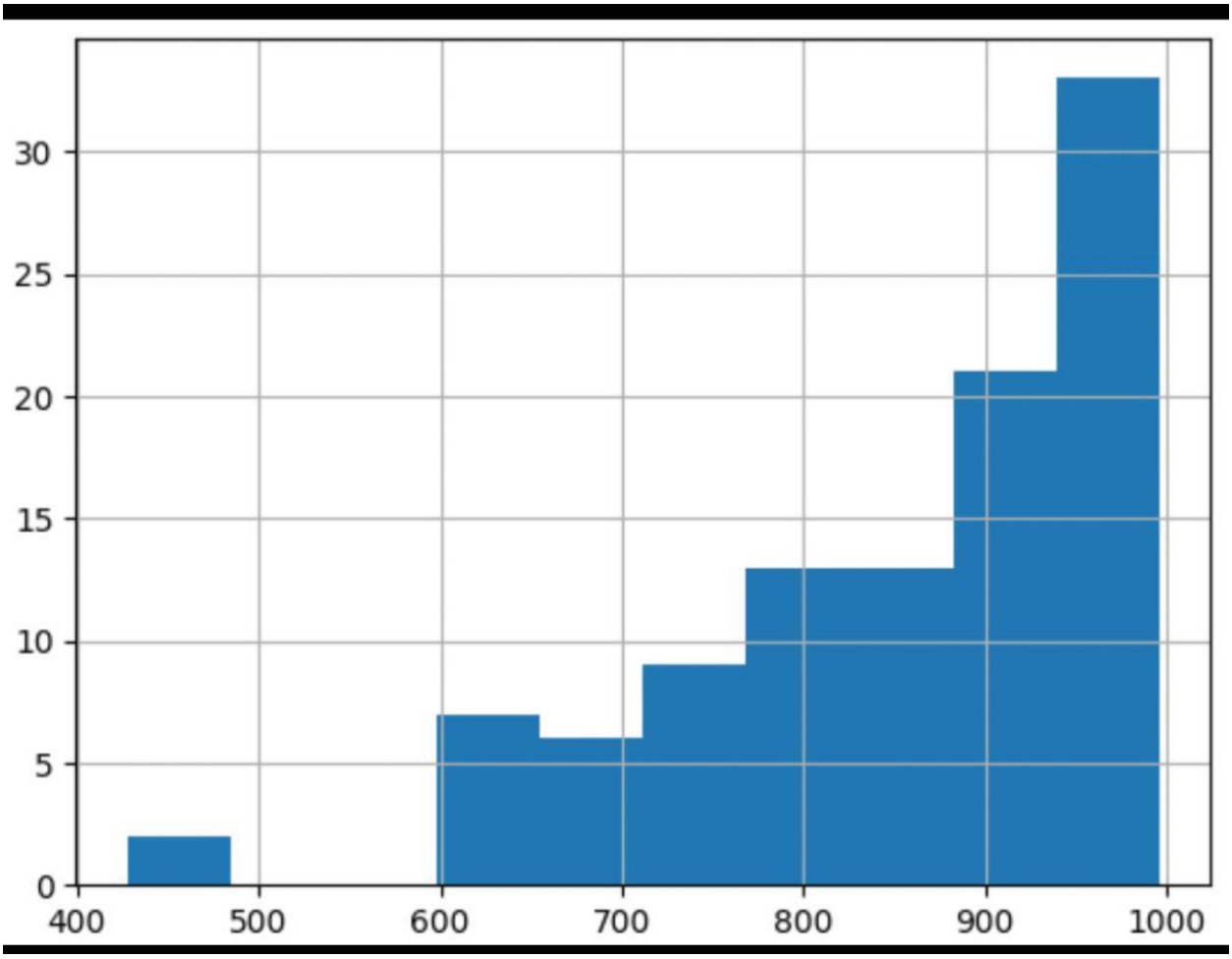
```
import numpy as np
import pandas as pd
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
color = sns.color_palette()
sns.set_style('darkgrid')
import warnings
def ignore_warn(*args, **kwargs):
 pass
warnings.warn = ignore_warn
from scipy import stats
from scipy.stats import norm, skew
pd.set_option('display.float_format', lambda x: '{:.3f}'.format(x))
from subprocess import check_output
print(check_output(["ls", "../input"]).decode("utf8"))
train = pd.read_csv('/kaggle/input/house-prices-advanced-regression-techni
ques/train.csv')
test = pd.read_csv('/kaggle/input/house-prices-advanced-regression-techniq
ues/test.csv')
train.head(5)
fig, ax = plt.subplots()
ax.scatter(x = train['GrLivArea'], y = train['SalePrice'])
plt.ylabel('SalePrice', fontsize=13)
plt.xlabel('GrLivArea', fontsize=13)
plt.show()
train["SalePrice"] = np.log1p(train["SalePrice"])
sns.distplot(train['SalePrice'] , fit=norm);
(mu, sigma) = norm.fit(train['SalePrice'])
```

```
print( '\n mu = {:.2f} and sigma = {:.2f}\n'.format(mu, sigma))

plt.legend(['Normal dist. ($\mu=$ {:.2f} and $\sigma=$ {:.2f} )'.format(mu

, sigma)],

 loc='best')

plt.ylabel('Frequency')

plt.title('SalePrice distribution')

fig = plt.figure()

res = stats.probplot(train['SalePrice'], plot=plt)

plt.show()

import pandas as pd

import numpy as np

Import matplotlib as mpl

Import matplotlib. Pyplot as plt

%matplotlib inline

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler

data=pd.read_csv(‘data.csv’)

data=pd.read_csv(‘usa housing.csv’)

data.dropna() ▪ Remove rows

data.fillna(value) ▪ Replace missing value with our specific value

data.drop_duplicate() ▪ Remove duplicate data

data[‘column_name’].replace(old_value, new_value, inplace=True) ▪ Replace the value of old value
which is no more needed,to a value with

newly update

pd.merge()

df = pd.merge(df1, df2, on = ‘common attribute ‘)

df..head(10)

#Sort by name

sorted = df.sort_values(by=['name'])
```

