0.000000
BedroomAbvGr	0.000000
KitchenAbvGr	0.000000
KitchenQual	0.000000
TotRmsAbvGrd	0.000000
Functional	0.000000
Fireplaces	0.000000
FireplaceQu	47.260274
GarageType	5.547945
GarageYrBlt	5.547945
GarageFinish	5.547945
GarageCars	0.000000
GarageArea	0.000000
GarageQual	5.547945
GarageCond	5.547945
PavedDrive	0.000000
WoodDeckSF	0.000000
OpenPorchSF	0.000000
EnclosedPorch	0.000000
3SsnPorch	0.000000
ScreenPorch	0.000000
PoolArea	0.000000
PoolQC	99.520548
Fence	80.753425
MiscFeature	96.301370
MiscVal	0.000000
MoSold	0.000000
YrSold	0.000000
SaleType	0.000000
SaleCondition	0.000000
SalePrice	0.000000
dtype: float64	

```
[40]: missing_value_column = missing_value_percent[missing_value_percent > 17].keys()
print(missing_value_column)
```

```
'''
```

*The code is used to identify the columns in a dataset that have missing values
↳ exceeding 17%.*

*It retrieves the keys (column names) from the `missing_value_percent`
↳ dictionary where the corresponding values
are greater than 17%.*

```
'''
```

```
Index(['LotFrontage', 'Alley', 'FireplaceQu', 'PoolQC', 'Fence',
       'MiscFeature'],
      dtype='object')
```

```
[40]: '\n\nThe code is used to identify the columns in a dataset that have missing
values exceeding 17%. \n\nIt retrieves the keys (column names) from the
`missing_value_percent` dictionary where the corresponding values \n\nare greater
than 17%.\n\n'
```

data1 after dropping missing value column > 17% :

```
[20]: data1 = data.drop(columns = missing_value_column)
```

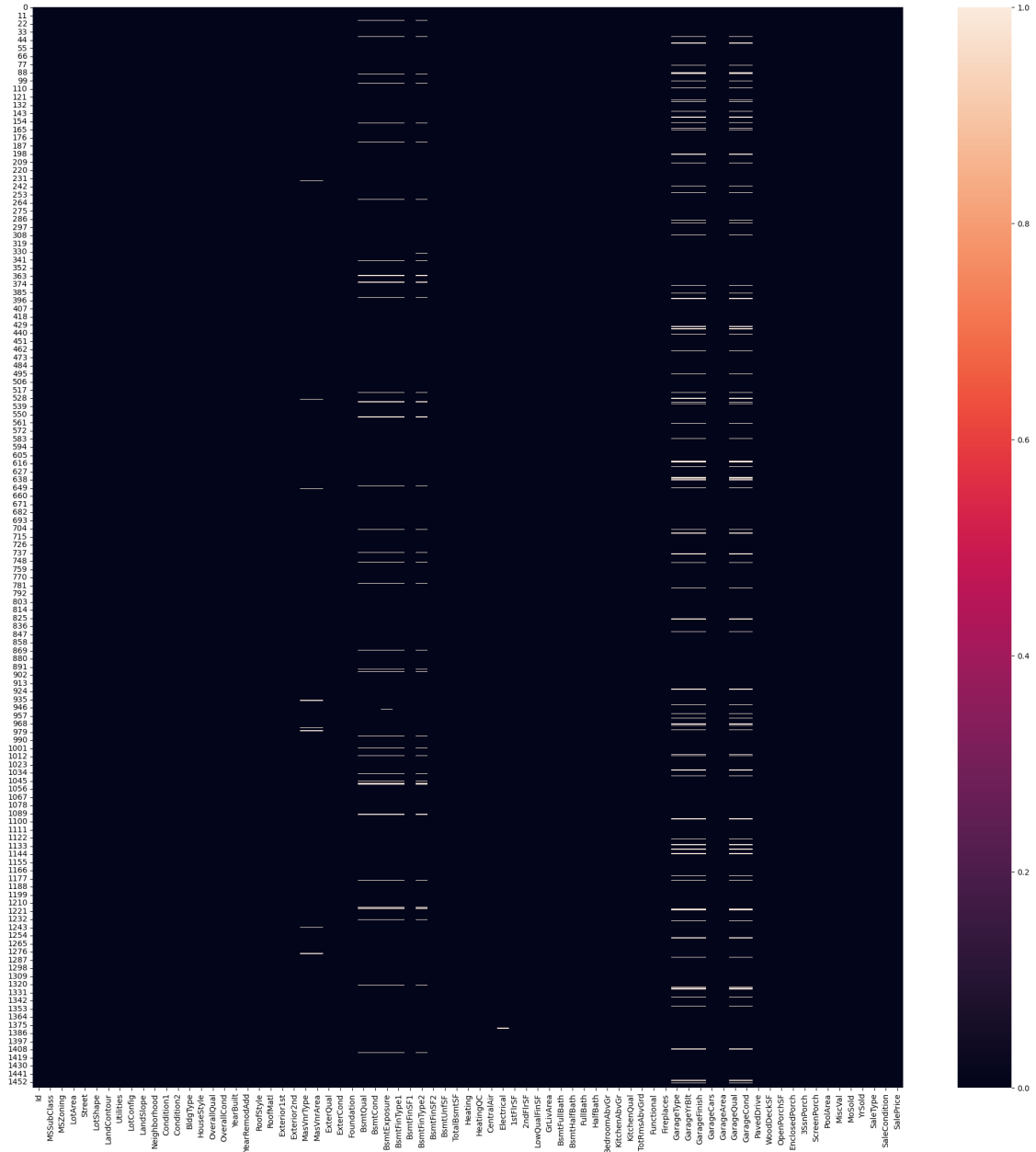
```
[21]: plt.figure(figsize=(25,25))
sns.heatmap(data1.isnull())
plt.show()
```

```
'''
```

*The code is used to get visualization of the missing values after removing
↳ columns*

where the missing values exceed 17% of the total data in those columns.

```
'''
```

[21]: '\n\nThe code is used to get visualization of the missing values after removing columns \n\nwhere the missing values exceed 20% of the total data in those columns.\n\n'

Our objective is to eliminate all the white lines from the visualization in order to achieve a cleaner and more organized appearance.

```
[22]: data1.shape # Checking the shape of the DataFrame , DataFrame contains Rows: 1460 and Columns: 76
```

[22]: (1460, 75)

data2 after dropping missing value rows.

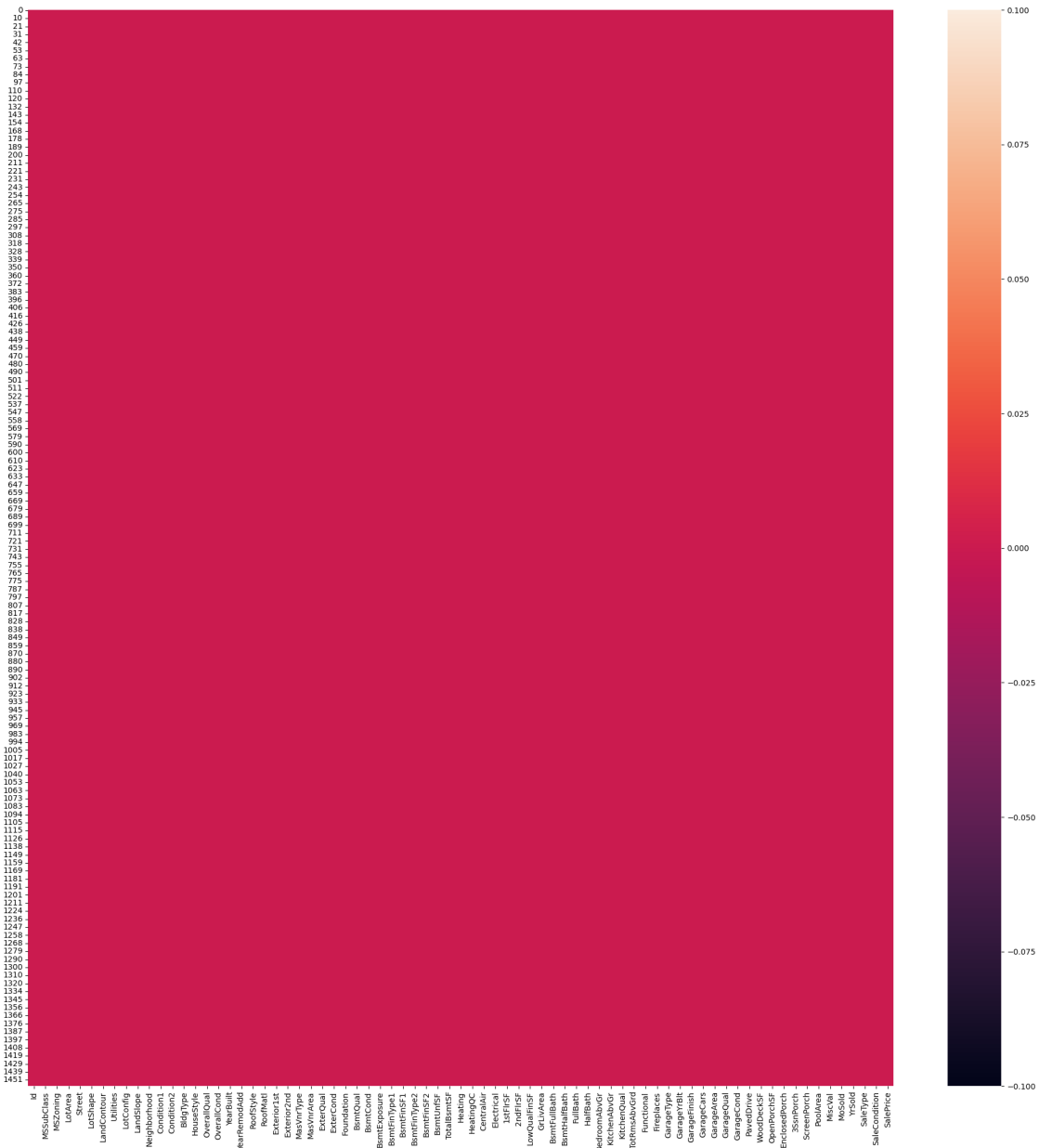
```
[23]: data2 = data1.dropna() # deleting missing value rows.
```

```
[41]: data2.shape
```

[41]: (1338, 75)

```
[18]: plt.figure(figsize=(25,25))
      sns.heatmap(df1.isnull())
      plt.show()
```

```
# This heatmap creates a visualization of missing values after dropping rows.
```



Here, we have achieved a cleaner and more organized appearance.

```
[26]: data2.isnull().sum().sum() # Here, we can see that our DataFrame doesn't contain
      ↪ any null values.
```

[26]: 0

Checking the data distribution before and after dealing with missing values for numerical column.

```
[27]: data2.select_dtypes(include=['int64','float64']).columns # Cheking the column_
      ↪names of numerical DataFrame.
```

```
[27]: Index(['Id', 'MSSubClass', 'LotArea', 'OverallQual', 'OverallCond',
          'YearBuilt', 'YearRemodAdd', 'MasVnrArea', 'BsmtFinSF1', 'BsmtFinSF2',
          'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF',
          'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath', 'HalfBath',
          'BedroomAbvGr', 'KitchenAbvGr', 'TotRmsAbvGrd', 'Fireplaces',
          'GarageYrBlt', 'GarageCars', 'GarageArea', 'WoodDeckSF', 'OpenPorchSF',
          'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'MiscVal',
          'MoSold', 'YrSold', 'SalePrice'],
          dtype='object')
```

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

```
[30]: plt.figure(figsize=(25,25))

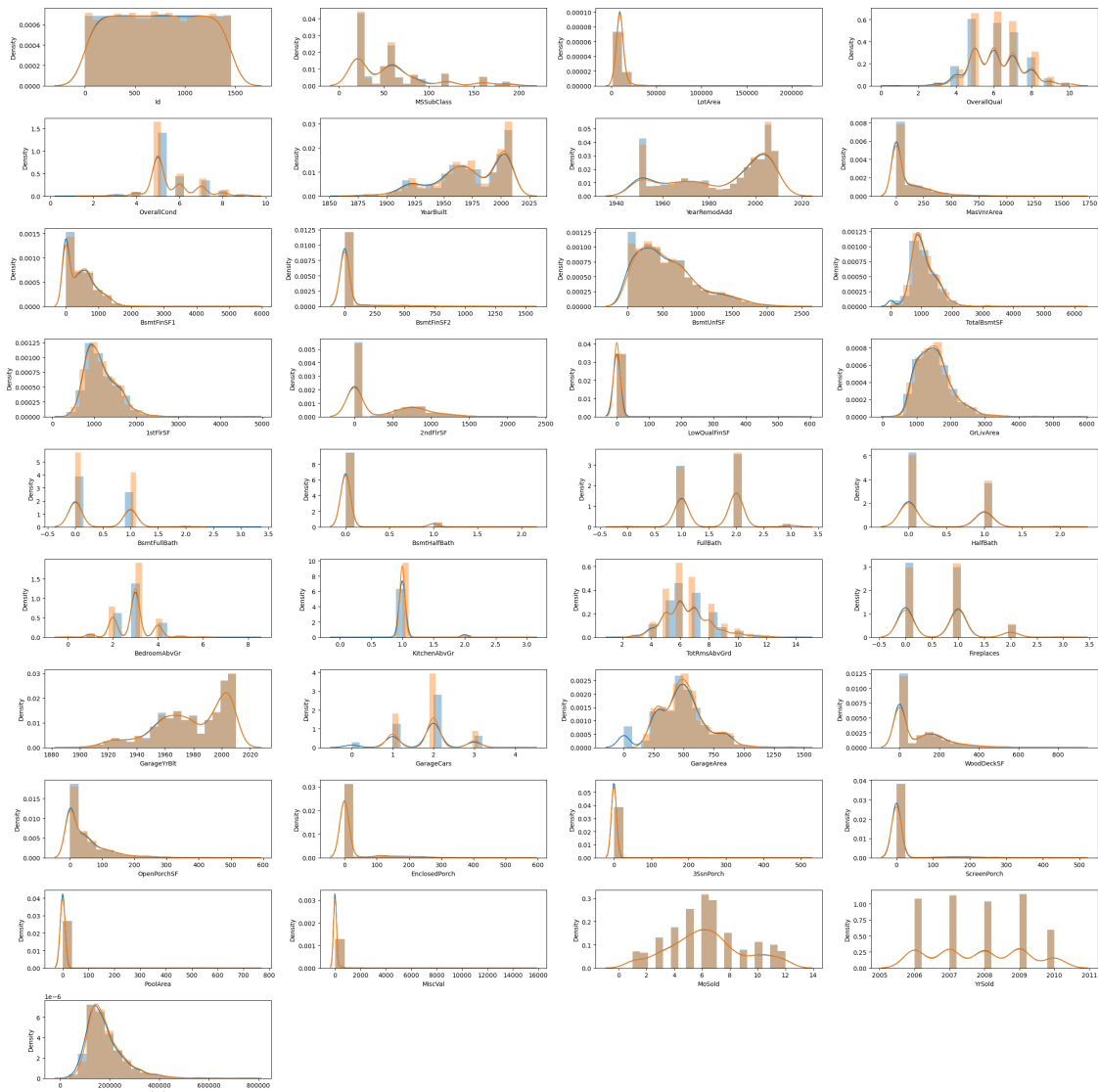
      # Calculate the number of rows and columns for the subplots

      num_rows = (len(num_var) + 3) // 4 # Adjust the number of columns as needed
      num_cols = 4

      for i, var in enumerate(num_var):
          plt.subplot(num_rows, num_cols, i+1)
          sns.distplot(data[var], bins=20)
          sns.distplot(data2[var], bins=20)

      plt.tight_layout() # Optional: Adjusts the spacing between subplots

      plt.show()
```



We can see there is not much difference between original dataset and cleaned dataset.

Checking the data distribution before and after dealing with missing values for categorical column.

```
[32]: data2.select_dtypes(include=['object']).columns
```

```
# It is used to retrieve the column names of a DataFrame df1 that have object_  
↳ data type.
```

```
[32]: Index(['MSZoning', 'Street', 'LotShape', 'LandContour', 'Utilities',  
'LotConfig', 'LandSlope', 'Neighborhood', 'Condition1', 'Condition2',  
'BldgType', 'HouseStyle', 'RoofStyle', 'RoofMatl', 'Exterior1st',  
'Exterior2nd', 'MasVnrType', 'ExterQual', 'ExterCond', 'Foundation',
```

```
'BsmtQual', 'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2',
'Heating', 'HeatingQC', 'CentralAir', 'Electrical', 'KitchenQual',
'Functional', 'GarageType', 'GarageFinish', 'GarageQual', 'GarageCond',
'PavedDrive', 'SaleType', 'SaleCondition'],
dtype='object')
```

```
[33]: cat_var = ['MSZoning', 'Street', 'LotShape', 'LandContour', 'Utilities',
'LotConfig', 'LandSlope', 'Neighborhood', 'Condition1', 'Condition2',
'BldgType', 'HouseStyle', 'RoofStyle', 'RoofMat1', 'Exterior1st',
'Exterior2nd', 'MasVnrType', 'ExterQual', 'ExterCond', 'Foundation',
'BsmtQual', 'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2',
'Heating', 'HeatingQC', 'CentralAir', 'Electrical', 'KitchenQual',
'Functional', 'GarageType', 'GarageFinish', 'GarageQual', 'GarageCond',
'PavedDrive', 'SaleType', 'SaleCondition']
```

```
[36]: pd.concat([data['MSZoning'].value_counts() / data.shape[0] * 100,
data2['MSZoning'].value_counts() / data2.shape[0] * 100], axis=1,
keys=['MSZoning_original', 'MSZoning_clean'])
```

```
[36]:
```

	MSZoning_original	MSZoning_clean
RL	78.835616	79.671151
RM	14.931507	14.275037
FV	4.452055	4.633782
RH	1.095890	0.822123
C (all)	0.684932	0.597907

The code snippet you provided combines the value counts of the ‘MSZoning’ column from the original data (data) and the cleaned data (df1) using pd.concat(). It creates a DataFrame that shows the percentage distribution of the values in the ‘MSZoning’ column for both datasets.

```
[37]: def cat_var_dist(var):
return pd.concat([data[var].value_counts() / data.shape[0]*100,
data2[var].value_counts() / data2.shape[0] * 100], axis=1,
keys=[var + '_original', var + '_clean'])
```

```
[38]: cat_var_dist('BsmtQual')
```

```
[38]:
```

	BsmtQual_original	BsmtQual_clean
TA	44.452055	44.394619
Gd	42.328767	44.245142
Ex	8.287671	8.968610
Fa	2.397260	2.391629

The cat_var_dist() function you provided calculates the percentage distribution of values in a categorical variable for both the original data (data) and the cleaned data (df1). It returns a DataFrame that combines these distributions side by side.

```
[39]: for var in cat_var:
      result = cat_var_dist(var)
      print(result)
      print('\n')
```

	MSZoning_ordinal	MSZoning_clean
RL	78.835616	79.671151
RM	14.931507	14.275037
FV	4.452055	4.633782
RH	1.095890	0.822123
C (all)	0.684932	0.597907

