Data Handling And Preproccessing using Python

```
import pandas as pd
import numpy as np

from sklearn.preprocessing import MinMaxScaler

data = pd.read_csv('train.csv')
df = pd.DataFrame(data)
df.head()
```

```
   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]
```

```
df.isnull().sum()
```

```
Id                0
MSSubClass        0
MSZoning          0
```

```
LotFrontage      259
LotArea            0
                 ...
MoSold             0
YrSold             0
SaleType           0
SaleCondition      0
SalePrice          0
Length: 81, dtype: int64

# Fill numerical columns with median
num_cols = df.select_dtypes(include=np.number).columns
df[num_cols] = df[num_cols].fillna(df[num_cols].median())
```

Normalization (Min-Max Scaling)

```
scaler = MinMaxScaler()
df_normalized = df.copy()
df_normalized[num_cols] = scaler.fit_transform(df[num_cols])

print("After Normalization:")
print(df_normalized[num_cols].head())

After Normalization:
         Id  MSSubClass  LotFrontage   LotArea  OverallQual
OverallCond  \
0  0.000000    0.235294     0.150685  0.033420     0.666667
0.500
1  0.000685    0.000000     0.202055  0.038795     0.555556
0.875
2  0.001371    0.235294     0.160959  0.046507     0.666667
0.500
3  0.002056    0.294118     0.133562  0.038561     0.666667
0.500
4  0.002742    0.235294     0.215753  0.060576     0.777778
0.500

   YearBuilt  YearRemodAdd  MasVnrArea  BsmtFinSF1  ...  WoodDeckSF  \
0   0.949275      0.883333     0.12250    0.125089  ...    0.000000
1   0.753623      0.433333     0.00000    0.173281  ...    0.347725
2   0.934783      0.866667     0.10125    0.086109  ...    0.000000
3   0.311594      0.333333     0.00000    0.038271  ...    0.000000
4   0.927536      0.833333     0.21875    0.116052  ...    0.224037

   OpenPorchSF  EnclosedPorch  3SsnPorch  ScreenPorch  PoolArea
MiscVal  \
0     0.111517       0.000000        0.0          0.0       0.0
0.0
1     0.000000       0.000000        0.0          0.0       0.0
0.0
```

```
2     0.076782       0.000000        0.0           0.0       0.0
0.0
3     0.063985       0.492754        0.0           0.0       0.0
0.0
4     0.153565       0.000000        0.0           0.0       0.0
0.0

     MoSold  YrSold  SalePrice
0  0.090909    0.50   0.241078
1  0.363636    0.25   0.203583
2  0.727273    0.50   0.261908
3  0.090909    0.00   0.145952
4  1.000000    0.50   0.298709

[5 rows x 38 columns]
```

Standardization (Z-Score Scaling)

```
from sklearn.preprocessing import StandardScaler

# Apply Standardization
scaler = StandardScaler()

df_standardized = df.copy()
df_standardized[num_cols] = scaler.fit_transform(df[num_cols])

print("After Standardization:")
print(df_standardized[num_cols].head())

After Standardization:
         Id  MSSubClass  LotFrontage   LotArea  OverallQual
OverallCond  \
0 -1.730865    0.073375    -0.220875 -0.207142     0.651479    -
0.517200
1 -1.728492   -0.872563     0.460320 -0.091886    -0.071836
2.179628
2 -1.726120    0.073375    -0.084636  0.073480     0.651479    -
0.517200
3 -1.723747    0.309859    -0.447940 -0.096897     0.651479    -
0.517200
4 -1.721374    0.073375     0.641972  0.375148     1.374795    -
0.517200

   YearBuilt  YearRemodAdd  MasVnrArea  BsmtFinSF1  ...  WoodDeckSF  \
0   1.050994      0.878668    0.514104    0.575425  ...   -0.752176
1   0.156734     -0.429577   -0.570750    1.171992  ...    1.626195
2   0.984752      0.830215    0.325915    0.092907  ...   -0.752176
3  -1.863632     -0.720298   -0.570750   -0.499274  ...   -0.752176
4   0.951632      0.733308    1.366489    0.463568  ...    0.780197
```

```
   OpenPorchSF  EnclosedPorch  3SsnPorch  ScreenPorch  PoolArea
MiscVal  \
0     0.216503      -0.359325  -0.116339    -0.270208 -0.068692 -
0.087688
1    -0.704483      -0.359325  -0.116339    -0.270208 -0.068692 -
0.087688
2    -0.070361      -0.359325  -0.116339    -0.270208 -0.068692 -
0.087688
3    -0.176048       4.092524  -0.116339    -0.270208 -0.068692 -
0.087688
4     0.563760      -0.359325  -0.116339    -0.270208 -0.068692 -
0.087688

     MoSold    YrSold  SalePrice
0 -1.599111  0.138777   0.347273
1 -0.489110 -0.614439   0.007288
2  0.990891  0.138777   0.536154
3 -1.599111 -1.367655  -0.515281
4  2.100892  0.138777   0.869843

[5 rows x 38 columns]
```