```
data["price"].max()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-1-8fe122fbccf5> in <module>()
----> 1 data["price"].max()

NameError: name 'data' is not defined
```

SEARCH STACK OVERFLOW

```
import pandas as pd
data = pd.read_csv("http://hackveda.in/sistec/Housing_Modified.csv")
print(data)
```

```
        price  lotsize  bedrooms  bathrms  ... gashw airco garagepl prefarea
0     42000.0     5850         3        1  ...    no    no        1       no
1     38500.0     4000         2        1  ...    no    no        0       no
2     49500.0     3060         3        1  ...    no    no        0       no
3     60500.0     6650         3        1  ...    no    no        0       no
4     61000.0     6360         2        1  ...    no    no        0       no
..        ...      ...       ...      ...  ...   ...   ...      ...      ...
541   91500.0     4800         3        2  ...    no   yes        0       no
542   94000.0     6000         3        2  ...    no   yes        0       no
543  103000.0     6000         3        2  ...    no   yes        1       no
544  105000.0     6000         3        2  ...    no   yes        1       no
545  105000.0     6000         3        1  ...    no   yes        1       no

[546 rows x 12 columns]
```

```
data["price"].max()
data["price"].min()
data["price"].mean()
data["price"].sum()
```

```
37194392.0
```

```
data["bedrooms"].unique()
data["airco"].unique()
```

```
array(['no', 'yes'], dtype=object)
```

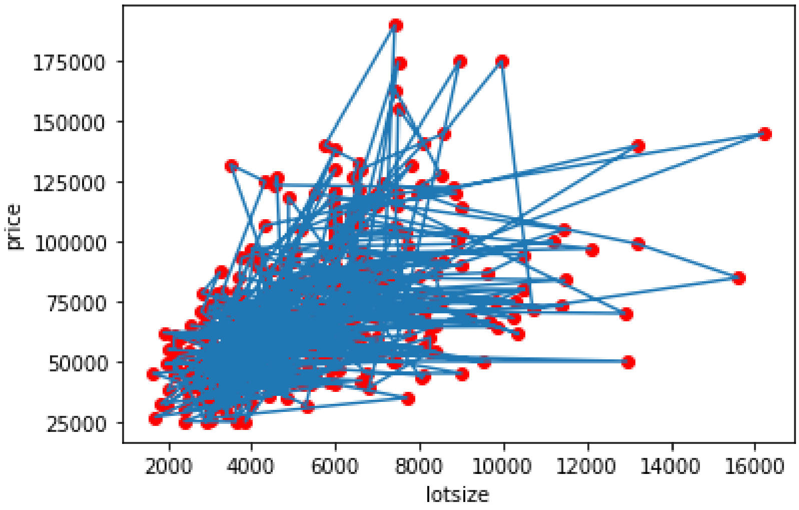
```
data.line("price","lotsize")
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-13-80e5c431fdb5> in <module>()
----> 1 data.line("price","lotsize")

/usr/local/lib/python3.6/dist-packages/pandas/core/generic.py in __getattr__(self, na
   5272             if self._info_axis._can_hold_identifiers_and_holds_name(name):
   5273                 return self[name]
-> 5274             return object.__getattribute__(self, name)
   5275
```

```
import matplotlib.pyplot as plt
plt.scatter(data["lotsize"],data["price"],color="red")
plt.plot(data["lotsize"],data["price"])
plt.xlabel("lotsize")
plt.ylabel("price")
```

```
Text(0, 0.5, 'price')
```



```
data.head(3)
```

	price	lotsize	bedrooms	bathrms	stories	driveway	recroom	fullbase	gashw
0	42000.0	5850	3	1	two	yes	no	yes	no
1	38500.0	4000	2	1	one	yes	no	no	no
2	49500.0	3060	3	1	one	yes	no	no	no

```
#convert text to numbers using Label Binarizer
import sklearn.preprocessing as pp
lb = pp.LabelBinarizer()
data.driveway = lb.fit_transform(data.driveway)
data.recroom = lb.fit_transform(data.recroom)
data.head(15)
```