	Street_ordinal	Street_clean
Pave	99.589041	99.626308
Grvl	0.410959	0.373692

	LotShape_ordinal	LotShape_clean
Reg	63.356164	61.958146
IR1	33.150685	34.304933
IR2	2.808219	2.989537
IR3	0.684932	0.747384

	LandContour_ordinal	LandContour_clean
Lvl	89.794521	90.134529
Bnk	4.315068	3.886398
HLS	3.424658	3.587444
Low	2.465753	2.391629

	Utilities_ordinal	Utilities_clean
AllPub	99.931507	99.925262
NoSeWa	0.068493	0.074738

	LotConfig_ordinal	LotConfig_clean
Inside	72.054795	71.524664
Corner	18.013699	18.236173
CulDSac	6.438356	6.726457
FR2	3.219178	3.213752
FR3	0.273973	0.298954

	LandSlope_ordinal	LandSlope_clean
Gtl	94.657534	94.544096

Mod	4.452055	4.559043
Sev	0.890411	0.896861

	Neighborhood_organic	Neighborhood_clean
NAmes	15.410959	15.620329
CollgCr	10.273973	10.911809
OldTown	7.739726	7.473842
Edwards	6.849315	5.231689
Somerst	5.890411	6.203288
Gilbert	5.410959	5.754858
NridgHt	5.273973	5.605381
Sawyer	5.068493	5.156951
NWAmes	5.000000	5.455904
SawyerW	4.041096	3.961136
BrkSide	3.972603	3.512706
Crawfor	3.493151	3.736921
Mitchel	3.356164	3.139013
NoRidge	2.808219	3.064275
Timber	2.602740	2.765321
IDOTRR	2.534247	2.167414
ClearCr	1.917808	1.943199
StoneBr	1.712329	1.868460
SWISU	1.712329	1.494768
MeadowV	1.164384	0.896861
Blmngtn	1.164384	1.270553
BrDale	1.095890	1.121076
Veenker	0.753425	0.822123
NPkVill	0.616438	0.672646
Blueste	0.136986	0.149477

	Condition1_organic	Condition1_clean
Norm	86.301370	86.846039
Feedr	5.547945	4.708520
Artery	3.287671	3.213752
RRAn	1.780822	1.943199
PosN	1.301370	1.420030
RR Ae	0.753425	0.747384
PosA	0.547945	0.597907
RRNn	0.342466	0.373692
RRNe	0.136986	0.149477

	Condition2_organic	Condition2_clean
Norm	98.972603	98.953662
Feedr	0.410959	0.373692
Artery	0.136986	0.149477

RRNn	0.136986	0.149477
PosN	0.136986	0.149477
PosA	0.068493	0.074738
RRAn	0.068493	0.074738
RR Ae	0.068493	0.074738

	BldgType_original	BldgType_clean
1Fam	83.561644	85.052317
TwnhsE	7.808219	8.370703
Duplex	3.561644	2.092676
Twnhs	2.945205	2.840060
2fmCon	2.123288	1.644245

	HouseStyle_original	HouseStyle_clean
1Story	49.726027	49.103139
2Story	30.479452	31.838565
1.5Fin	10.547945	10.014948
SLvl	4.452055	4.783259
SFoyer	2.534247	2.242152
1.5Unf	0.958904	0.822123
2.5Unf	0.753425	0.747384
2.5Fin	0.547945	0.448430

	RoofStyle_original	RoofStyle_clean
Gable	78.150685	77.503737
Hip	19.589041	20.328849
Flat	0.890411	0.822123
Gambrel	0.753425	0.747384
Mansard	0.479452	0.448430
Shed	0.136986	0.149477

	RoofMatl_original	RoofMatl_clean
CompShg	98.219178	98.206278
Tar&Grv	0.753425	0.672646
WdShngl	0.410959	0.448430
WdShake	0.342466	0.373692
Metal	0.068493	0.074738
Membran	0.068493	0.074738
Roll	0.068493	0.074738
ClyTile	0.068493	0.074738

	Exterior1st_original	Exterior1st_clean
VinylSd	35.273973	36.322870

HdBoard	15.205479	15.769806
MetalSd	15.068493	15.022422
Wd Sdng	14.109589	13.677130
Plywood	7.397260	7.473842
CemntBd	4.178082	3.886398
BrkFace	3.424658	3.288490
WdShing	1.780822	1.494768
Stucco	1.712329	1.569507
AsbShng	1.369863	1.121076
BrkComm	0.136986	0.074738
Stone	0.136986	0.149477
AsphShn	0.068493	NaN
ImStucc	0.068493	0.074738
CBlock	0.068493	0.074738

	Exterior2nd_orginal	Exterior2nd_clean
VinylSd	34.520548	35.500747
MetalSd	14.657534	14.723468
HdBoard	14.178082	14.723468
Wd Sdng	13.493151	13.153961
Plywood	9.726027	9.491779
CmentBd	4.109589	3.811659
Wd Shng	2.602740	2.391629
Stucco	1.780822	1.718984
BrkFace	1.712329	1.644245
AsbShng	1.369863	1.195815
ImStucc	0.684932	0.747384
Brk Cmn	0.479452	0.448430
Stone	0.342466	0.149477
AsphShn	0.205479	0.149477
Other	0.068493	0.074738
CBlock	0.068493	0.074738

	MasVnrType_orginal	MasVnrType_clean
None	59.178082	57.025411
BrkFace	30.479452	32.286996
Stone	8.767123	9.566517
BrkCmn	1.027397	1.121076

	ExterQual_orginal	ExterQual_clean
TA	62.054795	60.014948
Gd	33.424658	35.650224
Ex	3.561644	3.811659
Fa	0.958904	0.523169

	ExterCond_original	ExterCond_clean
TA	87.808219	88.415546
Gd	10.000000	10.239163
Fa	1.917808	1.195815
Ex	0.205479	0.149477
Po	0.068493	NaN

	Foundation_original	Foundation_clean
PConc	44.315068	46.337818
CBlock	43.424658	43.348281
BrkTil	10.000000	9.641256
Slab	1.643836	NaN
Stone	0.410959	0.448430
Wood	0.205479	0.224215

	BsmtQual_original	BsmtQual_clean
TA	44.452055	44.394619
Gd	42.328767	44.245142
Ex	8.287671	8.968610
Fa	2.397260	2.391629

	BsmtCond_original	BsmtCond_clean
TA	89.794521	92.451420
Gd	4.452055	4.633782
Fa	3.082192	2.840060
Po	0.136986	0.074738

	BsmtExposure_original	BsmtExposure_clean
No	65.273973	66.292975
Av	15.136986	15.919283
Gd	9.178082	9.491779
Mn	7.808219	8.295964

	BsmtFinType1_original	BsmtFinType1_clean
Unf	29.452055	29.297459
GLQ	28.630137	30.044843
ALQ	15.068493	15.620329
BLQ	10.136986	10.538117
Rec	9.109589	9.342302
LwQ	5.068493	5.156951

	BsmtFinType2_orginal	BsmtFinType2_clean
Unf	86.027397	87.892377
Rec	3.698630	3.961136
LwQ	3.150685	3.437967
BLQ	2.260274	2.391629
ALQ	1.301370	1.420030
GLQ	0.958904	0.896861

	Heating_orginal	Heating_clean
GasA	97.808219	98.505232
GasW	1.232877	1.195815
Grav	0.479452	0.224215
Wall	0.273973	NaN
OthW	0.136986	0.074738
Floor	0.068493	NaN

	HeatingQC_orginal	HeatingQC_clean
Ex	50.753425	52.615845
TA	29.315068	28.400598
Gd	16.506849	16.218236
Fa	3.356164	2.690583
Po	0.068493	0.074738

	CentralAir_orginal	CentralAir_clean
Y	93.493151	95.440957
N	6.506849	4.559043

	Electrical_orginal	Electrical_clean
SBrkr	91.369863	92.825112
FuseA	6.438356	5.680120
FuseF	1.849315	1.270553
FuseP	0.205479	0.149477
Mix	0.068493	0.074738

	KitchenQual_orginal	KitchenQual_clean
TA	50.342466	48.579970
Gd	40.136986	42.451420
Ex	6.849315	7.249626
Fa	2.671233	1.718984

	Functional_orginal	Functional_clean
Typ	93.150685	93.721973

Min2	2.328767	2.242152
Min1	2.123288	2.092676
Mod	1.027397	0.822123
Maj1	0.958904	0.747384
Maj2	0.342466	0.298954
Sev	0.068493	0.074738

	GarageType_original	GarageType_clean
Attchd	59.589041	63.677130
Detchd	26.506849	27.578475
BuiltIn	6.027397	6.352765
Basment	1.301370	1.420030
CarPort	0.616438	0.523169
2Types	0.410959	0.448430

	GarageFinish_original	GarageFinish_clean
Unf	41.438356	43.348281
RFn	28.904110	30.866966
Fin	24.109589	25.784753

	GarageQual_original	GarageQual_clean
TA	89.794521	94.917788
Fa	3.287671	3.587444
Gd	0.958904	1.046338
Ex	0.205479	0.224215
Po	0.205479	0.224215

	GarageCond_original	GarageCond_clean
TA	90.821918	96.188341
Fa	2.397260	2.466368
Gd	0.616438	0.672646
Po	0.479452	0.523169
Ex	0.136986	0.149477

	PavedDrive_original	PavedDrive_clean
Y	91.780822	93.946188
N	6.164384	4.035874
P	2.054795	2.017937

	SaleType_original	SaleType_clean
WD	86.780822	86.547085
New	8.356164	8.744395

COD	2.945205	3.139013
ConLD	0.616438	0.448430
ConLI	0.342466	0.298954
ConLw	0.342466	0.298954
CWD	0.273973	0.298954
Oth	0.205479	0.074738
Con	0.136986	0.149477

	SaleCondition_orginal	SaleCondition_clean
Normal	82.054795	82.511211
Partial	8.561644	8.968610
Abnorml	6.917808	6.427504
Family	1.369863	1.494768
Alloca	0.821918	0.523169
AdjLand	0.273973	0.074738

Above code iterate over each variable in the `cat_var` list, call the `cat_var_dist()` function for each variable, and print the resulting DataFrame.

e-imputation-by-mean-and-median-2

June 11, 2023

1 Data Cleaning :

1.1 Missing value imputation by Mean and Median :

Importing necessary libraries :

```
[1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
from IPython.display import Image
```

data (Original) :

```
[3]: data = pd.read_csv('train.csv') # Loading dataset
```

```
[4]: data.shape # Checking the shape of the DataFrame, DataFrame contains Rows: 1460
↳ 1460 and Columns: 81.
```

```
[4]: (1460, 81)
```

```
[5]: data.head(2) # Checking first 2 rows from the DataFrame.
```

```
[5]:   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

   LandContour Utilities  ... PoolArea PoolQC Fence MiscFeature MiscVal MoSold \
0          Lvl1   AllPub  ...         0   NaN   NaN           NaN         0     2
1          Lvl1   AllPub  ...         0   NaN   NaN           NaN         0     5

   YrSold  SaleType  SaleCondition  SalePrice
0    2008         WD           Normal    208500
1    2007         WD           Normal    181500
```

```
[2 rows x 81 columns]
```

When we check the shape of the data using the command “data.shape”, we can observe that the DataFrame consists of 1460 rows and 81 columns. However, when we examine the first two rows of the DataFrame using the command “data.head(2)”, it doesn’t display all 81 columns. As a result, we utilize the following code, specifically “pd.set_option()”, to address this issue.

```
[6]: pd.set_option('display.max_column', None)
pd.set_option('display.max_rows', None)
```

pd.set_option(‘display.max_columns’, None): This line sets the maximum number of columns to be displayed in the output to None, which means there is no limit. As a result, all columns in a DataFrame will be shown when you print or display it.

pd.set_option(‘display.max_rows’, None): This line sets the maximum number of rows to be displayed in the output to None, removing any limit. As a result, all rows in a DataFrame will be shown when you print or display it.

```
[7]: data.head(2) # The code `data.head(2)` displays the first two rows of the
↳ dataset, showing all 81 columns.
```

```
[7]:
```

	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	
	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Condition1	\		
0	Lvl	AllPub	Inside	Gtl	CollgCr	Norm			
1	Lvl	AllPub	FR2	Gtl	Veenker	Feedr			
	Condition2	BldgType	HouseStyle	OverallQual	OverallCond	YearBuilt	\		
0	Norm	1Fam	2Story	7	5	2003			
1	Norm	1Fam	1Story	6	8	1976			
	YearRemodAdd	RoofStyle	RoofMat1	Exterior1st	Exterior2nd	MasVnrType	\		
0	2003	Gable	CompShg	VinylSd	VinylSd	BrkFace			
1	1976	Gable	CompShg	MetalSd	MetalSd	None			
	MasVnrArea	ExterQual	ExterCond	Foundation	BsmtQual	BsmtCond	BsmtExposure	\	
0	196.0	Gd	TA	PConc	Gd	TA	No		
1	0.0	TA	TA	CBlock	Gd	TA	Gd		
	BsmtFinType1	BsmtFinSF1	BsmtFinType2	BsmtFinSF2	BsmtUnfSF	TotalBsmtSF	\		
0	GLQ	706	Unf	0	150	856			
1	ALQ	978	Unf	0	284	1262			
	Heating	HeatingQC	CentralAir	Electrical	1stFlrSF	2ndFlrSF	LowQualFinSF	\	
0	GasA	Ex	Y	SBrkr	856	854	0		
1	GasA	Ex	Y	SBrkr	1262	0	0		
	GrLivArea	BsmtFullBath	BsmtHalfBath	FullBath	HalfBath	BedroomAbvGr	\		


```

0      1710      1      0      2      1      3
1      1262      0      1      2      0      3

KitchenAbvGr KitchenQual TotRmsAbvGrd Functional Fireplaces FireplaceQu \
0          1          Gd          8          Typ          0          NaN
1          1          TA          6          Typ          1          TA

GarageType GarageYrBltd GarageFinish GarageCars GarageArea GarageQual \
0 Attchd      2003.0      RFn          2          548          TA
1 Attchd      1976.0      RFn          2          460          TA

GarageCond PavedDrive WoodDeckSF OpenPorchSF EnclosedPorch 3SsnPorch \
0 TA          Y          0          61          0          0
1 TA          Y          298         0          0          0

ScreenPorch PoolArea PoolQC Fence MiscFeature MiscVal MoSold YrSold \
0          0          0      NaN      NaN          NaN          0          2      2008
1          0          0      NaN      NaN          NaN          0          5      2007

SaleType SaleCondition SalePrice
0 WD          Normal      208500
1 WD          Normal      181500

```

```
[8]: data.tail(2) # The code `data.tail(2)` displays the last two rows of the
↳ dataset.
```

```
[8]:      Id MSSubClass MSZoning LotFrontage LotArea Street Alley LotShape \
1458 1459          20      RL          68.0      9717  Pave  NaN      Reg
1459 1460          20      RL          75.0      9937  Pave  NaN      Reg

LandContour Utilities LotConfig LandSlope Neighborhood Condition1 \
1458      Lvl      AllPub      Inside      Gtl          NNames      Norm
1459      Lvl      AllPub      Inside      Gtl          Edwards      Norm

Condition2 BldgType HouseStyle OverallQual OverallCond YearBuilt \
1458      Norm      1Fam      1Story          5          6          1950
1459      Norm      1Fam      1Story          5          6          1965

YearRemodAdd RoofStyle RoofMatl Exterior1st Exterior2nd MasVnrType \
1458      1996      Hip      CompShg      MetalSd      MetalSd      None
1459      1965      Gable      CompShg      HdBoard      HdBoard      None

MasVnrArea ExterQual ExterCond Foundation BsmtQual BsmtCond \
1458      0.0      TA          TA      CBlock      TA          TA
1459      0.0      Gd          TA      CBlock      TA          TA

BsmtExposure BsmtFinType1 BsmtFinSF1 BsmtFinType2 BsmtFinSF2 \

```

1458	Mn	GLQ	49	Rec	1029			
1459	No	BLQ	830	LwQ	290			
	BsmtUnfSF	TotalBsmtSF	Heating	HeatingQC	CentralAir	Electrical		
1458	0	1078	GasA	Gd	Y	FuseA		
1459	136	1256	GasA	Gd	Y	SBrkr		
	1stFlrSF	2ndFlrSF	LowQualFinSF	GrLivArea	BsmtFullBath	BsmtHalfBath		
1458	1078	0	0	1078	1	0		
1459	1256	0	0	1256	1	0		
	FullBath	HalfBath	BedroomAbvGr	KitchenAbvGr	KitchenQual			
1458	1	0	2	1	Gd			
1459	1	1	3	1	TA			
	TotRmsAbvGrd	Functional	Fireplaces	FireplaceQu	GarageType	GarageYrBlt		
1458	5	Typ	0	NaN	Attchd	1950.0		
1459	6	Typ	0	NaN	Attchd	1965.0		
	GarageFinish	GarageCars	GarageArea	GarageQual	GarageCond	PavedDrive		
1458	Unf	1	240	TA	TA	Y		
1459	Fin	1	276	TA	TA	Y		
	WoodDeckSF	OpenPorchSF	EnclosedPorch	3SsnPorch	ScreenPorch			
1458	366	0	112	0	0			
1459	736	68	0	0	0			
	PoolArea	PoolQC	Fence	MiscFeature	MiscVal	MoSold	YrSold	SaleType
1458	0	NaN	NaN	NaN	0	4	2010	WD
1459	0	NaN	NaN	NaN	0	6	2008	WD
	SaleCondition	SalePrice						
1458	Normal	142125						
1459	Normal	147500						

```
[9]: data.info()
```

```
'''
The `data.info()` function provides a concise summary of the dataset, including
information about the number of rows,
columns, data types, memory usage and missing values.
'''
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 81 columns):
```

#	Column	Non-Null Count	Dtype
0	Id	1460 non-null	int64
1	MSSubClass	1460 non-null	int64
2	MSZoning	1460 non-null	object
3	LotFrontage	1201 non-null	float64
4	LotArea	1460 non-null	int64
5	Street	1460 non-null	object
6	Alley	91 non-null	object
7	LotShape	1460 non-null	object
8	LandContour	1460 non-null	object
9	Utilities	1460 non-null	object
10	LotConfig	1460 non-null	object
11	LandSlope	1460 non-null	object
12	Neighborhood	1460 non-null	object
13	Condition1	1460 non-null	object
14	Condition2	1460 non-null	object
15	BldgType	1460 non-null	object
16	HouseStyle	1460 non-null	object
17	OverallQual	1460 non-null	int64
18	OverallCond	1460 non-null	int64
19	YearBuilt	1460 non-null	int64
20	YearRemodAdd	1460 non-null	int64
21	RoofStyle	1460 non-null	object
22	RoofMatl	1460 non-null	object
23	Exterior1st	1460 non-null	object
24	Exterior2nd	1460 non-null	object
25	MasVnrType	1452 non-null	object
26	MasVnrArea	1452 non-null	float64
27	ExterQual	1460 non-null	object
28	ExterCond	1460 non-null	object
29	Foundation	1460 non-null	object
30	BsmtQual	1423 non-null	object
31	BsmtCond	1423 non-null	object
32	BsmtExposure	1422 non-null	object
33	BsmtFinType1	1423 non-null	object
34	BsmtFinSF1	1460 non-null	int64
35	BsmtFinType2	1422 non-null	object
36	BsmtFinSF2	1460 non-null	int64
37	BsmtUnfSF	1460 non-null	int64
38	TotalBsmtSF	1460 non-null	int64
39	Heating	1460 non-null	object
40	HeatingQC	1460 non-null	object
41	CentralAir	1460 non-null	object
42	Electrical	1459 non-null	object
43	1stFlrSF	1460 non-null	int64
44	2ndFlrSF	1460 non-null	int64
45	LowQualFinSF	1460 non-null	int64

```

46 GrLivArea      1460 non-null  int64
47 BsmtFullBath  1460 non-null  int64
48 BsmtHalfBath  1460 non-null  int64
49 FullBath      1460 non-null  int64
50 HalfBath      1460 non-null  int64
51 BedroomAbvGr 1460 non-null  int64
52 KitchenAbvGr 1460 non-null  int64
53 KitchenQual   1460 non-null  object
54 TotRmsAbvGrd 1460 non-null  int64
55 Functional    1460 non-null  object
56 Fireplaces    1460 non-null  int64
57 FireplaceQu   770 non-null   object
58 GarageType    1379 non-null  object
59 GarageYrBlt   1379 non-null  float64
60 GarageFinish  1379 non-null  object
61 GarageCars    1460 non-null  int64
62 GarageArea    1460 non-null  int64
63 GarageQual    1379 non-null  object
64 GarageCond    1379 non-null  object
65 PavedDrive    1460 non-null  object
66 WoodDeckSF    1460 non-null  int64
67 OpenPorchSF   1460 non-null  int64
68 EnclosedPorch 1460 non-null  int64
69 3SsnPorch     1460 non-null  int64
70 ScreenPorch   1460 non-null  int64
71 PoolArea      1460 non-null  int64
72 PoolQC        7 non-null     object
73 Fence         281 non-null   object
74 MiscFeature   54 non-null    object
75 MiscVal       1460 non-null  int64
76 MoSold        1460 non-null  int64
77 YrSold        1460 non-null  int64
78 SaleType      1460 non-null  object
79 SaleCondition 1460 non-null  object
80 SalePrice     1460 non-null  int64
dtypes: float64(3), int64(35), object(43)
memory usage: 924.0+ KB

```

[9]: '\n\nThe `data.info()` function provides a concise summary of the dataset, including information about the number of rows, \ncolumns, data types, memory usage and missing values.\n\n'

[10]: `data.isnull().sum()`

```

# The code `data.isnull().sum()` calculates the sum of missing values in each
↳ column of the dataset.