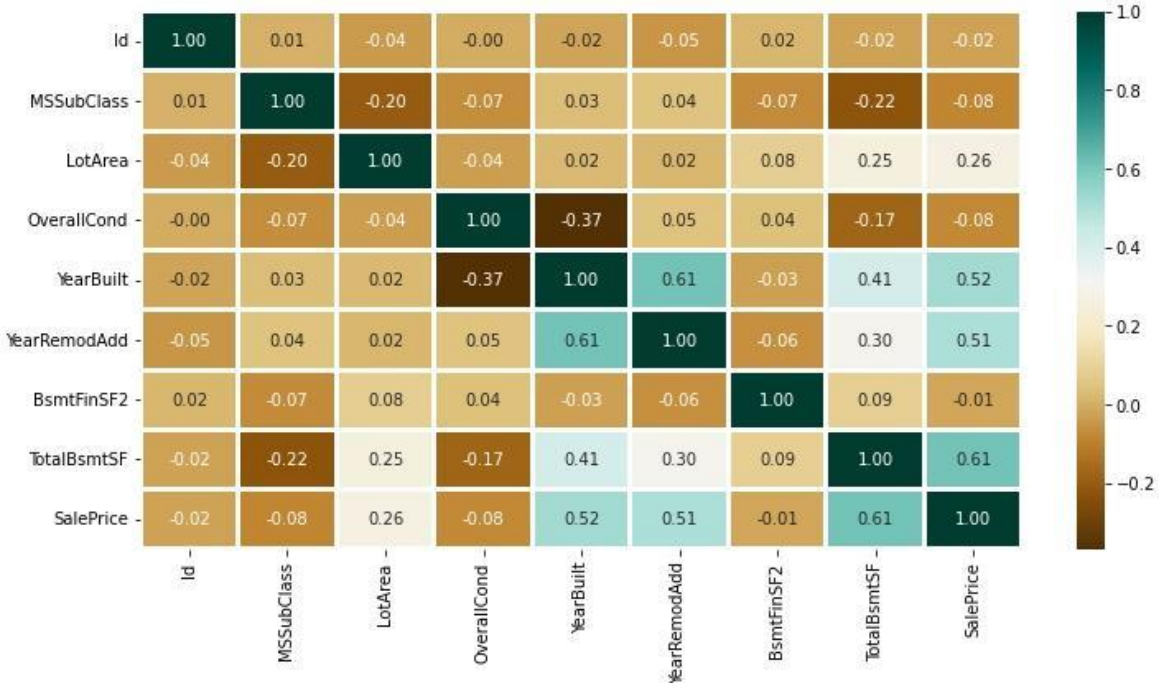
```
display(sorted)
#Filter rows
just_students = df.query('is_student==True')
display(just_students)

#Filter columns
no_birthday = df.filter(['name','is_student','target'])
display(no_birthday)
#Rename column
renamed = df.rename(columns={'target':'target_score'})
display(renamed)
#Splitting
splitnames = df.copy()
split = splitnames['name'].str.split(' ', expand = True)
splitnames['first'] = split[0]
splitnames['last'] = split[1]
display(splitnames)
#Data Value transforms
```