	price	lotsize	bedrooms	bathrms	stories	driveway	recroom	fullbase	gashw
0	42000.0	5850	3	1	two	1	0	yes	no
1	38500.0	4000	2	1	one	1	0	no	no
2	49500.0	3060	3	1	one	1	0	no	no
3	60500.0	6650	3	1	two	1	1	no	no
4	61000.0	6360	2	1	one	1	0	no	no
5	66000.0	4160	3	1	one	1	1	yes	no
6	66000.0	3880	3	2	two	1	0	yes	no
7	69000.0	4160	3	1	three	1	0	no	no
8	83800.0	4800	3	1	one	1	1	yes	no
9	88500.0	5500	3	2	four	1	1	no	no
10	90000.0	7200	3	2	one	1	0	yes	no

```
data.corr()
```

	price	lotsize	bedrooms	bathrms	driveway	recroom	garagepl
price	1.000000	0.535796	0.366447	0.516719	0.297167	0.254960	0.383302
lotsize	0.535796	1.000000	0.151851	0.193833	0.288778	0.140327	0.352872
bedrooms	0.366447	0.151851	1.000000	0.373769	-0.011996	0.080492	0.139117
bathrms	0.516719	0.193833	0.373769	1.000000	0.041955	0.126892	0.178178
driveway	0.297167	0.288778	-0.011996	0.041955	1.000000	0.091959	0.203682
recroom	0.254960	0.140327	0.080492	0.126892	0.091959	1.000000	0.038122
garagepl	0.383302	0.352872	0.139117	0.178178	0.203682	0.038122	1.000000

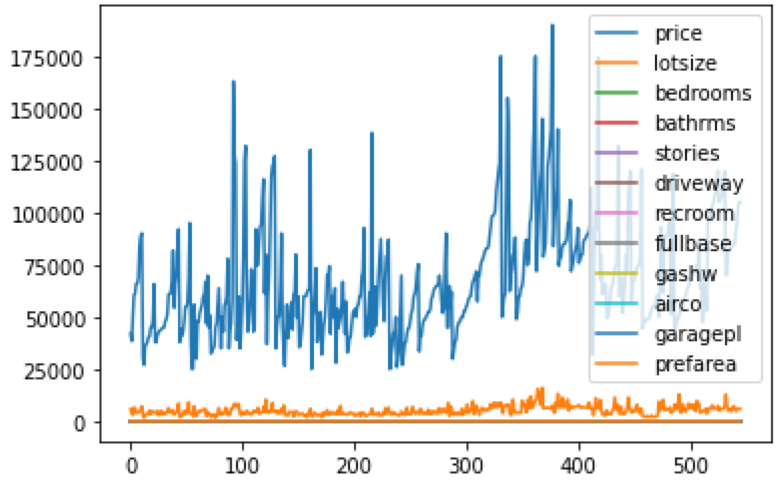
```
data = pd.read_csv("Housing_Modified_prepared.csv")
data.corr()
```

	price	lotsize	bedrooms	bathrms	stories	driveway	recroom	full
price	1.000000	0.535796	0.366447	0.516719	0.421190	0.297167	0.254960	0.18
lotsize	0.535796	1.000000	0.151851	0.193833	0.083675	0.288778	0.140327	0.04
bedrooms	0.366447	0.151851	1.000000	0.373769	0.407974	0.011996	0.080492	0.09

```
?pp.LabelEncoder()
```

```
import pandas as pd
data = pd.read_csv("Housing_Modified_prepared.csv")
data.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f67c8866da0>
```



```
#Data transformation using normalization
#Implement standardization
# Formula : (X -Xmean)/Xstd
# X = Independent variable
# Xmean = Mean of X variable
# Xstd = Standard Deviation of X independent variable
print("Columns in data : ", data.columns)
print("Max Value of lotsize before transformation :", data["lotsize"].max())
print("Min Value of lotsize before transformation :", data["lotsize"].min())
print("Mean Value of lotsize before transformation :", data["lotsize"].mean())

#Perform transformation using standard score/Standardization
X = data["lotsize"] #Select the independent variable
Xmean = X.mean() # calculate the mean of independent variable
Xstd = X.std() # calculate the standard deviation of independent variable
Xnorm = (X - Xmean) / Xstd
print("X normalized after transformation\n")
Xnorm

print("Min value of Xnorm", Xnorm.min())
print("Max value of Xnorm", Xnorm.max())
```

```
Columns in data :  Index(['price', 'lotsize', 'bedrooms', 'bathrms', 'stories', 'dri
       'recroom', 'fullbase', 'gashw', 'airco', 'garagepl', 'prefarea'],
      dtype='object')
Max Value of lotsize before transformation : 16200
```

```
Min Value of lotsize before transformation : 1650
Mean Value of lotsize before transformation : 5150.2655677655675
X normalized after transformation

Min value of Xnorm -1.6143954439482222
Max value of Xnorm 5.096367855203785
```

```
independent_variables = data.columns
independent_variables = independent_variables.delete(0)
print("Independent Variables : ", independent_variables)
```

```
Independent Variables :  Index(['lotsize', 'bedrooms', 'bathrms', 'stories', 'drivewa
       'fullbase', 'gashw', 'airco', 'garagepl', 'prefarea'],
      dtype='object')
```

```
data[independent_variables]
```