```

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

```

BsmtFullBath      0
BsmtHalfBath     0
FullBath         0
HalfBath         0
BedroomAbvGr     0
KitchenAbvGr     0
KitchenQual      0
TotRmsAbvGrd    0
Functional       0
Fireplaces       0
FireplaceQu     690
GarageType       81
GarageYrBlt     81
GarageFinish     81
GarageCars       0
GarageArea       0
GarageQual      81
GarageCond      81
PavedDrive       0
WoodDeckSF      0
OpenPorchSF     0
EnclosedPorch   0
3SsnPorch       0
ScreenPorch     0
PoolArea        0
PoolQC         1453
Fence           1179
MiscFeature     1406
MiscVal         0
MoSold          0
YrSold          0
SaleType        0
SaleCondition   0
SalePrice       0
dtype: int64

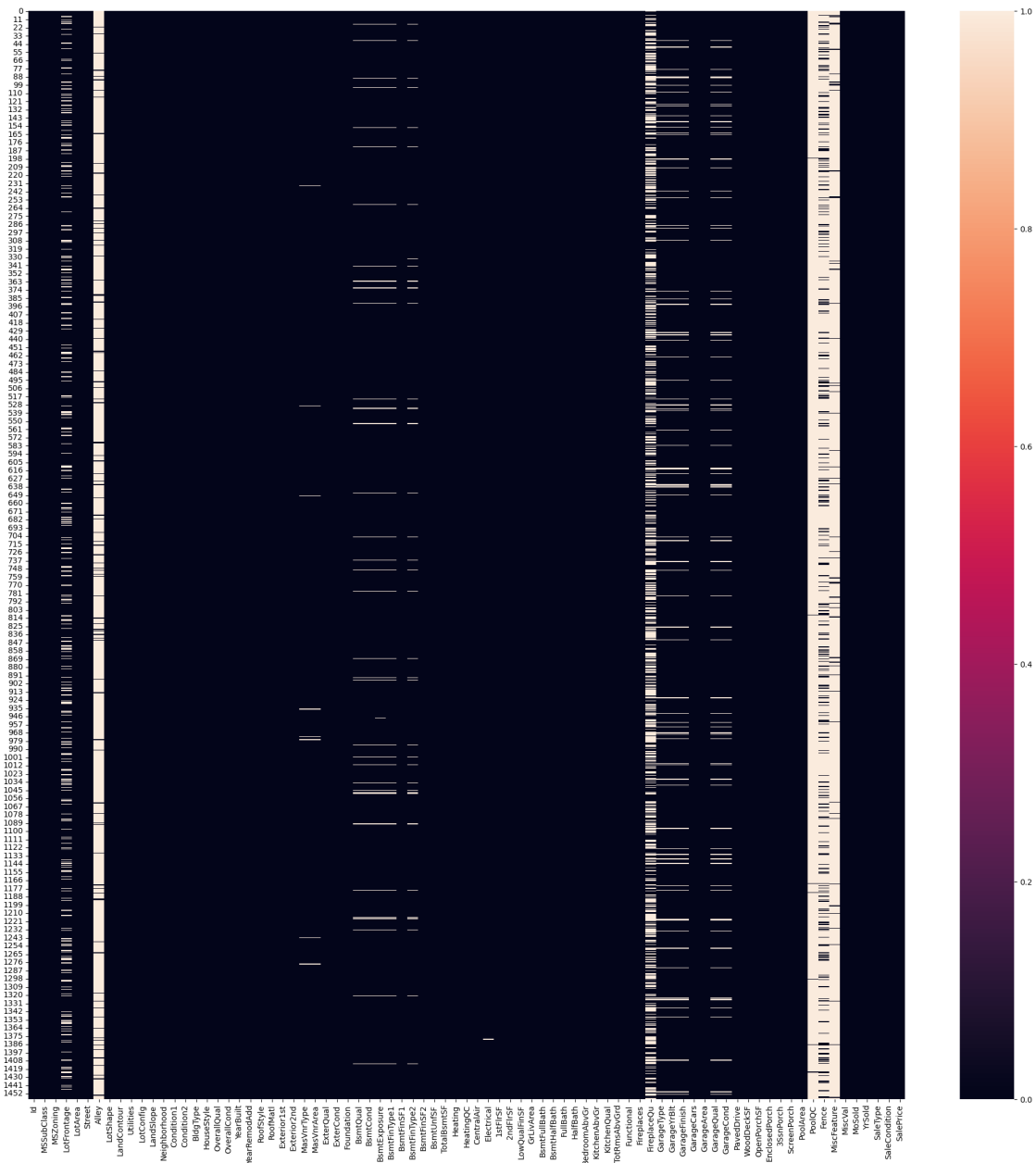
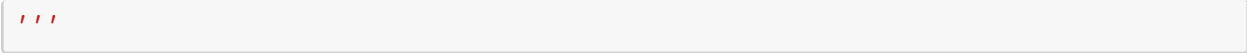
```

```
[11]: plt.figure(figsize=(25,25))
sns.heatmap(data.isnull())
plt.show()
```

'''

The code `plt.figure(figsize=(25,25))` sets the figure size for the plot.

*The next line `sns.heatmap(data.isnull())` creates a heatmap visualization of
↳ the missing values in the dataset
using the seaborn library.*



[11]: '\n\nThe code `plt.figure(figsize=(25,25))` sets the figure size for the plot.
\n\nThe next line `sns.heatmap(data.isnull())` creates a heatmap visualization of the missing values in the dataset \nusing the seaborn library.\n\n'

```
[12]: missing_value_percent = data.isnull().sum() / data.shape[0] * 100
print(missing_value_percent)
```

```
# The code calculates the percentage of missing values in each column of the
↳ dataset and then prints the resulting percentages.
```

Id	0.000000
MSSubClass	0.000000
MSZoning	0.000000
LotFrontage	17.739726
LotArea	0.000000
Street	0.000000
Alley	93.767123
LotShape	0.000000
LandContour	0.000000
Utilities	0.000000
LotConfig	0.000000
LandSlope	0.000000
Neighborhood	0.000000
Condition1	0.000000
Condition2	0.000000
BldgType	0.000000
HouseStyle	0.000000
OverallQual	0.000000
OverallCond	0.000000
YearBuilt	0.000000
YearRemodAdd	0.000000
RoofStyle	0.000000
RoofMatl	0.000000
Exterior1st	0.000000
Exterior2nd	0.000000
MasVnrType	0.547945
MasVnrArea	0.547945
ExterQual	0.000000
ExterCond	0.000000
Foundation	0.000000
BsmtQual	2.534247
BsmtCond	2.534247
BsmtExposure	2.602740
BsmtFinType1	2.534247
BsmtFinSF1	0.000000
BsmtFinType2	2.602740
BsmtFinSF2	0.000000
BsmtUnfSF	0.000000
TotalBsmtSF	0.000000
Heating	0.000000
HeatingQC	0.000000
CentralAir	0.000000
Electrical	0.068493
1stFlrSF	0.000000

2ndFlrSF	0.000000
LowQualFinSF	0.000000
GrLivArea	0.000000
BsmtFullBath	0.000000
BsmtHalfBath	0.000000
FullBath	0.000000
HalfBath	0.000000
BedroomAbvGr	0.000000
KitchenAbvGr	0.000000
KitchenQual	0.000000
TotRmsAbvGrd	0.000000
Functional	0.000000
Fireplaces	0.000000
FireplaceQu	47.260274
GarageType	5.547945
GarageYrBlt	5.547945
GarageFinish	5.547945
GarageCars	0.000000
GarageArea	0.000000
GarageQual	5.547945
GarageCond	5.547945
PavedDrive	0.000000
WoodDeckSF	0.000000
OpenPorchSF	0.000000
EnclosedPorch	0.000000
3SsnPorch	0.000000
ScreenPorch	0.000000
PoolArea	0.000000
PoolQC	99.520548
Fence	80.753425
MiscFeature	96.301370
MiscVal	0.000000
MoSold	0.000000
YrSold	0.000000
SaleType	0.000000
SaleCondition	0.000000
SalePrice	0.000000

dtype: float64

```
[13]: missing_value_column = missing_value_percent[missing_value_percent > 20].keys()
print(missing_value_column)
```

```
'''
```

The code is used to identify the columns in a dataset that have missing values, exceeding 20%.

It retrieves the keys (column names) from the `missing_value_percent` dictionary where the corresponding values

```
are greater than 20%.
```

```
'''
```

```
Index(['Alley', 'FireplaceQu', 'PoolQC', 'Fence', 'MiscFeature'],  
      dtype='object')
```

[13]: '\n\nThe code is used to identify the columns in a dataset that have missing values exceeding 20%. \n\nIt retrieves the keys (column names) from the `missing_value_percent` dictionary where the corresponding values \n\nare greater than 20%.\n\n'

data1 after dropping missing value column > 20% :

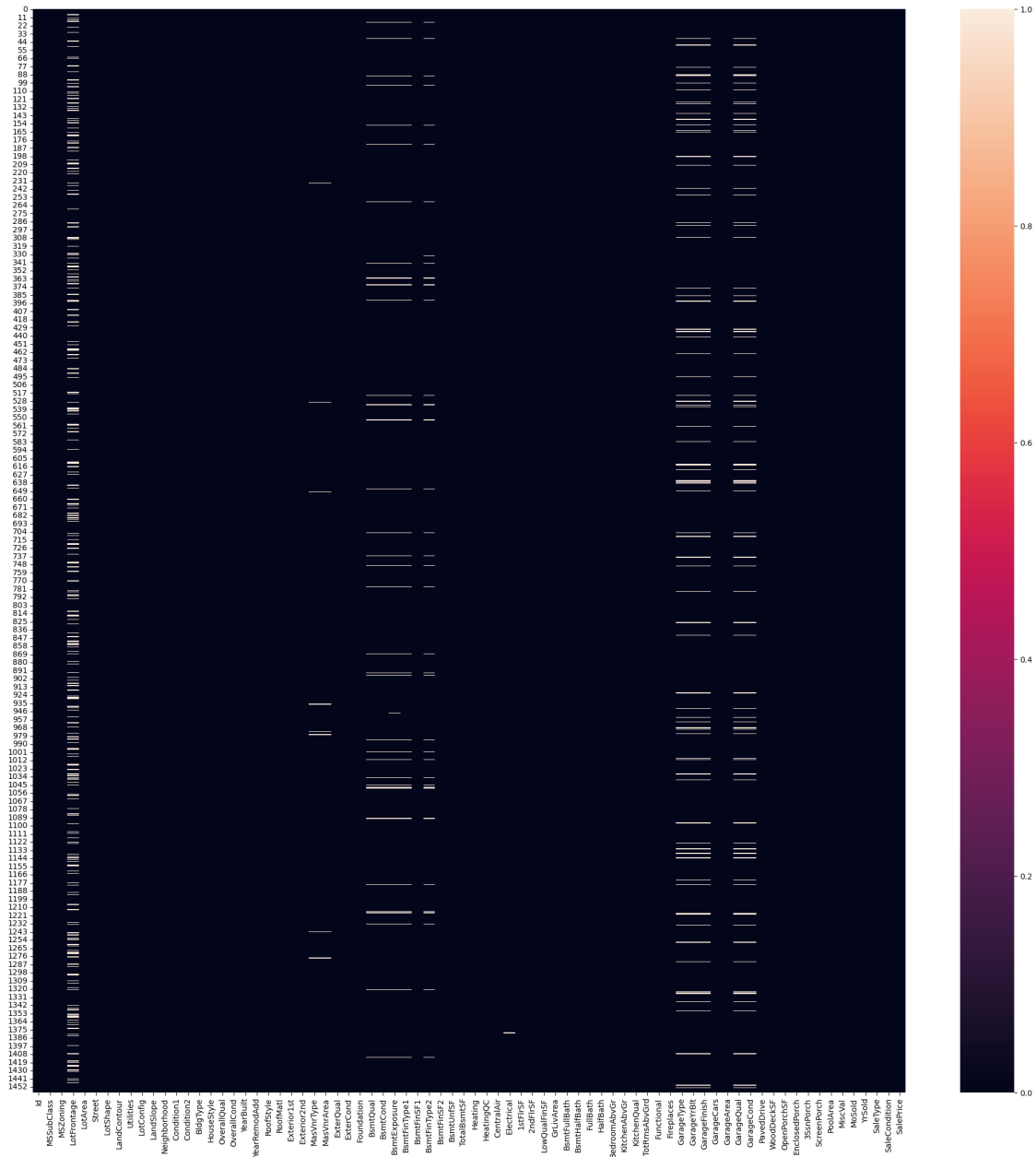
```
[14]: data1 = data.drop(columns = missing_value_column)
```

```
[15]: plt.figure(figsize=(25,25))  
      sns.heatmap(data1.isnull())  
      plt.show()
```

```
'''
```

```
The code is used to get visualization of the missing values after removing  
↪ columns  
where the missing values exceed 20% of the total data in those columns.
```

```
'''
```



[15]: '\n\nThe code is used to get visualization of the missing values after removing columns \n\nwhere the missing values exceed 20% of the total data in those columns.\n\n\n'

```
[16]: data1.shape # Checking the shape of the DataFrame , DataFrame contains Rows: 1460 and Columns: 76
```

[16]: (1460, 76)

numerical_DataFrame (selecting DataFrame that contains only numerical data types)

```
:
[17]: numerical_DataFrame = data1.select_dtypes(include=['int64', 'float64'])
numerical_DataFrame.head(2)
```

```
[17]:
```

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	\
0	1	60	65.0	8450	7	5	2003	
1	2	20	80.0	9600	6	8	1976	

	YearRemodAdd	MasVnrArea	BsmtFinSF1	BsmtFinSF2	BsmtUnfSF	TotalBsmtSF	\
0	2003	196.0	706	0	150	856	
1	1976	0.0	978	0	284	1262	

	1stFlrSF	2ndFlrSF	LowQualFinSF	GrLivArea	BsmtFullBath	BsmtHalfBath	\
0	856	854	0	1710	1	0	
1	1262	0	0	1262	0	1	

	FullBath	HalfBath	BedroomAbvGr	KitchenAbvGr	TotRmsAbvGrd	Fireplaces	\
0	2	1	3	1	8	0	
1	2	0	3	1	6	1	

	GarageYrBlt	GarageCars	GarageArea	WoodDeckSF	OpenPorchSF	\
0	2003.0	2	548	0	61	
1	1976.0	2	460	298	0	

	EnclosedPorch	3SsnPorch	ScreenPorch	PoolArea	MiscVal	MoSold	YrSold	\
0	0	0	0	0	0	2	2008	
1	0	0	0	0	0	5	2007	