Df['target'].hist()

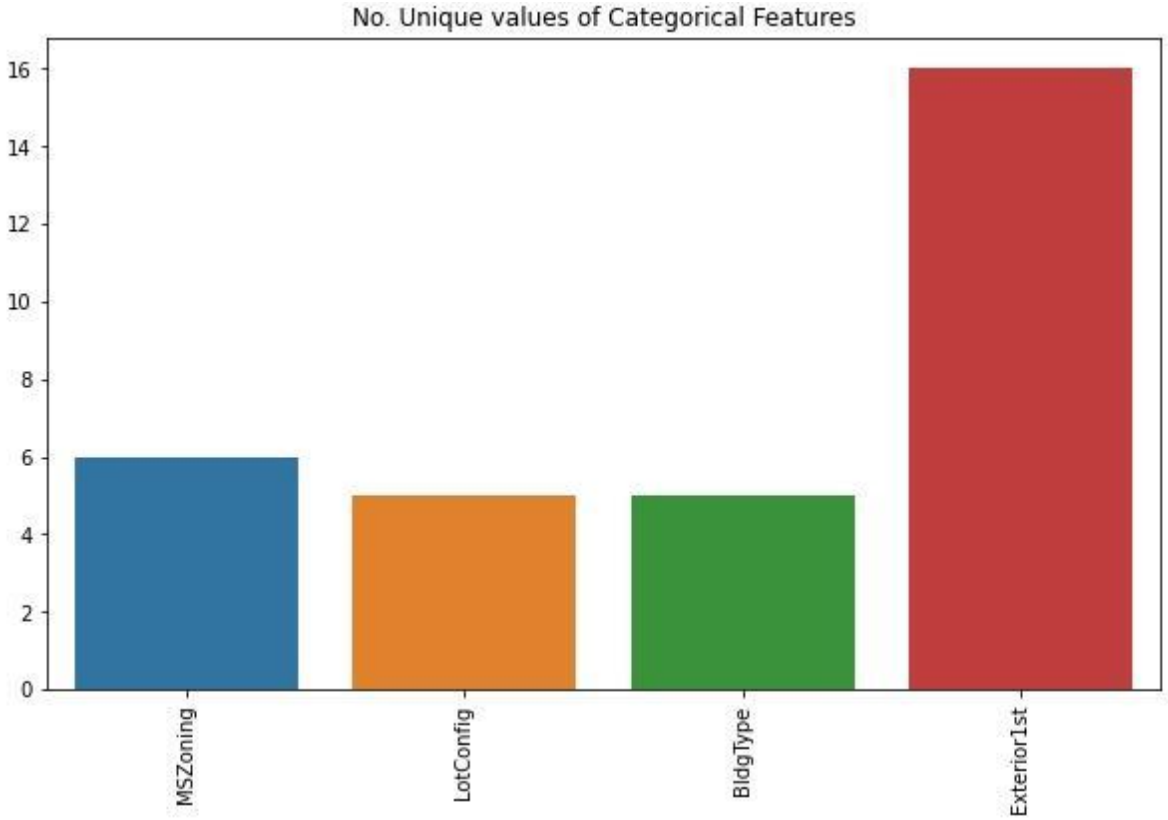



```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import r2_score, mean_absolute_error,mean_squared_error
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import Lasso
```