	lotsize	bedrooms	bathrms	stories	driveway	recroom	fullbase	gashw	airco
0	5850	3	1	2	1	0	1	0	0
1	4000	2	1	1	1	0	0	0	0
2	3060	3	1	1	1	0	0	0	0
3	6650	3	1	2	1	1	0	0	0
4	6360	2	1	1	1	0	0	0	0
...	...	...	...	...	...	...	...	...	...
541	4800	3	2	4	1	1	0	0	1
542	6000	3	2	4	1	0	0	0	1
543	6000	3	2	4	1	1	0	0	1
544	6000	3	2	2	1	1	0	0	1
545	6000	3	1	2	1	0	0	0	1

546 rows × 11 columns

```
X = data[independent_variables] # Independent variables
Y = data["price"] # Dependent variable
X
Y

# perform standardization on independent variables
from sklearn.preprocessing import StandardScaler
scale = StandardScaler()
Xnorm = scale.fit_transform(X)
print("Normalized data using standardization")
Xnorm
```

```
Normalized data using standardization
array([[ 0.32302806,  0.0472349 , -0.5694948 , ..., -0.68103375,
         0.35756661, -0.55337157],
       [-0.53101296, -1.31014696, -0.5694948 , ..., -0.68103375,
        -0.80452487, -0.55337157],
       [-0.96495812,  0.0472349 , -0.5694948 , ..., -0.68103375,
        -0.80452487, -0.55337157],
       ...,
       [ 0.39227462,  0.0472349 ,  1.42373699, ...,  1.46835601,
         0.35756661, -0.55337157],
       [ 0.39227462,  0.0472349 ,  1.42373699, ...,  1.46835601,
         0.35756661, -0.55337157],
       [ 0.39227462,  0.0472349 , -0.5694948 , ...,  1.46835601,
         0.35756661, -0.55337157]])
```

```
# feature scaling to normalize data on a common scale (0-1)
# feature scaling is also known as Min-Max Scaling
# formula for min-max scaling is : Xnorm = (X - Xmin)/(Xmax - Xmin)
print("Max Value of lotsize  :", data["lotsize"].max())
print("Min Value of lotsize  :", data["lotsize"].min())
data["lotsize"].head(2)
```

```
Max Value of lotsize  : 16200
Min Value of lotsize  : 1650
0    5850
1    4000
Name: lotsize, dtype: int64
```

```
X = data["lotsize"] # select the variable
Xmin = X.min() #calculate the min value of data variable
Xmax = X.max() #calculate the max value of data variable
Xnorm = (X - Xmin)/(Xmax - Xmin)
print("Normalized values after feature scaling\n")
Xnorm

print("Min value of Xnorm", Xnorm.min())
print("Max value of Xnorm", Xnorm.max())
```

```
Normalized values after feature scaling

Min value of Xnorm 0.0
Max value of Xnorm 1.0
```

```
X = data[independent_variables]
# feature scaling using Min-Max Scaling
Xmin = X.min() #calculate the min value of data variable
Xmax = X.max() #calculate the max value of data variable
Xnorm = (X - Xmin)/(Xmax - Xmin)
print("Feature scaled independent variables after transformations\n")
Xnorm
```

```
Feature scaled independent variables after transformations
```

	lotsize	bedrooms	bathrms	stories	driveway	recroom	fullbase	gashw	airco
0	0.288660	0.4	0.000000	0.333333	1.0	0.0	1.0	0.0	0.(
1	0.161512	0.2	0.000000	0.000000	1.0	0.0	0.0	0.0	0.(
2	0.096907	0.4	0.000000	0.000000	1.0	0.0	0.0	0.0	0.(
3	0.343643	0.4	0.000000	0.333333	1.0	1.0	0.0	0.0	0.(
4	0.323711	0.2	0.000000	0.000000	1.0	0.0	0.0	0.0	0.(
...	...	...	...	...	...	...	...	...	..
541	0.216495	0.4	0.333333	1.000000	1.0	1.0	0.0	0.0	1.(
542	0.298969	0.4	0.333333	1.000000	1.0	0.0	0.0	0.0	1.(
543	0.298969	0.4	0.333333	1.000000	1.0	1.0	0.0	0.0	1.(
544	0.298969	0.4	0.333333	0.333333	1.0	1.0	0.0	0.0	1.(
545	0.298969	0.4	0.000000	0.333333	1.0	0.0	0.0	0.0	1.(