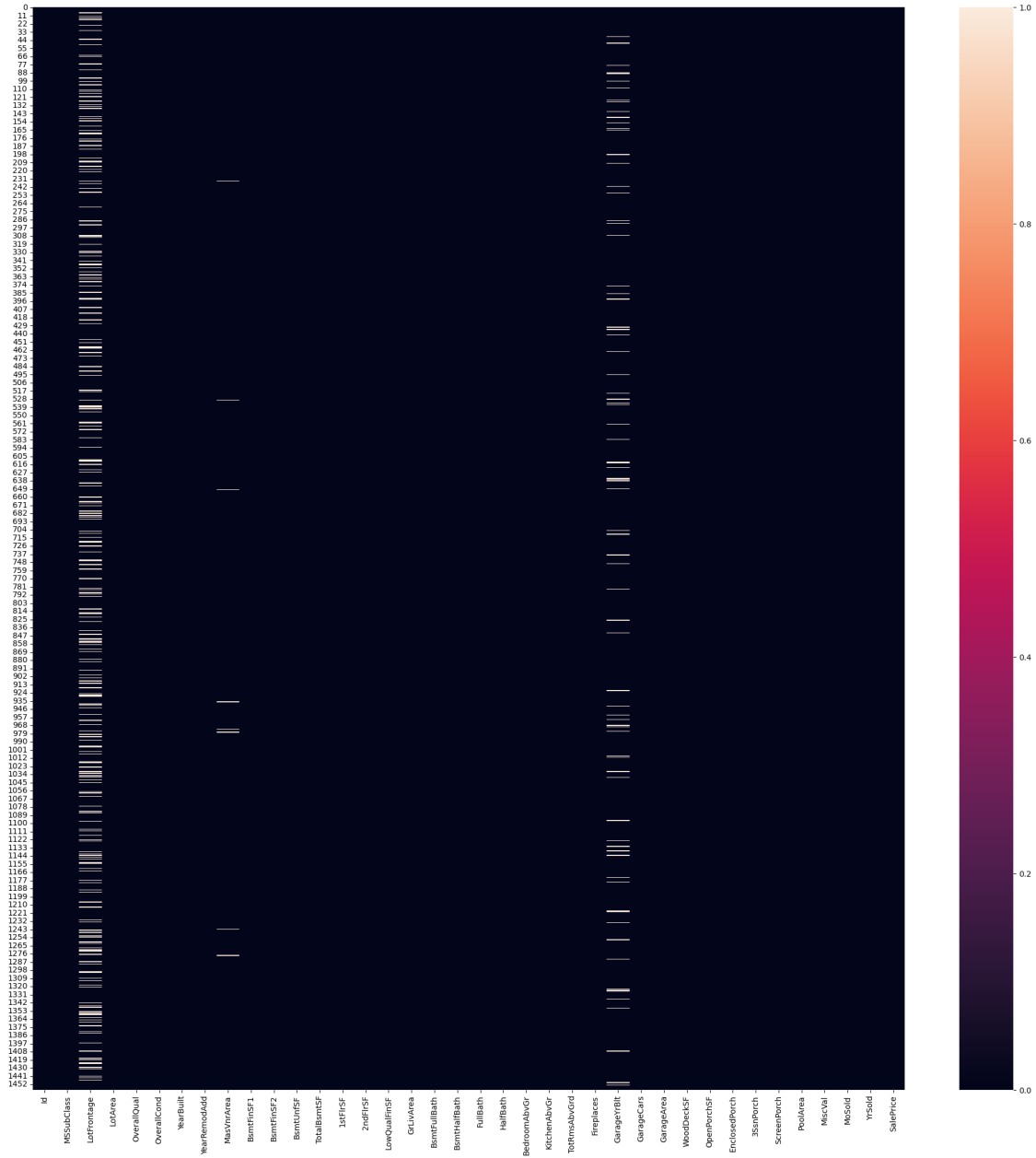
	SalePrice
0	208500
1	181500

```
[18]: numerical_DataFrame.shape
# Here, the numerical_DataFrame contains 1460 rows and 38 columns.
```

```
[18]: (1460, 38)
```

```
[19]: plt.figure(figsize=(25,25))
sns.heatmap(numerical_DataFrame.isnull())
plt.show()

# The code is used to get visualization of the missing values of numerical_
↳ DataFrame.
```



```
[63]: numerical_DataFrame_null_Rows = numerical_DataFrame[numerical_DataFrame.
      ↪ isnull().any(axis=1)]
numerical_DataFrame_null_Rows.head(5)

'''
The code `numerical_DataFrame[numerical_DataFrame.isnull().any(axis=1)]` is
↪ used to filter rows in a DataFrame
called `numerical_DataFrame` that contain at least one null value.
```

'''

```
[63]:
```

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	\
7	8	60	NaN	10382	7	6	1973	
12	13	20	NaN	12968	5	6	1962	
14	15	20	NaN	10920	6	5	1960	
16	17	20	NaN	11241	6	7	1970	
24	25	20	NaN	8246	5	8	1968	

	YearRemodAdd	MasVnrArea	BsmtFinSF1	BsmtFinSF2	BsmtUnfSF	TotalBsmtSF	\
7	1973	240.0	859	32	216	1107	
12	1962	0.0	737	0	175	912	
14	1960	212.0	733	0	520	1253	
16	1970	180.0	578	0	426	1004	
24	2001	0.0	188	668	204	1060	

	1stFlrSF	2ndFlrSF	LowQualFinSF	GrLivArea	BsmtFullBath	BsmtHalfBath	\
7	1107	983	0	2090	1	0	
12	912	0	0	912	1	0	
14	1253	0	0	1253	1	0	
16	1004	0	0	1004	1	0	
24	1060	0	0	1060	1	0	

	FullBath	HalfBath	BedroomAbvGr	KitchenAbvGr	TotRmsAbvGrd	Fireplaces	\
7	2	1	3	1	7	2	
12	1	0	2	1	4	0	
14	1	1	2	1	5	1	
16	1	0	2	1	5	1	
24	1	0	3	1	6	1	

	GarageYrBlt	GarageCars	GarageArea	WoodDeckSF	OpenPorchSF	\
7	1973.0	2	484	235	204	
12	1962.0	1	352	140	0	
14	1960.0	1	352	0	213	
16	1970.0	2	480	0	0	
24	1968.0	1	270	406	90	

	EnclosedPorch	3SsnPorch	ScreenPorch	PoolArea	MiscVal	MoSold	YrSold	\
7	228	0	0	0	350	11	2009	
12	0	0	176	0	0	9	2008	
14	176	0	0	0	0	5	2008	
16	0	0	0	0	700	3	2010	
24	0	0	0	0	0	5	2010	

	SalePrice
7	200000
12	144000

```
14    157000
16    149000
24    154000
```

```
[22]: num_DF_MissingValue = numerical_DataFrame.isnull().sum()
      print(num_DF_MissingValue)
```

```
Id                0
MSSubClass        0
LotFrontage      259
LotArea           0
OverallQual       0
OverallCond       0
YearBuilt         0
YearRemodAdd      0
MasVnrArea        8
BsmtFinSF1        0
BsmtFinSF2        0
BsmtUnfSF         0
TotalBsmtSF       0
1stFlrSF          0
2ndFlrSF          0
LowQualFinSF      0
GrLivArea         0
BsmtFullBath      0
BsmtHalfBath      0
FullBath          0
HalfBath          0
BedroomAbvGr     0
KitchenAbvGr     0
TotRmsAbvGrd     0
Fireplaces        0
GarageYrBlt       81
GarageCars        0
GarageArea        0
WoodDeckSF        0
OpenPorchSF       0
EnclosedPorch     0
3SsnPorch         0
ScreenPorch       0
PoolArea          0
MiscVal           0
MoSold            0
YrSold            0
SalePrice         0
dtype: int64
```

```
[23]: num_DF_MissingValue_columns = num_DF_MissingValue[num_DF_MissingValue > 0].
      ↪keys()
      print(num_DF_MissingValue_columns)

      '''
      This code is used to retrieve the column names from a DataFrame called
      ↪num_DF_MissingValue where
      the corresponding values are greater than zero (indicating missing values).

      '''
```

```
Index(['LotFrontage', 'MasVnrArea', 'GarageYrBlt'], dtype='object')
```

```
[23]: '\nThis code is used to retrieve the column names from a DataFrame called
      num_DF_MissingValue where \nthe corresponding values are greater than zero
      (indicating missing values).\n\n'
```

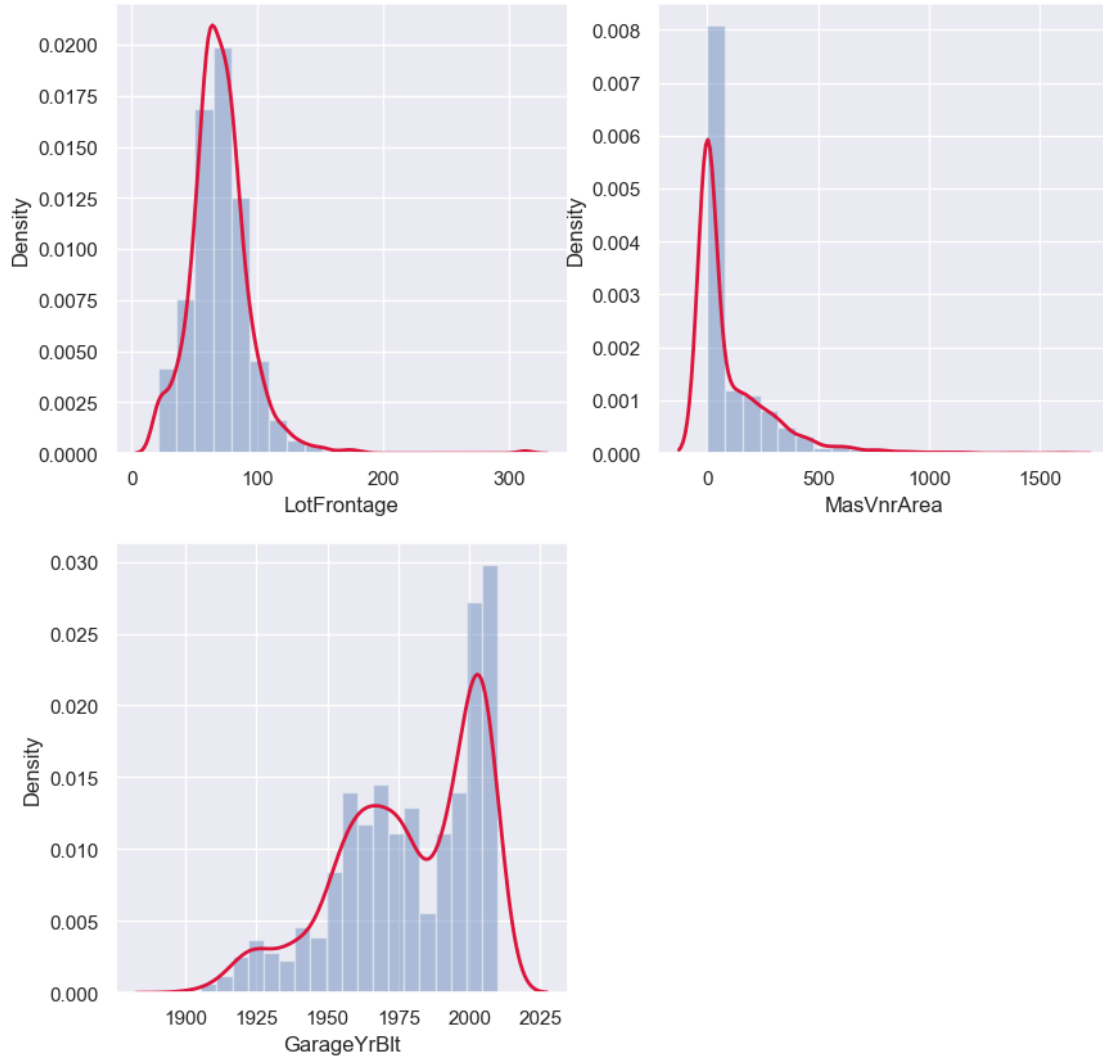
The code creates a figure with a specific size, sets the seaborn style, and then iterates over the `num_DF_MissingValue_columns`. For each column, it creates a subplot and plots the distribution of the corresponding column from the numerical `DataFrame` using the seaborn `distplot` function. The `bins` parameter specifies the number of bins in the histogram, and the `kde_kws` parameter sets the properties of the kernel density estimation line (linewidth and color).

```
[24]: # Visualization of Missing Numeric Data Distribution

      plt.figure(figsize=(10,10))

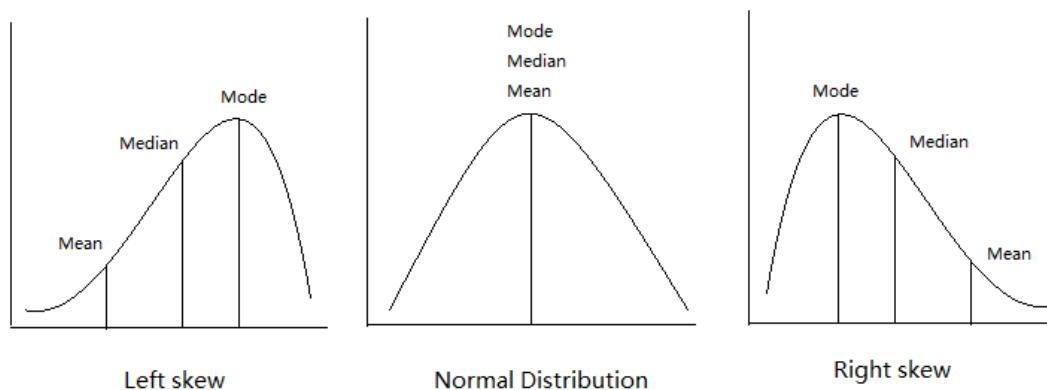
      sns.set()

      for i, var in enumerate(num_DF_MissingValue_columns):
          plt.subplot(2,2,i+1)
          sns.distplot(numerical_DataFrame[var], bins=20, kde_kws={'linewidth':2,
          ↪'color': '#DC143C'})
```

```
[25]: Image('C:/Users/Server/Desktop/NIBZZ/Intellipaat/SQL Project/img.png')
```

```
[25]:
```



When we compare the visualization with the image, we can observe that LotFrontage follows an almost normal distribution, MasVnrArea is skewed to the right, and GarageYrBlt is skewed to the left. Therefore, we use the mean as a measure of central tendency for LotFrontage, and the mode as a measure of central tendency for the other two variables.

Filling Missing Values in “LotFrontage” Column with Mean as Measure of Central Tendency :

```
[65]: numerical_DataFrame_copy1 = numerical_DataFrame.copy() # Creating a copy of
      ↪ numerical_DataFrame
      numerical_DataFrame_copy1.shape
```

```
[65]: (1460, 38)
```

```
[66]: lotfrontage_mean = numerical_DataFrame_copy1['LotFrontage'].mean() #
      ↪ Calculating mean of column LotFrontage and printing it.
      print('LotFrontage Mean: ',lotfrontage_mean)
```

```
LotFrontage Mean: 70.04995836802665
```

```
[69]: numerical_DataFrame_copy1['LotFrontage'].fillna(LotFrontage_mean,inplace=True)
      numerical_DataFrame_copy1[num_DF_MissingValue_columns].isnull().sum()
      # Filling the missing value LotFrontage column with it's mean and checking if
      ↪ the missing values of that column is handled.
```

```
[69]: LotFrontage      0
      MasVnrArea      8
      GarageYrBlt     81
      dtype: int64
```

Filling Missing Values in “MasVnrArea” and “GarageYrBlt” Column with Median as Measure of Central Tendency :

```
[70]: MasVnrArea_Median = numerical_DataFrame_copy1['MasVnrArea'].median()
print("MasVnrArea_Median: ", MasVnrArea_Median)

# Finding the median of column MasVnrArea and printing it.
```

MasVnrArea_Median: 0.0

```
[71]: GarageYrBlt_Median = numerical_DataFrame_copy1['GarageYrBlt'].median()
print("GarageYrBlt_Median: ",GarageYrBlt_Median)

# Finding the median of column GarageYrBlt and printing it.
```

GarageYrBlt_Median: 1980.0

```
[72]: numerical_DataFrame_copy1['MasVnrArea'].fillna(MasVnrArea_Median, inplace=True)
numerical_DataFrame_copy1['GarageYrBlt'].fillna(GarageYrBlt_Median,
↪inplace=True)

# Filling the missing value MasVnrArea, GarageYrBlt columns with it's median.
```

```
[73]: numerical_DataFrame_copy1[num_DF_MissingValue_columns].isnull().sum()

# Checking if the missing values of that column is handled.
```

```
[73]: LotFrontage    0
MasVnrArea        0
GarageYrBlt      0
dtype: int64
```

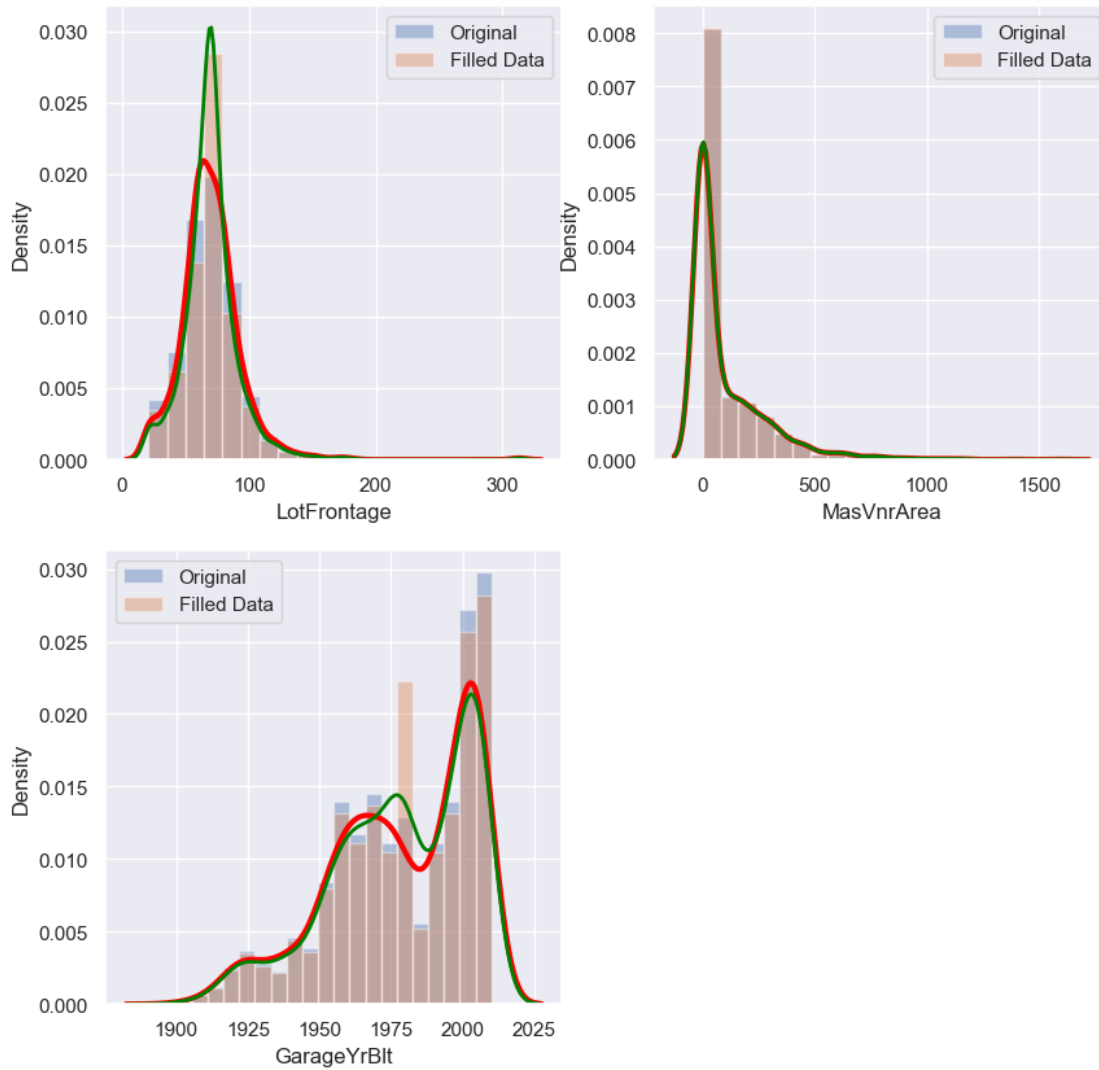
Visualizing Original and Filled Data Distributions for Numerical DataFrame :

```
[74]: plt.figure(figsize=(10,10))

sns.set()

for i, var in enumerate(num_DF_MissingValue_columns):
    plt.subplot(2,2,i+1)
    sns.distplot(numerical_DataFrame[var], bins=20, kde_kws={'linewidth':3,
↪'color':'red'}, label="Original",)
    sns.distplot(numerical_DataFrame_copy1[var], bins=20, kde_kws={'linewidth':
↪2, 'color':'green'},label="Filled Data",)

plt.legend()
```



Concatenating Numerical DataFrames to Fill Missing Values :

```
[75]: df_concat = pd.
      ↪ concat([numerical_DataFrame[num_DF_MissingValue_columns], numerical_DataFrame_copy1[num_DF_M
```

```
[76]: df_concat_DF = df_concat[df_concat.isnull().any(axis=1)]
      df_concat_DF.head(5)
```

```
[76]:
```

	LotFrontage	MasVnrArea	GarageYrBlt	LotFrontage	MasVnrArea	GarageYrBlt
7	NaN	240.0	1973.0	70.049958	240.0	1973.0
12	NaN	0.0	1962.0	70.049958	0.0	1962.0
14	NaN	212.0	1960.0	70.049958	212.0	1960.0
16	NaN	180.0	1970.0	70.049958	180.0	1970.0
24	NaN	0.0	1968.0	70.049958	0.0	1968.0

Based on the information provided above, it is evident that the missing value in the LotFrontage column has been replaced with the mean value of the LotFrontage column. Similarly, the missing values in the MasVnrArea and GarageYrBlt columns have been replaced with the median values of the MasVnrArea and GarageYrBlt columns respectively.

central-tendency-for-each-class-1

June 11, 2023

1 Data Cleaning :

1.1 Missing value imputation of numerical column by Measure of central tendency for each class:

Importing necessary libraries

```
[1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

data (Original) :

```
[2]: data = pd.read_csv("train.csv") # Loading dataset
```

```
[3]: data.shape data.shape # Checking the shape of the dataset, dataset contains
↳ 1460 rows and 81 columns
```

```
[3]: (1460, 81)
```

```
[4]: data.head() # Checking first 5 rows from the DataFrame.
```

```
[4]:
```

	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]

When we check the shape of the data using the command “data.shape”, we can observe that the DataFrame consists of 1460 rows and 81 columns. However, when we examine the first five rows of the DataFrame using the command “data.head(5)”, it doesn’t display all 81 columns. As a result, we utilize the following code, specifically “pd.set_option()”, to address this issue.