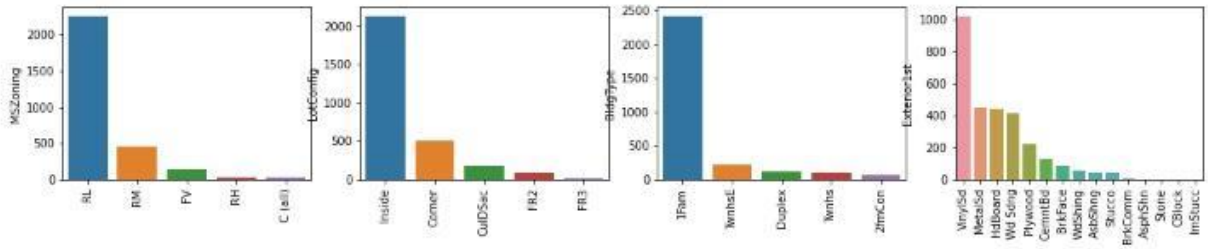
```
from sklearn.ensemble import RandomForestRegressor
from sklearn.svm import SVR
import xgboost as xg
Warnings.filterwarnings("ignore")
Sns.histplot(dataset, x='Price', bins=50, color='y')
Out:
<Axes:xlabel='Price', ylabel='Count'>
Plt.figure(figsize=(12, 6))
Sns.heatmap(dataset.corr(), Cmap = 'BrBG' Fmt = '.2f' Linewidths = 2, Annot = True)
```



```
unique_values = []
for col in object_cols:
unique_values.append(dataset[col].unique().size)
plt.figure(figsize=(10,6))
plt.title('No. Unique values of Categorical Features')
```



```
Plt.figure(figsize=(18, 36))
Plt.title('Categorical Features: Distribution')
Plt.xticks(rotation=90)
Index = 1

For col in object_cols:
   Y = dataset[col].value_counts()
   Plt.subplot(11, 4, index)
   Plt.xticks(rotation=90)
   Sns.barplot(x=list(y.index), y=y)
   Index += 1
```




```
Import pandas as pd
From sklearn.preprocessing import LabelEncoder
From sklearn.model_selection import train_test_split
From sklearn.impute import SimpleImputer
From sklearn.preprocessing import StandardScaler
Data = pd.read_csv('E:/USA_Housing.csv')
Print("Dataset Preview:")
Print(data.head())
Numeric_cols = data.select_dtypes(include='number').columns
Categorical_cols = data.select_dtypes(exclude='number').columns
```

Imputer_numeric = SimpleImputer(strategy='mean')

Imputer_categorical = SimpleImputer(strategy='most_frequent')

Data[numeric_cols] =Imputer_numeric.fit_transform(data[numeric_cols])

Data[categorical_cols] =**Imputer_categorical.fit_transform(data[categorical_cols])**

Label_encoder = LabelEncoder()

For col in categorical_cols:

Data[col] = label_encoder.fit_transform(data[col])

 # Split Data into Features (X) and Target (y)

X = data.drop(columns=['Price']) # Features

Y = data['Price'] # Target

Scaler = StandardScaler()

X_scaled = scaler.fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(X_scaled, y,

Test_size=0.2, random_state=42)

Print("\nPreprocessed Data:")

Print(X_train[:5]) # Display first 5 rows of preprocessed features

Print(y_train[:5]) # Display first 5 rows of target values

OUTPUT

Avg. Area Income Avg. Area House Age Avg. Area Number of Roo

**ms **

0 79545.458574 5.682861 7.009188

1 79248.642455 6.002900 6.730821

2 61287.067179 5.865890 8.512727

3 63345.240046 7.188236 5.586729

4 59982.197226 5.040555 7.839388

**Avg. Area Number of Bedrooms Area Population Price **

0 4.09 23086.800503 1.059034e+06

1 3.09 40173.072174 1.505891e+06

2 5.13 36882.159400 1.058988e+06

3 3.26 34310.242831 1.260617e+06

4 4.23 26354.109472 6.309435e+05

Address

0 208 Michael Ferry Apt. 674\nLaurabury, NE 3701...

1 188 Johnson Views Suite 079\nLake Kathleen, CA...

2 9127 Elizabeth Stravenue\nDanieltown, WI 06482...

3 USS Barnett\nFPO AP 44820

4 USNS Raymond\nFPO AE 09386

Preprocessed Data:

[[-0.19105816 -0.13226994 -0.13969293 0.12047677 -0.83757985 -1.0

0562872]

[-1.39450169 0.42786736 0.79541275 -0.55212509 1.15729018 1.61

946754]

[-0.35137865 0.46394489 1.70199509 0.03133676 -0.32671213 1.63

886651]

[-0.13944143 0.1104872 0.22289331 -0.75471601 -0.90401197 -1.54

810704]

[0.62516685 2.20969666 0.42984356 -0.45488144 0.12566216 0.98

830821]]

4227 1.094880e+06

4676 1.300389e+06

800 1.382172e+06

3671 1.027428e+06

4193 1.562887e+06

Name: Price, dtype: float64