```
[6]: pd.set_option('display.max_columns',None)
pd.set_option('display.max_rows',None)
```

pd.set_option(‘display.max_columns’, None): This line sets the maximum number of columns to be displayed in the output to None, which means there is no limit. As a result, all columns in a DataFrame will be shown when you print or display it.

pd.set_option(‘display.max_rows’, None): This line sets the maximum number of rows to be displayed in the output to None, removing any limit. As a result, all rows in a DataFrame will be shown when you print or display it.

```
[7]: data.head(2) # The code `data.head(2)` displays the first two rows of the
↳dataset, showing all 81 columns.
```

```
[7]:   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

   LandContour  Utilities  LotConfig  LandSlope  Neighborhood  Condition1  \
0           Lvl    AllPub    Inside     Gtl      CollgCr      Norm
1           Lvl    AllPub     FR2     Gtl      Veenker      Feedr

   Condition2  BldgType  HouseStyle  OverallQual  OverallCond  YearBuilt  \
0          Norm     1Fam     2Story           7             5         2003
1          Norm     1Fam     1Story           6             8         1976

   YearRemodAdd  RoofStyle  RoofMatl  Exterior1st  Exterior2nd  MasVnrType  \
0           2003     Gable  CompShg     VinylSd     VinylSd     BrkFace
1           1976     Gable  CompShg     MetalSd     MetalSd         None

   MasVnrArea  ExterQual  ExterCond  Foundation  BsmtQual  BsmtCond  BsmtExposure  \
0         196.0         Gd         TA         PConc         Gd         TA         No
1           0.0         TA         TA         CBlock         Gd         TA         Gd
```

	BsmtFinType1	BsmtFinSF1	BsmtFinType2	BsmtFinSF2	BsmtUnfSF	TotalBsmtSF	\
0	GLQ	706	Unf	0	150	856	
1	ALQ	978	Unf	0	284	1262	

	Heating	HeatingQC	CentralAir	Electrical	1stFlrSF	2ndFlrSF	LowQualFinSF	\
0	GasA	Ex	Y	SBrkr	856	854	0	
1	GasA	Ex	Y	SBrkr	1262	0	0	

	GrLivArea	BsmtFullBath	BsmtHalfBath	FullBath	HalfBath	BedroomAbvGr	\
0	1710	1	0	2	1	3	
1	1262	0	1	2	0	3	

	KitchenAbvGr	KitchenQual	TotRmsAbvGrd	Functional	Fireplaces	FireplaceQu	\
0	1	Gd	8	Typ	0	NaN	
1	1	TA	6	Typ	1	TA	

	GarageType	GarageYrBlt	GarageFinish	GarageCars	GarageArea	GarageQual	\
0	Attchd	2003.0	RFn	2	548	TA	
1	Attchd	1976.0	RFn	2	460	TA	

	GarageCond	PavedDrive	WoodDeckSF	OpenPorchSF	EnclosedPorch	3SsnPorch	\
0	TA	Y	0	61	0	0	
1	TA	Y	298	0	0	0	

	ScreenPorch	PoolArea	PoolQC	Fence	MiscFeature	MiscVal	MoSold	YrSold	\
0	0	0	NaN	NaN	NaN	0	2	2008	
1	0	0	NaN	NaN	NaN	0	5	2007	

	SaleType	SaleCondition	SalePrice
0	WD	Normal	208500
1	WD	Normal	181500

```
[8]: data.tail(2) # The code `data.tail(2)` displays the last two rows of the
↳ dataset.
```

```
[8]:
```

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	\
1458	1459	20	RL	68.0	9717	Pave	NaN	Reg	
1459	1460	20	RL	75.0	9937	Pave	NaN	Reg	

	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Condition1	\
1458	Lvl	AllPub	Inside	Gtl	Names	Norm	
1459	Lvl	AllPub	Inside	Gtl	Edwards	Norm	

	Condition2	BldgType	HouseStyle	OverallQual	OverallCond	YearBuilt	\
1458	Norm	1Fam	1Story	5	6	1950	
1459	Norm	1Fam	1Story	5	6	1965	

	YearRemodAdd	RoofStyle	RoofMatl	Exterior1st	Exterior2nd	MasVnrType	\		
1458	1996	Hip	CompShg	MetalSd	MetalSd	None			
1459	1965	Gable	CompShg	HdBoard	HdBoard	None			
	MasVnrArea	ExterQual	ExterCond	Foundation	BsmtQual	BsmtCond	\		
1458	0.0	TA	TA	CBlock	TA	TA			
1459	0.0	Gd	TA	CBlock	TA	TA			
	BsmtExposure	BsmtFinType1	BsmtFinSF1	BsmtFinType2	BsmtFinSF2	\			
1458	Mn	GLQ	49	Rec	1029				
1459	No	BLQ	830	LwQ	290				
	BsmtUnfSF	TotalBsmtSF	Heating	HeatingQC	CentralAir	Electrical	\		
1458	0	1078	GasA	Gd	Y	FuseA			
1459	136	1256	GasA	Gd	Y	SBrkr			
	1stFlrSF	2ndFlrSF	LowQualFinSF	GrLivArea	BsmtFullBath	BsmtHalfBath	\		
1458	1078	0	0	1078	1	0			
1459	1256	0	0	1256	1	0			
	FullBath	HalfBath	BedroomAbvGr	KitchenAbvGr	KitchenQual	\			
1458	1	0	2	1	Gd				
1459	1	1	3	1	TA				
	TotRmsAbvGrd	Functional	Fireplaces	FireplaceQu	GarageType	GarageYrBlt	\		
1458	5	Typ	0	NaN	Attchd	1950.0			
1459	6	Typ	0	NaN	Attchd	1965.0			
	GarageFinish	GarageCars	GarageArea	GarageQual	GarageCond	PavedDrive	\		
1458	Unf	1	240	TA	TA	Y			
1459	Fin	1	276	TA	TA	Y			
	WoodDeckSF	OpenPorchSF	EnclosedPorch	3SsnPorch	ScreenPorch	\			
1458	366	0	112	0	0				
1459	736	68	0	0	0				
	PoolArea	PoolQC	Fence	MiscFeature	MiscVal	MoSold	YrSold	SaleType	\
1458	0	NaN	NaN	NaN	0	4	2010	WD	
1459	0	NaN	NaN	NaN	0	6	2008	WD	
	SaleCondition	SalePrice	\						
1458	Normal	142125							
1459	Normal	147500							

```
[9]: data.info()
```

```
...
```

The `data.info()` function provides a concise summary of the dataset, including information about the number of rows, columns, data types, memory usage and missing values.

'''

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 81 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Id                    1460 non-null   int64
1   MSSubClass            1460 non-null   int64
2   MSZoning              1460 non-null   object
3   LotFrontage          1201 non-null   float64
4   LotArea               1460 non-null   int64
5   Street               1460 non-null   object
6   Alley                91 non-null     object
7   LotShape              1460 non-null   object
8   LandContour          1460 non-null   object
9   Utilities            1460 non-null   object
10  LotConfig             1460 non-null   object
11  LandSlope            1460 non-null   object
12  Neighborhood          1460 non-null   object
13  Condition1           1460 non-null   object
14  Condition2           1460 non-null   object
15  BldgType             1460 non-null   object
16  HouseStyle           1460 non-null   object
17  OverallQual          1460 non-null   int64
18  OverallCond          1460 non-null   int64
19  YearBuilt            1460 non-null   int64
20  YearRemodAdd         1460 non-null   int64
21  RoofStyle            1460 non-null   object
22  RoofMatl             1460 non-null   object
23  Exterior1st          1460 non-null   object
24  Exterior2nd          1460 non-null   object
25  MasVnrType           1452 non-null   object
26  MasVnrArea           1452 non-null   float64
27  ExterQual            1460 non-null   object
28  ExterCond            1460 non-null   object
29  Foundation           1460 non-null   object
30  BsmtQual             1423 non-null   object
31  BsmtCond             1423 non-null   object
32  BsmtExposure         1422 non-null   object
33  BsmtFinType1         1423 non-null   object
34  BsmtFinSF1           1460 non-null   int64
35  BsmtFinType2         1422 non-null   object
```

36	BsmtFinSF2	1460	non-null	int64
37	BsmtUnfSF	1460	non-null	int64
38	TotalBsmtSF	1460	non-null	int64
39	Heating	1460	non-null	object
40	HeatingQC	1460	non-null	object
41	CentralAir	1460	non-null	object
42	Electrical	1459	non-null	object
43	1stFlrSF	1460	non-null	int64
44	2ndFlrSF	1460	non-null	int64
45	LowQualFinSF	1460	non-null	int64
46	GrLivArea	1460	non-null	int64
47	BsmtFullBath	1460	non-null	int64
48	BsmtHalfBath	1460	non-null	int64
49	FullBath	1460	non-null	int64
50	HalfBath	1460	non-null	int64
51	BedroomAbvGr	1460	non-null	int64
52	KitchenAbvGr	1460	non-null	int64
53	KitchenQual	1460	non-null	object
54	TotRmsAbvGrd	1460	non-null	int64
55	Functional	1460	non-null	object
56	Fireplaces	1460	non-null	int64
57	FireplaceQu	770	non-null	object
58	GarageType	1379	non-null	object
59	GarageYrBlt	1379	non-null	float64
60	GarageFinish	1379	non-null	object
61	GarageCars	1460	non-null	int64
62	GarageArea	1460	non-null	int64
63	GarageQual	1379	non-null	object
64	GarageCond	1379	non-null	object
65	PavedDrive	1460	non-null	object
66	WoodDeckSF	1460	non-null	int64
67	OpenPorchSF	1460	non-null	int64
68	EnclosedPorch	1460	non-null	int64
69	3SsnPorch	1460	non-null	int64
70	ScreenPorch	1460	non-null	int64
71	PoolArea	1460	non-null	int64
72	PoolQC	7	non-null	object
73	Fence	281	non-null	object
74	MiscFeature	54	non-null	object
75	MiscVal	1460	non-null	int64
76	MoSold	1460	non-null	int64
77	YrSold	1460	non-null	int64
78	SaleType	1460	non-null	object
79	SaleCondition	1460	non-null	object
80	SalePrice	1460	non-null	int64

dtypes: float64(3), int64(35), object(43)
memory usage: 924.0+ KB

[9]: '\n\nThe `data.info()` function provides a concise summary of the dataset, including information about the number of rows, \nncolumns, data types, memory usage and missing values.\n\n'

```
[10]: data.isnull().sum()
```

```
[10]: Id                0
      MSSubClass        0
      MSZoning          0
      LotFrontage      259
      LotArea           0
      Street            0
      Alley            1369
      LotShape          0
      LandContour       0
      Utilities         0
      LotConfig         0
      LandSlope         0
      Neighborhood     0
      Condition1        0
      Condition2        0
      BldgType          0
      HouseStyle        0
      OverallQual       0
      OverallCond       0
      YearBuilt         0
      YearRemodAdd      0
      RoofStyle         0
      RoofMatl          0
      Exterior1st       0
      Exterior2nd       0
      MasVnrType        8
      MasVnrArea        8
      ExterQual         0
      ExterCond         0
      Foundation        0
      BsmtQual          37
      BsmtCond          37
      BsmtExposure     38
      BsmtFinType1     37
      BsmtFinSF1        0
      BsmtFinType2     38
      BsmtFinSF2        0
      BsmtUnfSF         0
      TotalBsmtSF       0
      Heating           0
      HeatingQC         0
```

```
CentralAir      0
Electrical      1
1stFlrSF       0
2ndFlrSF       0
LowQualFinSF   0
GrLivArea      0
BsmtFullBath   0
BsmtHalfBath   0
FullBath       0
HalfBath       0
BedroomAbvGr   0
KitchenAbvGr   0
KitchenQual    0
TotRmsAbvGrd   0
Functional     0
Fireplaces     0
FireplaceQu    690
GarageType     81
GarageYrBlt    81
GarageFinish    81
GarageCars     0
GarageArea     0
GarageQual     81
GarageCond     81
PavedDrive     0
WoodDeckSF     0
OpenPorchSF    0
EnclosedPorch  0
3SsnPorch     0
ScreenPorch    0
PoolArea       0
PoolQC         1453
Fence          1179
MiscFeature    1406
MiscVal        0
MoSold         0
YrSold         0
SaleType       0
SaleCondition  0
SalePrice      0
dtype: int64
```

```
[12]: plt.figure(figsize=(25,25))

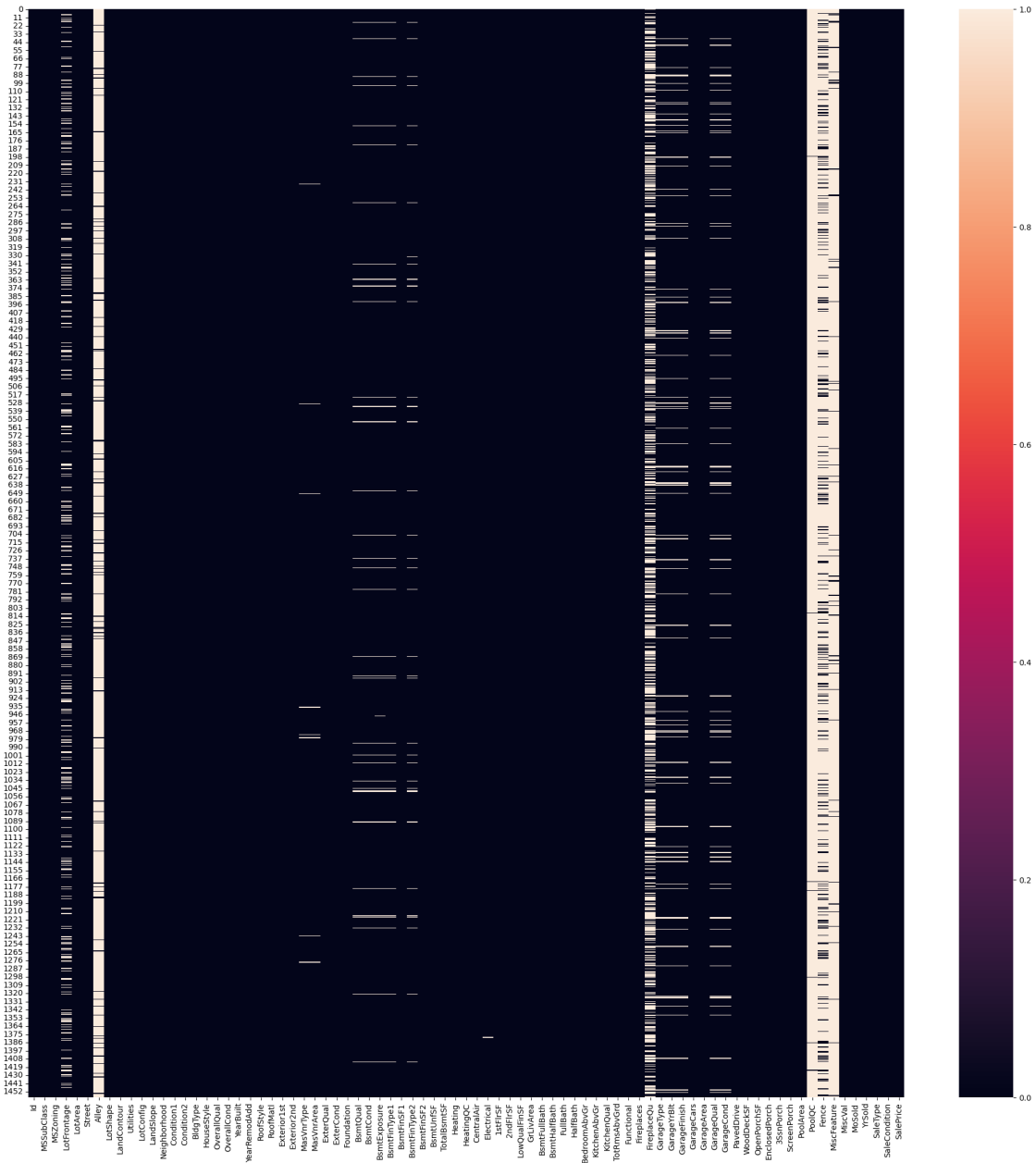
sns.heatmap(data.isnull())
plt.show()
```

'''

The code `plt.figure(figsize=(25,25))` sets the figure size for the plot.

The next line `sns.heatmap(data.isnull())` creates a heatmap visualization of the missing values in the dataset using the seaborn library.

'''



```
[12]: '\n\nThe code `plt.figure(figsize=(25,25))` sets the figure size for the plot.\n\n\nThe next line `sns.heatmap(data.isnull())` creates a heatmap visualization of the missing values in the dataset \nusing the seaborn library.\n\n'
```

```
[15]: missing_value_percent = data.isnull().mean() * 100
print(missing_value_percent)

# The code calculates the percentage of missing values in each column of the
↳ dataset and then prints the resulting percentages.
```

Id	0.000000
MSSubClass	0.000000
MSZoning	0.000000
LotFrontage	17.739726
LotArea	0.000000
Street	0.000000
Alley	93.767123
LotShape	0.000000
LandContour	0.000000
Utilities	0.000000
LotConfig	0.000000
LandSlope	0.000000
Neighborhood	0.000000
Condition1	0.000000
Condition2	0.000000
BldgType	0.000000
HouseStyle	0.000000
OverallQual	0.000000
OverallCond	0.000000
YearBuilt	0.000000
YearRemodAdd	0.000000
RoofStyle	0.000000
RoofMatl	0.000000
Exterior1st	0.000000
Exterior2nd	0.000000
MasVnrType	0.547945
MasVnrArea	0.547945
ExterQual	0.000000
ExterCond	0.000000
Foundation	0.000000
BsmtQual	2.534247
BsmtCond	2.534247
BsmtExposure	2.602740
BsmtFinType1	2.534247
BsmtFinSF1	0.000000
BsmtFinType2	2.602740
BsmtFinSF2	0.000000

BsmtUnfSF	0.000000
TotalBsmtSF	0.000000
Heating	0.000000
HeatingQC	0.000000
CentralAir	0.000000
Electrical	0.068493
1stFlrSF	0.000000
2ndFlrSF	0.000000
LowQualFinSF	0.000000
GrLivArea	0.000000
BsmtFullBath	0.000000
BsmtHalfBath	0.000000
FullBath	0.000000
HalfBath	0.000000
BedroomAbvGr	0.000000
KitchenAbvGr	0.000000
KitchenQual	0.000000
TotRmsAbvGrd	0.000000
Functional	0.000000
Fireplaces	0.000000
FireplaceQu	47.260274
GarageType	5.547945
GarageYrBlt	5.547945
GarageFinish	5.547945
GarageCars	0.000000
GarageArea	0.000000
GarageQual	5.547945
GarageCond	5.547945
PavedDrive	0.000000
WoodDeckSF	0.000000
OpenPorchSF	0.000000
EnclosedPorch	0.000000
3SsnPorch	0.000000
ScreenPorch	0.000000
PoolArea	0.000000
PoolQC	99.520548
Fence	80.753425
MiscFeature	96.301370
MiscVal	0.000000
MoSold	0.000000
YrSold	0.000000
SaleType	0.000000
SaleCondition	0.000000
SalePrice	0.000000
dtype: float64	


```
[16]: missing_value_column = missing_value_percent[missing_value_percent > 20].keys()
print(missing_value_column)
```

```
'''
```

*The code is used to identify the columns in a dataset that have missing values
↳ exceeding 20%.*

*It retrieves the keys (column names) from the `missing_value_percent`
↳ dictionary where the corresponding values
are greater than 20%.*

```
'''
```

```
Index(['Alley', 'FireplaceQu', 'PoolQC', 'Fence', 'MiscFeature'],
      dtype='object')
```

data1 after dropping missing value column > 20% :

```
[17]: data1 = data.drop(columns=missing_value_column)
```

```
[18]: data1.shape # Checking the shape of the DataFrame , DataFrame contains Rows:
↳ 1460 and Columns: 76
```

```
[18]: (1460, 76)
```

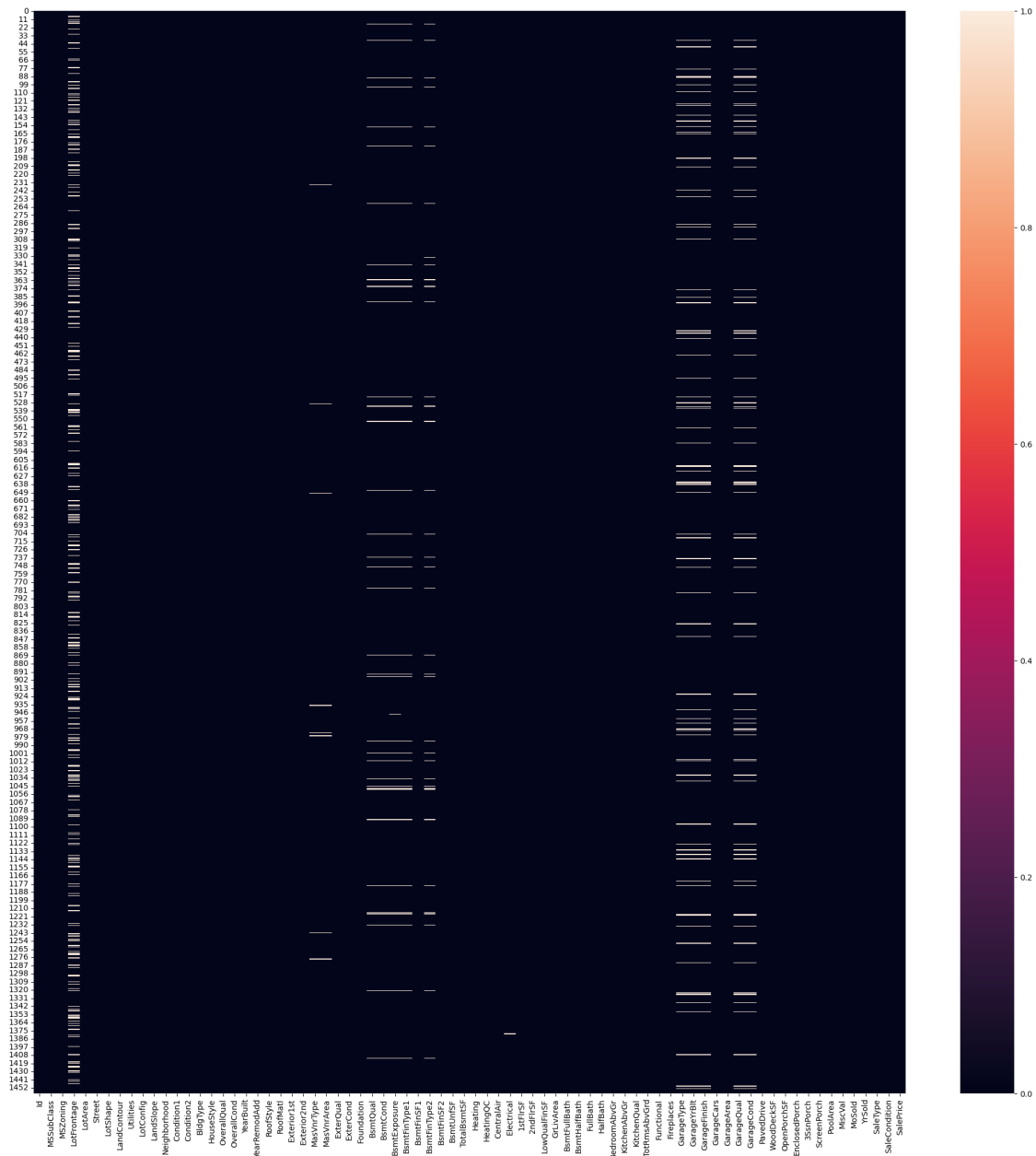
```
[20]: plt.figure(figsize=(25,25))
```

```
sns.heatmap(data1.isnull())
plt.show()
```

```
'''
```

*The code is used to get visualization of the missing values after removing
↳ columns
where the missing values exceed 17% of the total data in those columns.*

```
'''
```



Our objective is to eliminate all the white lines from the visualization in order to achieve a cleaner and more organized appearance.

numerical_DataFrame (selecting DataFrame that contains only numerical data types)

```

:
[22]: numerical_dataframe = data.select_dtypes(include=['int64', 'float64'])
      numerical_dataframe.head(2)

```

```

[22]:  Id  MSSubClass  LotFrontage  LotArea  OverallQual  OverallCond  YearBuilt  \
0     1           60           65.0    8450           7           5         2003
1     2           20           80.0    9600           6           8         1976

      YearRemodAdd  MasVnrArea  BsmtFinSF1  BsmtFinSF2  BsmtUnfSF  TotalBsmtSF  \
0           2003           196.0          706           0           150           856
1           1976            0.0           978           0           284          1262

      1stFlrSF  2ndFlrSF  LowQualFinSF  GrLivArea  BsmtFullBath  BsmtHalfBath  \
0           856           854            0          1710            1            0
1          1262            0            0          1262            0            1

      FullBath  HalfBath  BedroomAbvGr  KitchenAbvGr  TotRmsAbvGrd  Fireplaces  \
0             2           1             3             1             8             0
1             2           0             3             1             6             1

      GarageYrBlt  GarageCars  GarageArea  WoodDeckSF  OpenPorchSF  \
0           2003.0            2           548            0            61
1           1976.0            2           460           298            0

      EnclosedPorch  3SsnPorch  ScreenPorch  PoolArea  MiscVal  MoSold  YrSold  \
0                 0            0            0            0            0            2    2008
1                 0            0            0            0            0            5    2007

      SalePrice
0       208500
1       181500

```

```

[23]: numerical_dataframe.shape # Here, the numerical_DataFrame contains 1460 rows
      ↪and 38 columns.

```

```

[23]: (1460, 38)

```

```

[27]: Numerical_DF_missing_value = numerical_dataframe.isnull().sum() # Checking the
      ↪null values of numerical DataFrame.
      Numerical_DF_missing_value

```

```

[27]: Id                0
      MSSubClass        0
      LotFrontage       259
      LotArea           0
      OverallQual        0
      OverallCond        0
      YearBuilt          0
      YearRemodAdd       0
      MasVnrArea         8
      BsmtFinSF1         0

```

```

BsmtFinSF2      0
BsmtUnfSF       0
TotalBsmtSF     0
1stFlrSF        0
2ndFlrSF        0
LowQualFinSF    0
GrLivArea       0
BsmtFullBath    0
BsmtHalfBath    0
FullBath        0
HalfBath        0
BedroomAbvGr   0
KitchenAbvGr   0
TotRmsAbvGrd   0
Fireplaces      0
GarageYrBlt     81
GarageCars      0
GarageArea      0
WoodDeckSF      0
OpenPorchSF     0
EnclosedPorch   0
3SsnPorch       0
ScreenPorch     0
PoolArea        0
MiscVal         0
MoSold          0
YrSold          0
SalePrice       0
dtype: int64

```

```

[30]: Num_DF_missing_value_columns =
      ↪ Numerical_DF_missing_value[Numerical_DF_missing_value > 0].keys()
      Num_DF_missing_value_columns

      # Checking columns names of numerical_dataframe having missing value > 0

```

```

[30]: Index(['LotFrontage', 'MasVnrArea', 'GarageYrBlt'], dtype='object')

```

```

[33]: Missing_col_DF =
      ↪ numerical_dataframe[Num_DF_missing_value_columns][numerical_dataframe[Num_DF_missing_value_
      ↪ isnull().any(axis=1)]
      Missing_col_DF.head()

      # Having look at columns having missing value

```

```

[33]:   LotFrontage  MasVnrArea  GarageYrBlt
      7           NaN         240.0         1973.0

```

12	NaN	0.0	1962.0
14	NaN	212.0	1960.0
16	NaN	180.0	1970.0
24	NaN	0.0	1968.0

In this process, we impute missing values in a numerical column by using a measure of central tendency for each class. To find the missing value, we need to consider a categorical column that is related to it. This requires having domain knowledge. Then, we proceed to impute the missing value by taking the mean or median, depending on the different categories within the categorical column.

Handling missing value by class :

```
[81]: data1.head()
```

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

```
Utilities  LotConfig  LandSlope  Neighborhood  Condition1  Condition2  BldgType  \
0  AllPub      Inside      Gtl          CollgCr      Norm        Norm        1Fam
1  AllPub      FR2         Gtl          Veenker      Feedr       Norm        1Fam
2  AllPub      Inside      Gtl          CollgCr      Norm        Norm        1Fam
3  AllPub      Corner     Gtl          Crawfor      Norm        Norm        1Fam
4  AllPub      FR2         Gtl          NoRidge     Norm        Norm        1Fam
```

```
HouseStyle  OverallQual  OverallCond  YearBuilt  YearRemodAdd  RoofStyle  \
0  2Story      7           5          2003         2003        Gable
1  1Story      6           8          1976         1976        Gable
2  2Story      7           5          2001         2002        Gable
3  2Story      7           5          1915         1970        Gable
4  2Story      8           5          2000         2000        Gable
```

```
RoofMatl  Exterior1st  Exterior2nd  MasVnrType  MasVnrArea  ExterQual  ExterCond  \
0  CompShg     VinylSd     VinylSd     BrkFace     196.0      Gd         TA
1  CompShg     MetalSd     MetalSd     None         0.0        TA         TA
2  CompShg     VinylSd     VinylSd     BrkFace     162.0      Gd         TA
3  CompShg     Wd Sdng    Wd Shng     None         0.0        TA         TA
4  CompShg     VinylSd     VinylSd     BrkFace     350.0      Gd         TA
```

```
Foundation  BsmtQual  BsmtCond  BsmtExposure  BsmtFinType1  BsmtFinSF1  \
0  PConc     Gd        TA           No             GLQ          706
1  CBlock    Gd        TA           Gd             ALQ          978
2  PConc     Gd        TA           Mn             GLQ          486
3  BrkTil    TA        Gd           No             ALQ          216
```

	PConc	Gd	TA	Av	GLQ	655	
	BsmtFinType2	BsmtFinSF2	BsmtUnfSF	TotalBsmtSF	Heating	HeatingQC	\
0	Unf	0	150	856	GasA	Ex	
1	Unf	0	284	1262	GasA	Ex	
2	Unf	0	434	920	GasA	Ex	
3	Unf	0	540	756	GasA	Gd	
4	Unf	0	490	1145	GasA	Ex	
	CentralAir	Electrical	1stFlrSF	2ndFlrSF	LowQualFinSF	GrLivArea	\
0	Y	SBrkr	856	854	0	1710	
1	Y	SBrkr	1262	0	0	1262	
2	Y	SBrkr	920	866	0	1786	
3	Y	SBrkr	961	756	0	1717	
4	Y	SBrkr	1145	1053	0	2198	
	BsmtFullBath	BsmtHalfBath	FullBath	HalfBath	BedroomAbvGr	KitchenAbvGr	\
0	1	0	2	1	3	1	
1	0	1	2	0	3	1	
2	1	0	2	1	3	1	
3	1	0	1	0	3	1	
4	1	0	2	1	4	1	
	KitchenQual	TotRmsAbvGrd	Functional	Fireplaces	GarageType	GarageYrBlt	\
0	Gd	8	Typ	0	Attchd	2003.0	
1	TA	6	Typ	1	Attchd	1976.0	
2	Gd	6	Typ	1	Attchd	2001.0	
3	Gd	7	Typ	1	Detchd	1998.0	
4	Gd	9	Typ	1	Attchd	2000.0	
	GarageFinish	GarageCars	GarageArea	GarageQual	GarageCond	PavedDrive	\
0	RFn	2	548	TA	TA	Y	
1	RFn	2	460	TA	TA	Y	
2	RFn	2	608	TA	TA	Y	
3	Unf	3	642	TA	TA	Y	
4	RFn	3	836	TA	TA	Y	
	WoodDeckSF	OpenPorchSF	EnclosedPorch	3SsnPorch	ScreenPorch	PoolArea	\
0	0	61	0	0	0	0	
1	298	0	0	0	0	0	
2	0	42	0	0	0	0	
3	0	35	272	0	0	0	
4	192	84	0	0	0	0	
	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

To analyze missing value of LotFrontage let's analyze LotConfig column.

```
[82]: data1['LotConfig'].unique() # Here, we check different categories under
↳LotConfig column.
```

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

```
[83]: mean_lotfrontage = data1[data1.loc[:, 'LotConfig'] == "Inside"]["LotFrontage"].
↳replace(np.nan, data1[data1.loc[:, 'LotConfig'] == "Inside"]["LotFrontage"].
↳mean())
mean_lotfrontage.head(10)
```

```
[83]: 0      65.000000
      2      68.000000
      5      85.000000
      6      75.000000
      8      51.000000
     10      70.000000
     11      85.000000
     12      67.715686
     13      91.000000
     17      72.000000
Name: LotFrontage, dtype: float64
```

```
[87]: data1_copy_1 = data1.copy()
data1_copy_1.isnull().sum()
```

```
[87]: Id                0
      MSSubClass        0
      MSZoning          0
      LotFrontage       259
      LotArea           0
      Street            0
      LotShape          0
      LandContour       0
      Utilities         0
      LotConfig         0
      LandSlope         0
      Neighborhood     0
      Condition1        0
      Condition2        0
      BldgType          0
      HouseStyle        0
```

OverallQual	0
OverallCond	0
YearBuilt	0
YearRemodAdd	0
RoofStyle	0
RoofMatl	0
Exterior1st	0
Exterior2nd	0
MasVnrType	8
MasVnrArea	8
ExterQual	0
ExterCond	0
Foundation	0
BsmtQual	37
BsmtCond	37
BsmtExposure	38
BsmtFinType1	37
BsmtFinSF1	0
BsmtFinType2	38
BsmtFinSF2	0
BsmtUnfSF	0
TotalBsmtSF	0
Heating	0
HeatingQC	0
CentralAir	0
Electrical	1
1stFlrSF	0
2ndFlrSF	0
LowQualFinSF	0
GrLivArea	0
BsmtFullBath	0
BsmtHalfBath	0
FullBath	0
HalfBath	0
BedroomAbvGr	0
KitchenAbvGr	0
KitchenQual	0
TotRmsAbvGrd	0
Functional	0
Fireplaces	0
GarageType	81
GarageYrBlt	81
GarageFinish	81
GarageCars	0
GarageArea	0
GarageQual	81
GarageCond	81


```

PavedDrive      0
WoodDeckSF      0
OpenPorchSF     0
EnclosedPorch   0
3SsnPorch       0
ScreenPorch     0
PoolArea        0
MiscVal         0
MoSold          0
YrSold          0
SaleType        0
SaleCondition    0
SalePrice       0
dtype: int64

```

```

[88]: for var_class in data1['LotConfig'].unique():
        data1_copy_1.update(data1[data1.loc[:, 'LotConfig'] ==
        ↪var_class]['LotFrontage'].replace(np.nan, data1[data1.loc[:, 'LotConfig'] ==
        ↪var_class]['LotFrontage'].mean()))

# This code performs missing value imputation in the "LotFrontage" column based
        ↪on different categories in the "LotConfig" column.

```

```

[91]: data1_copy_1[Num_DF_missing_value_columns].isnull().sum()

# We can see that missing value of LotFrontage column is handled

```

```

[91]: LotFrontage      0
MasVnrArea          8
GarageYrBlt        81
dtype: int64

```

Similarly, we will handle missing value of other numerical columns.

```

[92]: data1_copy2_mean = data1.copy()
data1_copy2_mean.head(10)

```

```

[92]:   Id  MSSubClass  MSZoning  LotFrontage  LotArea  Street  LotShape  LandContour  \
0    1           60         RL           65.0     8450   Pave      Reg          Lvl
1    2           20         RL           80.0     9600   Pave      Reg          Lvl
2    3           60         RL           68.0    11250   Pave      IR1          Lvl
3    4           70         RL           60.0     9550   Pave      IR1          Lvl
4    5           60         RL           84.0    14260   Pave      IR1          Lvl
5    6           50         RL           85.0    14115   Pave      IR1          Lvl
6    7           20         RL           75.0    10084   Pave      Reg          Lvl
7    8           60         RL           NaN     10382   Pave      IR1          Lvl
8    9           50         RM           51.0     6120   Pave      Reg          Lvl

```

9 10 190 RL 50.0 7420 Pave Reg Lvl

	Utilities	LotConfig	LandSlope	Neighborhood	Condition1	Condition2	BldgType	\
0	AllPub	Inside	Gtl	CollgCr	Norm	Norm	1Fam	
1	AllPub	FR2	Gtl	Veenker	Feedr	Norm	1Fam	
2	AllPub	Inside	Gtl	CollgCr	Norm	Norm	1Fam	
3	AllPub	Corner	Gtl	Crawfor	Norm	Norm	1Fam	
4	AllPub	FR2	Gtl	NoRidge	Norm	Norm	1Fam	
5	AllPub	Inside	Gtl	Mitchel	Norm	Norm	1Fam	
6	AllPub	Inside	Gtl	Somerst	Norm	Norm	1Fam	
7	AllPub	Corner	Gtl	NWAmes	PosN	Norm	1Fam	
8	AllPub	Inside	Gtl	OldTown	Artery	Norm	1Fam	
9	AllPub	Corner	Gtl	BrkSide	Artery	Artery	2fmCon	

	HouseStyle	OverallQual	OverallCond	YearBuilt	YearRemodAdd	RoofStyle	\
0	2Story	7	5	2003	2003	Gable	
1	1Story	6	8	1976	1976	Gable	
2	2Story	7	5	2001	2002	Gable	
3	2Story	7	5	1915	1970	Gable	
4	2Story	8	5	2000	2000	Gable	
5	1.5Fin	5	5	1993	1995	Gable	
6	1Story	8	5	2004	2005	Gable	
7	2Story	7	6	1973	1973	Gable	
8	1.5Fin	7	5	1931	1950	Gable	
9	1.5Unf	5	6	1939	1950	Gable	

	RoofMatl	Exterior1st	Exterior2nd	MasVnrType	MasVnrArea	ExterQual	ExterCond	\
0	CompShg	VinylSd	VinylSd	BrkFace	196.0	Gd	TA	
1	CompShg	MetalSd	MetalSd	None	0.0	TA	TA	
2	CompShg	VinylSd	VinylSd	BrkFace	162.0	Gd	TA	
3	CompShg	Wd Sdng	Wd Shng	None	0.0	TA	TA	
4	CompShg	VinylSd	VinylSd	BrkFace	350.0	Gd	TA	
5	CompShg	VinylSd	VinylSd	None	0.0	TA	TA	
6	CompShg	VinylSd	VinylSd	Stone	186.0	Gd	TA	
7	CompShg	HdBoard	HdBoard	Stone	240.0	TA	TA	
8	CompShg	BrkFace	Wd Shng	None	0.0	TA	TA	
9	CompShg	MetalSd	MetalSd	None	0.0	TA	TA	

	Foundation	BsmtQual	BsmtCond	BsmtExposure	BsmtFinType1	BsmtFinSF1	\
0	PConc	Gd	TA	No	GLQ	706	
1	CBlock	Gd	TA	Gd	ALQ	978	
2	PConc	Gd	TA	Mn	GLQ	486	
3	BrkTil	TA	Gd	No	ALQ	216	
4	PConc	Gd	TA	Av	GLQ	655	
5	Wood	Gd	TA	No	GLQ	732	
6	PConc	Ex	TA	Av	GLQ	1369	
7	CBlock	Gd	TA	Mn	ALQ	859	

8	BrkTil	TA	TA	No	Unf	0
9	BrkTil	TA	TA	No	GLQ	851

	BsmtFinType2	BsmtFinSF2	BsmtUnfSF	TotalBsmtSF	Heating	HeatingQC	\
0	Unf	0	150	856	GasA	Ex	
1	Unf	0	284	1262	GasA	Ex	
2	Unf	0	434	920	GasA	Ex	
3	Unf	0	540	756	GasA	Gd	
4	Unf	0	490	1145	GasA	Ex	
5	Unf	0	64	796	GasA	Ex	
6	Unf	0	317	1686	GasA	Ex	
7	BLQ	32	216	1107	GasA	Ex	
8	Unf	0	952	952	GasA	Gd	
9	Unf	0	140	991	GasA	Ex	

	CentralAir	Electrical	1stFlrSF	2ndFlrSF	LowQualFinSF	GrLivArea	\
0	Y	SBrkr	856	854	0	1710	
1	Y	SBrkr	1262	0	0	1262	
2	Y	SBrkr	920	866	0	1786	
3	Y	SBrkr	961	756	0	1717	
4	Y	SBrkr	1145	1053	0	2198	
5	Y	SBrkr	796	566	0	1362	
6	Y	SBrkr	1694	0	0	1694	
7	Y	SBrkr	1107	983	0	2090	
8	Y	FuseF	1022	752	0	1774	
9	Y	SBrkr	1077	0	0	1077	

	BsmtFullBath	BsmtHalfBath	FullBath	HalfBath	BedroomAbvGr	KitchenAbvGr	\
0	1	0	2	1	3	1	
1	0	1	2	0	3	1	
2	1	0	2	1	3	1	
3	1	0	1	0	3	1	
4	1	0	2	1	4	1	
5	1	0	1	1	1	1	
6	1	0	2	0	3	1	
7	1	0	2	1	3	1	
8	0	0	2	0	2	2	
9	1	0	1	0	2	2	

	KitchenQual	TotRmsAbvGrd	Functional	Fireplaces	GarageType	GarageYrBlt	\
0	Gd	8	Typ	0	Attchd	2003.0	
1	TA	6	Typ	1	Attchd	1976.0	
2	Gd	6	Typ	1	Attchd	2001.0	
3	Gd	7	Typ	1	Detchd	1998.0	
4	Gd	9	Typ	1	Attchd	2000.0	
5	TA	5	Typ	0	Attchd	1993.0	
6	Gd	7	Typ	1	Attchd	2004.0	

7	TA	7	Typ	2	Attchd	1973.0
8	TA	8	Min1	2	Detchd	1931.0
9	TA	5	Typ	2	Attchd	1939.0

	GarageFinish	GarageCars	GarageArea	GarageQual	GarageCond	PavedDrive	\
0	RFn	2	548	TA	TA	Y	
1	RFn	2	460	TA	TA	Y	
2	RFn	2	608	TA	TA	Y	
3	Unf	3	642	TA	TA	Y	
4	RFn	3	836	TA	TA	Y	
5	Unf	2	480	TA	TA	Y	
6	RFn	2	636	TA	TA	Y	
7	RFn	2	484	TA	TA	Y	
8	Unf	2	468	Fa	TA	Y	
9	RFn	1	205	Gd	TA	Y	

	WoodDeckSF	OpenPorchSF	EnclosedPorch	3SsnPorch	ScreenPorch	PoolArea	\
0	0	61	0	0	0	0	
1	298	0	0	0	0	0	
2	0	42	0	0	0	0	
3	0	35	272	0	0	0	
4	192	84	0	0	0	0	
5	40	30	0	320	0	0	
6	255	57	0	0	0	0	
7	235	204	228	0	0	0	
8	90	0	205	0	0	0	
9	0	4	0	0	0	0	

	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	700	10	2009	WD	Normal	143000
6	0	8	2007	WD	Normal	307000
7	350	11	2009	WD	Normal	200000
8	0	4	2008	WD	Abnorml	129900
9	0	1	2008	WD	Normal	118000

1.2 Mean :

```
[95]: num_vars_miss = ['LotFrontage', 'MasVnrArea', 'GarageYrBlt']
cat_vars = ['LotConfig', 'Exterior2nd', 'KitchenQual']

for cat_var, num_var_miss in zip(cat_vars, num_vars_miss):
    for var_class in data[cat_var].unique():
```

```
data1_copy2_mean.update(data[data.loc[:, cat_var] ==  
↪var_class][num_var_miss].replace(np.nan, data[data.loc[:, cat_var] ==  
↪var_class][num_var_miss].mean()))
```

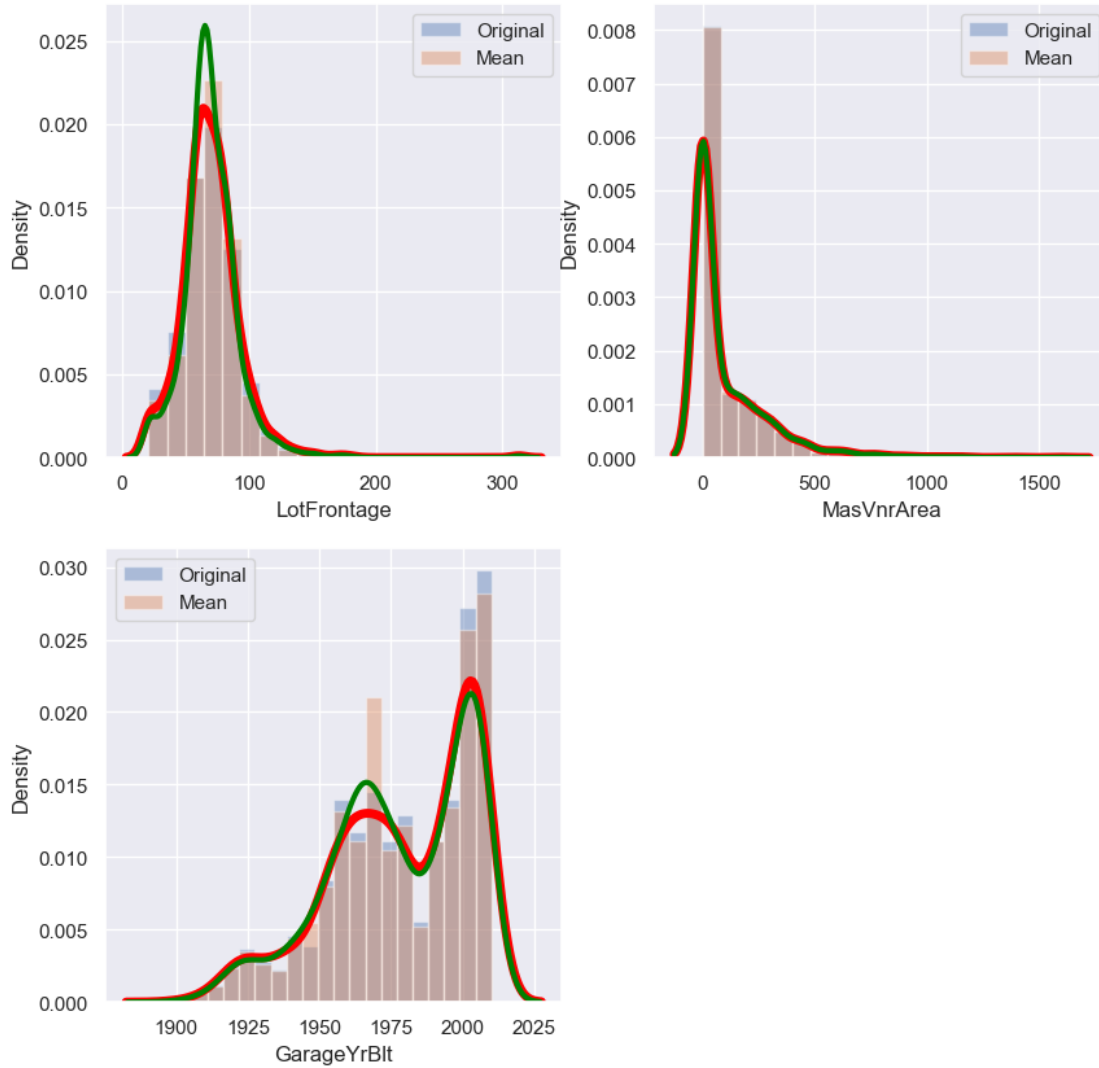
```
[97]: data1_copy2_mean[num_vars_miss].isnull().sum()
```

```
[97]: LotFrontage    0  
MasVnrArea        0  
GarageYrBlt       0  
dtype: int64
```

Here, we have addressed the issue of missing values in the numerical column by calculating the mean for each category in the categorical column.

1.2.1 Data Distribution :

```
[99]: plt.figure(figsize=(10,10))  
sns.set()  
for i, var in enumerate(num_vars_miss):  
    plt.subplot(2,2,i+1)  
    sns.distplot(data[var], bins=20, kde_kws={'linewidth':5, 'color':'red'},  
↪label="Original",)  
    sns.distplot(data1_copy2_mean[var], bins=20, kde_kws={'linewidth':3,  
↪'color':'green'},label="Mean",)  
    plt.legend()
```



1.3 Median :

```
[100]: data1_copy2_median = data1.copy()
data1_copy2_median.head()
```

```
[100]:
```

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	LotShape	LandContour	\
0	1	60	RL	65.0	8450	Pave	Reg	Lvl1	
1	2	20	RL	80.0	9600	Pave	Reg	Lvl1	
2	3	60	RL	68.0	11250	Pave	IR1	Lvl1	
3	4	70	RL	60.0	9550	Pave	IR1	Lvl1	
4	5	60	RL	84.0	14260	Pave	IR1	Lvl1	

	Utilities	LotConfig	LandSlope	Neighborhood	Condition1	Condition2	BldgType	\
0	AllPub	Inside	Gtl	CollgCr	Norm	Norm	1Fam	

1	AllPub	FR2	Gt1	Veenker	Feedr	Norm	1Fam
2	AllPub	Inside	Gt1	CollgCr	Norm	Norm	1Fam
3	AllPub	Corner	Gt1	Crawfor	Norm	Norm	1Fam
4	AllPub	FR2	Gt1	NoRidge	Norm	Norm	1Fam

	HouseStyle	OverallQual	OverallCond	YearBuilt	YearRemodAdd	RoofStyle	\
0	2Story	7	5	2003	2003	Gable	
1	1Story	6	8	1976	1976	Gable	
2	2Story	7	5	2001	2002	Gable	
3	2Story	7	5	1915	1970	Gable	
4	2Story	8	5	2000	2000	Gable	

	RoofMatl	Exterior1st	Exterior2nd	MasVnrType	MasVnrArea	ExterQual	ExterCond	\
0	CompShg	VinylSd	VinylSd	BrkFace	196.0	Gd	TA	
1	CompShg	MetalSd	MetalSd	None	0.0	TA	TA	
2	CompShg	VinylSd	VinylSd	BrkFace	162.0	Gd	TA	
3	CompShg	Wd Sdng	Wd Shng	None	0.0	TA	TA	
4	CompShg	VinylSd	VinylSd	BrkFace	350.0	Gd	TA	

	Foundation	BsmtQual	BsmtCond	BsmtExposure	BsmtFinType1	BsmtFinSF1	\
0	PConc	Gd	TA	No	GLQ	706	
1	CBlock	Gd	TA	Gd	ALQ	978	
2	PConc	Gd	TA	Mn	GLQ	486	
3	BrkTil	TA	Gd	No	ALQ	216	
4	PConc	Gd	TA	Av	GLQ	655	

	BsmtFinType2	BsmtFinSF2	BsmtUnfSF	TotalBsmtSF	Heating	HeatingQC	\
0	Unf	0	150	856	GasA	Ex	
1	Unf	0	284	1262	GasA	Ex	
2	Unf	0	434	920	GasA	Ex	
3	Unf	0	540	756	GasA	Gd	
4	Unf	0	490	1145	GasA	Ex	

	CentralAir	Electrical	1stFlrSF	2ndFlrSF	LowQualFinSF	GrLivArea	\
0	Y	SBrkr	856	854	0	1710	
1	Y	SBrkr	1262	0	0	1262	
2	Y	SBrkr	920	866	0	1786	
3	Y	SBrkr	961	756	0	1717	
4	Y	SBrkr	1145	1053	0	2198	

	BsmtFullBath	BsmtHalfBath	FullBath	HalfBath	BedroomAbvGr	KitchenAbvGr	\
0	1	0	2	1	3	1	
1	0	1	2	0	3	1	
2	1	0	2	1	3	1	
3	1	0	1	0	3	1	
4	1	0	2	1	4	1	

	KitchenQual	TotRmsAbvGrd	Functional	Fireplaces	GarageType	GarageYrBlt	\
0	Gd	8	Typ	0	Attchd	2003.0	
1	TA	6	Typ	1	Attchd	1976.0	
2	Gd	6	Typ	1	Attchd	2001.0	
3	Gd	7	Typ	1	Detchd	1998.0	
4	Gd	9	Typ	1	Attchd	2000.0	

	GarageFinish	GarageCars	GarageArea	GarageQual	GarageCond	PavedDrive	\
0	RFn	2	548	TA	TA	Y	
1	RFn	2	460	TA	TA	Y	
2	RFn	2	608	TA	TA	Y	
3	Unf	3	642	TA	TA	Y	
4	RFn	3	836	TA	TA	Y	

	WoodDeckSF	OpenPorchSF	EnclosedPorch	3SsnPorch	ScreenPorch	PoolArea	\
0	0	61	0	0	0	0	
1	298	0	0	0	0	0	
2	0	42	0	0	0	0	
3	0	35	272	0	0	0	
4	192	84	0	0	0	0	

	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

```
[101]: data1_copy2_median[num_vars_miss].isnull().sum()
```

```
[101]: LotFrontage    259
MasVnrArea         8
GarageYrBlt       81
dtype: int64
```

```
[102]: num_vars_miss = ['LotFrontage', 'MasVnrArea', 'GarageYrBlt']
cat_vars = ['LotConfig', 'Exterior2nd', 'KitchenQual']

for cat_var, num_var_miss in zip(cat_vars, num_vars_miss):
    for var_class in data[cat_var].unique():
        data1_copy2_median.update(data[data.loc[:, cat_var] ==
↳ var_class][num_var_miss].replace(np.nan, data[data.loc[:, cat_var] ==
↳ var_class][num_var_miss].median()))
```

```
[103]: data1_copy2_median[num_vars_miss].isnull().sum()
```

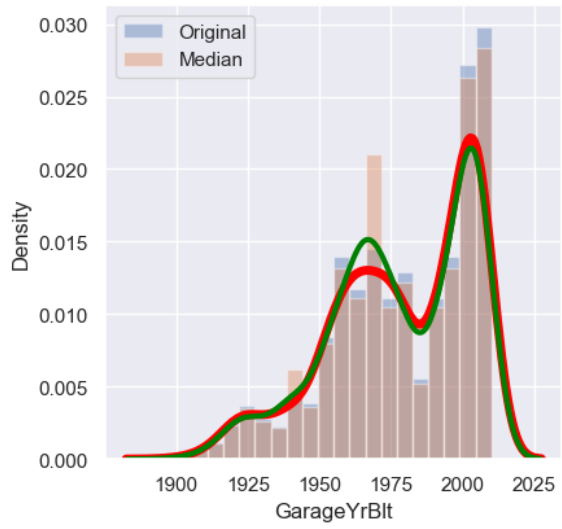
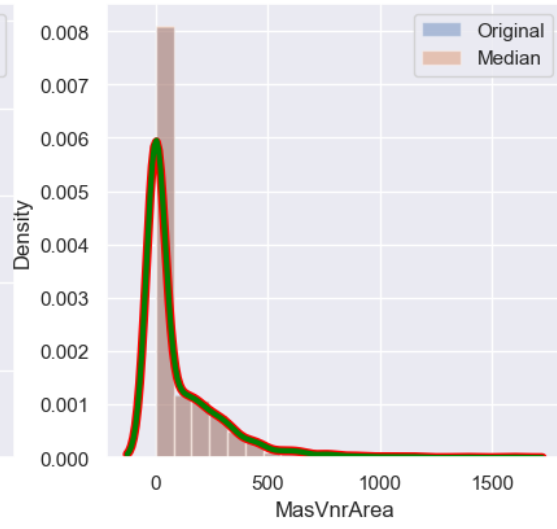
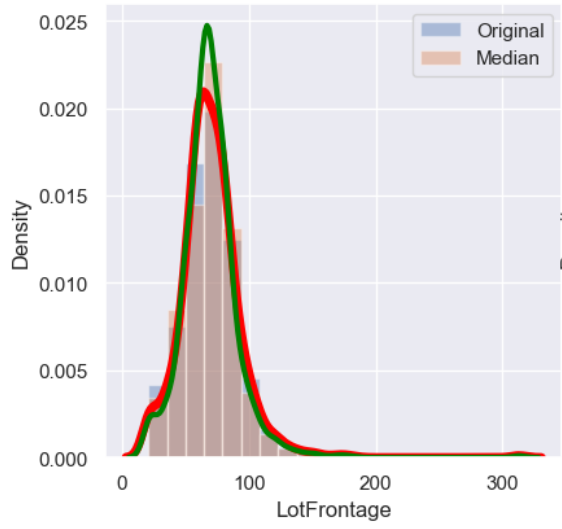


```
[103]: LotFrontage    0
      MasVnrArea    0
      GarageYrBlt   0
      dtype: int64
```

Here, we have addressed the issue of missing values in the numerical column by calculating the mean for each category in the categorical column.

1.3.1 Data Distribution :

```
[104]: plt.figure(figsize=(10,10))
      sns.set()
      for i, var in enumerate(num_vars_miss):
          plt.subplot(2,2,i+1)
          sns.distplot(data[var], bins=20, kde_kws={'linewidth':5, 'color':'red'},
          ↪label="Original",)
          sns.distplot(data1_copy2_median[var], bins=20, kde_kws={'linewidth':3,
          ↪'color':'green'},label="Median",)
          plt.legend()
```



categorical-missing-value-imputation

June 11, 2023

1 Data Cleaning :

1.1 Categorical Missing value imputation :

Importing necessary libraries :

```
[1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
from IPython.display import Image
```

data (Original) :

```
[2]: data = pd.read_csv('train.csv')
```

```
[3]: data.shape # Checking the shape of the DataFrame, DataFrame contains Rows:
↳1460 and Columns: 81.
```

```
[3]: (1460, 81)
```

```
[4]: data.head(2) # Checking first 2 rows from the DataFrame.
```

```
[4]:   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

      LandContour Utilities  ... PoolArea PoolQC Fence MiscFeature MiscVal MoSold \
0           Lvl1   AllPub  ...         0   NaN   NaN           NaN         0     2
1           Lvl1   AllPub  ...         0   NaN   NaN           NaN         0     5

      YrSold  SaleType  SaleCondition  SalePrice
0     2008         WD           Normal    208500
1     2007         WD           Normal    181500
```

```
[2 rows x 81 columns]
```

When we check the shape of the data using the command “data.shape”, we can observe that the DataFrame consists of 1460 rows and 81 columns. However, when we examine the first two rows of the DataFrame using the command “data.head(2)”, it doesn’t display all 81 columns. As a result, we utilize the following code, specifically “pd.set_option()”, to address this issue.

```
[5]: pd.set_option('display.max_column', None)
pd.set_option('display.max_rows', None)
```

pd.set_option(‘display.max_columns’, None): This line sets the maximum number of columns to be displayed in the output to None, which means there is no limit. As a result, all columns in a DataFrame will be shown when you print or display it.

pd.set_option(‘display.max_rows’, None): This line sets the maximum number of rows to be displayed in the output to None, removing any limit. As a result, all rows in a DataFrame will be shown when you print or display it.

```
[7]: categorical_DataFrame = data.select_dtypes(include='object') # selecting only
↳ categorical columns
```

```
[8]: categorical_DataFrame.head() # Checking first 5 rows of categorical DataFrame.
```

```
[8]: MSZoning Street Alley LotShape LandContour Utilities LotConfig LandSlope \
0      RL   Pave   NaN      Reg          Lvl   AllPub   Inside   Gtl
1      RL   Pave   NaN      Reg          Lvl   AllPub    FR2     Gtl
2      RL   Pave   NaN      IR1         Lvl   AllPub   Inside   Gtl
3      RL   Pave   NaN      IR1         Lvl   AllPub   Corner   Gtl
4      RL   Pave   NaN      IR1         Lvl   AllPub    FR2     Gtl

Neighborhood Condition1 Condition2 BldgType HouseStyle RoofStyle RoofMatl \
0   CollgCr      Norm      Norm      1Fam      2Story   Gable   CompShg
1   Veenker      Feedr      Norm      1Fam      1Story   Gable   CompShg
2   CollgCr      Norm      Norm      1Fam      2Story   Gable   CompShg
3   Crawfor      Norm      Norm      1Fam      2Story   Gable   CompShg
4   NoRidge      Norm      Norm      1Fam      2Story   Gable   CompShg

Exterior1st Exterior2nd MasVnrType ExterQual ExterCond Foundation BsmtQual \
0   VinylSd      VinylSd   BrkFace      Gd        TA        PConc      Gd
1   MetalSd      MetalSd      None        TA        TA        CBlock     Gd
2   VinylSd      VinylSd   BrkFace      Gd        TA        PConc      Gd
3   Wd Sdng      Wd Shng      None        TA        TA        BrkTil     TA
4   VinylSd      VinylSd   BrkFace      Gd        TA        PConc      Gd

BsmtCond BsmtExposure BsmtFinType1 BsmtFinType2 Heating HeatingQC \
0      TA            No            GLQ          Unf        GasA      Ex
1      TA            Gd            ALQ          Unf        GasA      Ex
2      TA            Mn            GLQ          Unf        GasA      Ex
3      Gd            No            ALQ          Unf        GasA      Gd
4      TA            Av            GLQ          Unf        GasA      Ex
```

	CentralAir	Electrical	KitchenQual	Functional	FireplaceQu	GarageType	\
0	Y	SBrkr	Gd	Typ	NaN	Attchd	
1	Y	SBrkr	TA	Typ	TA	Attchd	
2	Y	SBrkr	Gd	Typ	TA	Attchd	
3	Y	SBrkr	Gd	Typ	Gd	Detchd	
4	Y	SBrkr	Gd	Typ	TA	Attchd	

	GarageFinish	GarageQual	GarageCond	PavedDrive	PoolQC	Fence	MiscFeature	\
0	RFn	TA	TA	Y	NaN	NaN	NaN	
1	RFn	TA	TA	Y	NaN	NaN	NaN	
2	RFn	TA	TA	Y	NaN	NaN	NaN	
3	Unf	TA	TA	Y	NaN	NaN	NaN	
4	RFn	TA	TA	Y	NaN	NaN	NaN	

	SaleType	SaleCondition
0	WD	Normal
1	WD	Normal
2	WD	Normal
3	WD	Abnorml
4	WD	Normal

```
[9]: categorical_DataFrame.isnull().sum()
```

```
# It used to calculate the number of missing values in each column of a ↵  
↵DataFrame called categorical_DataFrame.
```

```
[9]: MSZoning          0  
Street              0  
Alley              1369  
LotShape           0  
LandContour       0  
Utilities         0  
LotConfig         0  
LandSlope         0  
Neighborhood      0  
Condition1        0  
Condition2        0  
BldgType          0  
HouseStyle        0  
RoofStyle         0  
RoofMatl          0  
Exterior1st       0  
Exterior2nd       0  
MasVnrType        8  
ExterQual         0  
ExterCond         0
```

```

Foundation      0
BsmtQual        37
BsmtCond        37
BsmtExposure    38
BsmtFinType1    37
BsmtFinType2    38
Heating         0
HeatingQC       0
CentralAir      0
Electrical      1
KitchenQual     0
Functional      0
FireplaceQu     690
GarageType      81
GarageFinish    81
GarageQual      81
GarageCond      81
PavedDrive     0
PoolQC         1453
Fence          1179
MiscFeature     1406
SaleType        0
SaleCondition   0
dtype: int64

```

```

[11]: missing_value_percent = categorical_DataFrame.isnull().mean()*100
missing_value_percent

#calculates the percentage of missing values in each column of the
↪categorical_DataFrame.

```

```

[11]: MSZoning      0.000000
Street          0.000000
Alley          93.767123
LotShape       0.000000
LandContour    0.000000
Utilities      0.000000
LotConfig      0.000000
LandSlope      0.000000
Neighborhood   0.000000
Condition1     0.000000
Condition2     0.000000
BldgType       0.000000
HouseStyle     0.000000
RoofStyle      0.000000
RoofMatl       0.000000
Exterior1st    0.000000

```

```

Exterior2nd      0.000000
MasVnrType       0.547945
ExterQual        0.000000
ExterCond        0.000000
Foundation       0.000000
BsmtQual         2.534247
BsmtCond         2.534247
BsmtExposure     2.602740
BsmtFinType1     2.534247
BsmtFinType2     2.602740
Heating          0.000000
HeatingQC        0.000000
CentralAir       0.000000
Electrical       0.068493
KitchenQual      0.000000
Functional       0.000000
FireplaceQu     47.260274
GarageType       5.547945
GarageFinish     5.547945
GarageQual       5.547945
GarageCond       5.547945
PavedDrive       0.000000
PoolQC          99.520548
Fence            80.753425
MiscFeature      96.301370
SaleType         0.000000
SaleCondition    0.000000
dtype: float64

```

```
[15]: drop_column = missing_value_percent[missing_value_percent > 20].keys()
print(drop_column)
```

```
'''
```

The code `drop_column = missing_value_percent[missing_value_percent > 20].keys()` selects the columns from `missing_value_percent` that have missing value percentages greater than 20% and assigns them to the `drop_column` variable.

```
'''
```

```
Index(['Alley', 'FireplaceQu', 'PoolQC', 'Fence', 'MiscFeature'],
      dtype='object')
```

```
[21]: new_data = categorical_DataFrame.drop(columns = drop_column)
new_data.head()
```

'''

The code `new_data = categorical_DataFrame.drop(columns=drop_column)` creates a new DataFrame called `new_data` by dropping the columns specified in the `drop_column` variable from the `categorical_DataFrame`

'''

```
[21]: MSZoning Street LotShape LandContour Utilities LotConfig LandSlope \
0      RL   Pave      Reg          Lvl   AllPub   Inside   Gtl
1      RL   Pave      Reg          Lvl   AllPub    FR2     Gtl
2      RL   Pave      IR1         Lvl   AllPub   Inside   Gtl
3      RL   Pave      IR1         Lvl   AllPub   Corner   Gtl
4      RL   Pave      IR1         Lvl   AllPub    FR2     Gtl

Neighborhood Condition1 Condition2 BldgType HouseStyle RoofStyle RoofMatl \
0   CollgCr      Norm      Norm    1Fam    2Story   Gable   CompShg
1   Veenker      Feedr      Norm    1Fam    1Story   Gable   CompShg
2   CollgCr      Norm      Norm    1Fam    2Story   Gable   CompShg
3   Crawfor      Norm      Norm    1Fam    2Story   Gable   CompShg
4   NoRidge      Norm      Norm    1Fam    2Story   Gable   CompShg

Exterior1st Exterior2nd MasVnrType ExterQual ExterCond Foundation BsmtQual \
0   VinylSd      VinylSd   BrkFace    Gd      TA      PConc    Gd
1   MetalSd      MetalSd   None       TA      TA      CBlock   Gd
2   VinylSd      VinylSd   BrkFace    Gd      TA      PConc    Gd
3   Wd Sdng      Wd Shng   None       TA      TA      BrkTil   TA
4   VinylSd      VinylSd   BrkFace    Gd      TA      PConc    Gd

BsmtCond BsmtExposure BsmtFinType1 BsmtFinType2 Heating HeatingQC \
0      TA          No          GLQ          Unf      GasA      Ex
1      TA          Gd          ALQ          Unf      GasA      Ex
2      TA          Mn          GLQ          Unf      GasA      Ex
3      Gd          No          ALQ          Unf      GasA      Gd
4      TA          Av          GLQ          Unf      GasA      Ex

CentralAir Electrical KitchenQual Functional GarageType GarageFinish \
0      Y          SBrkr      Gd          Typ      Attchd    RFn
1      Y          SBrkr      TA          Typ      Attchd    RFn
2      Y          SBrkr      Gd          Typ      Attchd    RFn
3      Y          SBrkr      Gd          Typ      Detchd    Unf
4      Y          SBrkr      Gd          Typ      Attchd    RFn

GarageQual GarageCond PavedDrive SaleType SaleCondition
0      TA          TA          Y          WD          Normal
1      TA          TA          Y          WD          Normal
2      TA          TA          Y          WD          Normal
```


3	TA	TA	Y	WD	Abnorml
4	TA	TA	Y	WD	Normal

```
[24]: Missing_val_columns = new_data.isnull().sum()
print(Missing_val_columns)

'''
The code Missing_val_columns = new_data.isnull().sum() calculates the number of
missing values in each column of the
new_data DataFrame and assigns the result to the Missing_val_columns variable.
'''
```

MSZoning	0
Street	0
LotShape	0
LandContour	0
Utilities	0
LotConfig	0
LandSlope	0
Neighborhood	0
Condition1	0
Condition2	0
BldgType	0
HouseStyle	0
RoofStyle	0
RoofMatl	0
Exterior1st	0
Exterior2nd	0
MasVnrType	8
ExterQual	0
ExterCond	0
Foundation	0
BsmtQual	37
BsmtCond	37
BsmtExposure	38
BsmtFinType1	37
BsmtFinType2	38
Heating	0
HeatingQC	0
CentralAir	0
Electrical	1
KitchenQual	0
Functional	0
GarageType	81
GarageFinish	81
GarageQual	81
GarageCond	81

```
PavedDrive      0
SaleType        0
SaleCondition   0
dtype: int64
```

```
[26]: Null_column = Missing_val_columns[Missing_val_columns > 0].keys()
print(Null_column)

'''
The code `Null_column = Missing_val_columns[Missing_val_columns > 0].keys()`
↳selects the columns from the
`Missing_val_columns` variable that have a count of missing values greater than
↳0 and assigns their column names to the
`Null_column` variable.
'''
```

```
Index(['MasVnrType', 'BsmtQual', 'BsmtCond', 'BsmtExposure', 'BsmtFinType1',
      'BsmtFinType2', 'Electrical', 'GarageType', 'GarageFinish',
      'GarageQual', 'GarageCond'],
      dtype='object')
```

```
[27]: new_data_copy = new_data.copy()
new_data_copy.shape
```

```
[27]: (1460, 38)
```

```
[28]: for var in Null_column:
        new_data_copy[var].fillna(new_data_copy[var].mode()[0], inplace = True)
        print(var, '=', new_data[var].mode()[0])
```

```
MasVnrType = None
BsmtQual = TA
BsmtCond = TA
BsmtExposure = No
BsmtFinType1 = Unf
BsmtFinType2 = Unf
Electrical = SBrkr
GarageType = Attchd
GarageFinish = Unf
GarageQual = TA
GarageCond = TA
```

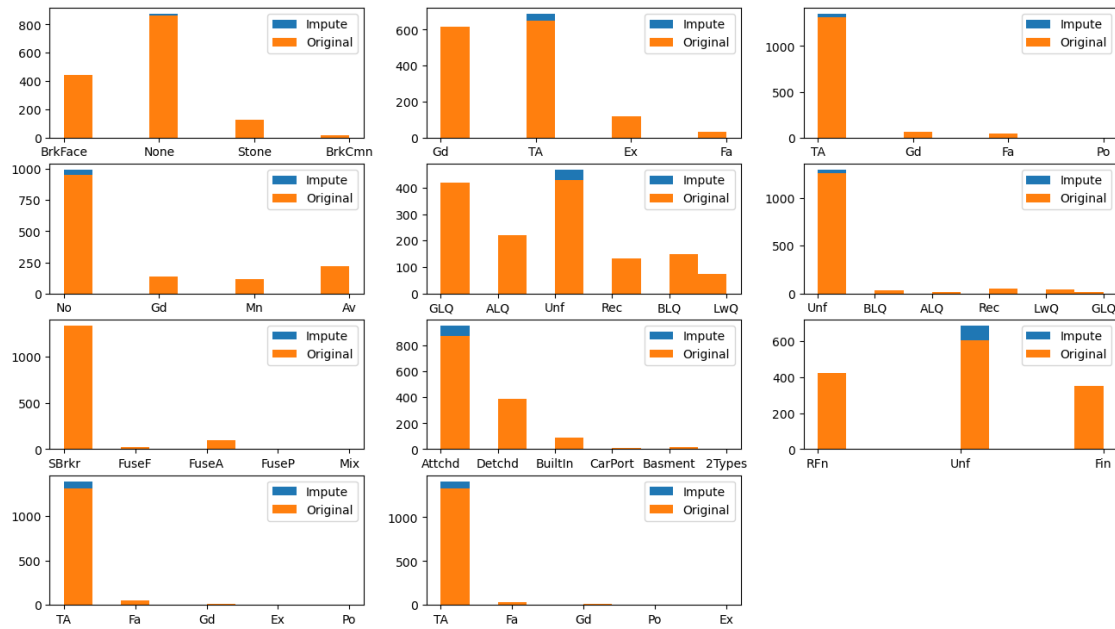
This loop iterates over each column name stored in `Null_column`, fills the missing values in that column with the mode (most frequent value), and then prints the column name along with the mode value. The `fillna` function is used to replace the missing values with the specified mode value in place.

```
[30]: new_data_copy.isnull().sum() # We can see null value is handled
```

```
[30]: MSZoning      0
      Street      0
      LotShape    0
      LandContour 0
      Utilities   0
      LotConfig   0
      LandSlope   0
      Neighborhood 0
      Condition1  0
      Condition2  0
      BldgType    0
      HouseStyle  0
      RoofStyle   0
      RoofMatl    0
      Exterior1st 0
      Exterior2nd 0
      MasVnrType  0
      ExterQual   0
      ExterCond   0
      Foundation  0
      BsmtQual    0
      BsmtCond    0
      BsmtExposure 0
      BsmtFinType1 0
      BsmtFinType2 0
      Heating     0
      HeatingQC   0
      CentralAir  0
      Electrical  0
      KitchenQual 0
      Functional  0
      GarageType  0
      GarageFinish 0
      GarageQual  0
      GarageCond  0
      PavedDrive  0
      SaleType    0
      SaleCondition 0
      dtype: int64
```

```
[31]: plt.figure(figsize=(16,9))
      for i,var in enumerate(Null_column):
          plt.subplot(4,3,i+1)
          plt.hist(new_data_copy[var],label="Impute")
          plt.hist(new_data[var].dropna(),label="Original")
```

```
plt.legend()
```



The code you provided is creating a figure and subplots using `plt.subplot()` to visualize the distribution of the imputed values (`new_data_copy[var]`) compared to the original values (`new_data[var]`) for each column in `Null_column`.