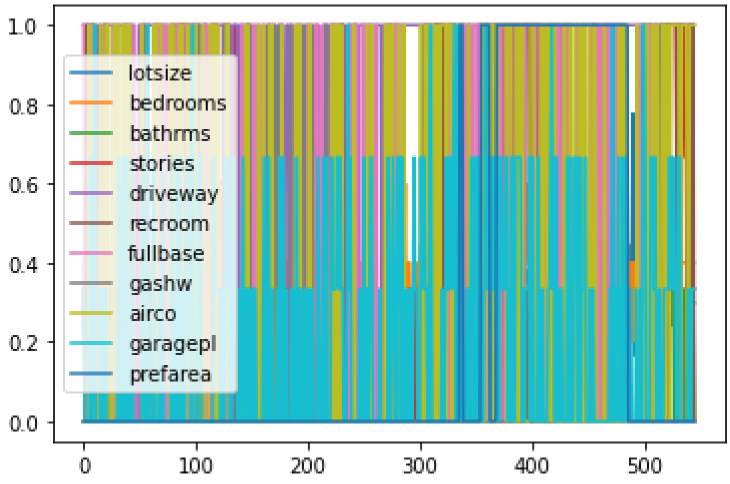
546 rows x 11 columns

```
X = data[independent_variables]
# feature scaling using Min-Max Scaling
Xmin = X.min() #calculate the min value of data variable
Xmax = X.max() #calculate the max value of data variable
Xnorm = (X - Xmin)/(Xmax - Xmin)
print("Feature scaled independent variables after transformations\n")

# Make a plot of normalized values
Xnorm.plot()
```

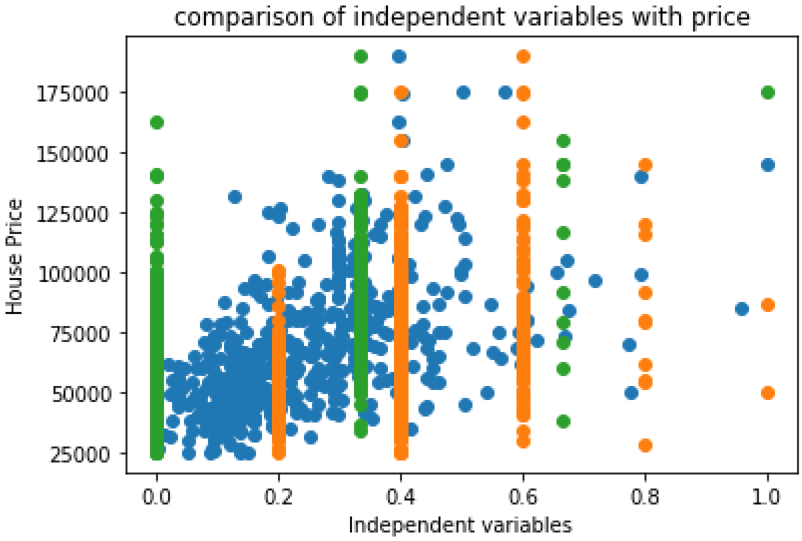
```
Feature scaled independent variables after transformations

<matplotlib.axes._subplots.AxesSubplot at 0x7fd2b0c00a20>
```



```
# Compare independent variables  and their influence on price
import matplotlib.pyplot as plt
plt.scatter(Xnorm["lotsize"], data["price"])
plt.scatter(Xnorm["bedrooms"], data["price"])
plt.scatter(Xnorm["bathrms"], data["price"])
plt.scatter(Xnorm["bathrms"], data["price"])
plt.title("comparison of independent variables with price")
plt.xlabel("Independent variables")
plt.ylabel("House Price")
```

```
     Text(0, 0.5, 'House Price')
```



```
# perform feature scaling using min-max scaler
from sklearn.preprocessing import MinMaxScaler
scale = MinMaxScaler()
Xnorm = scale.fit_transform(X)
print("transformation using min-max scaler\n")
Xnorm
```

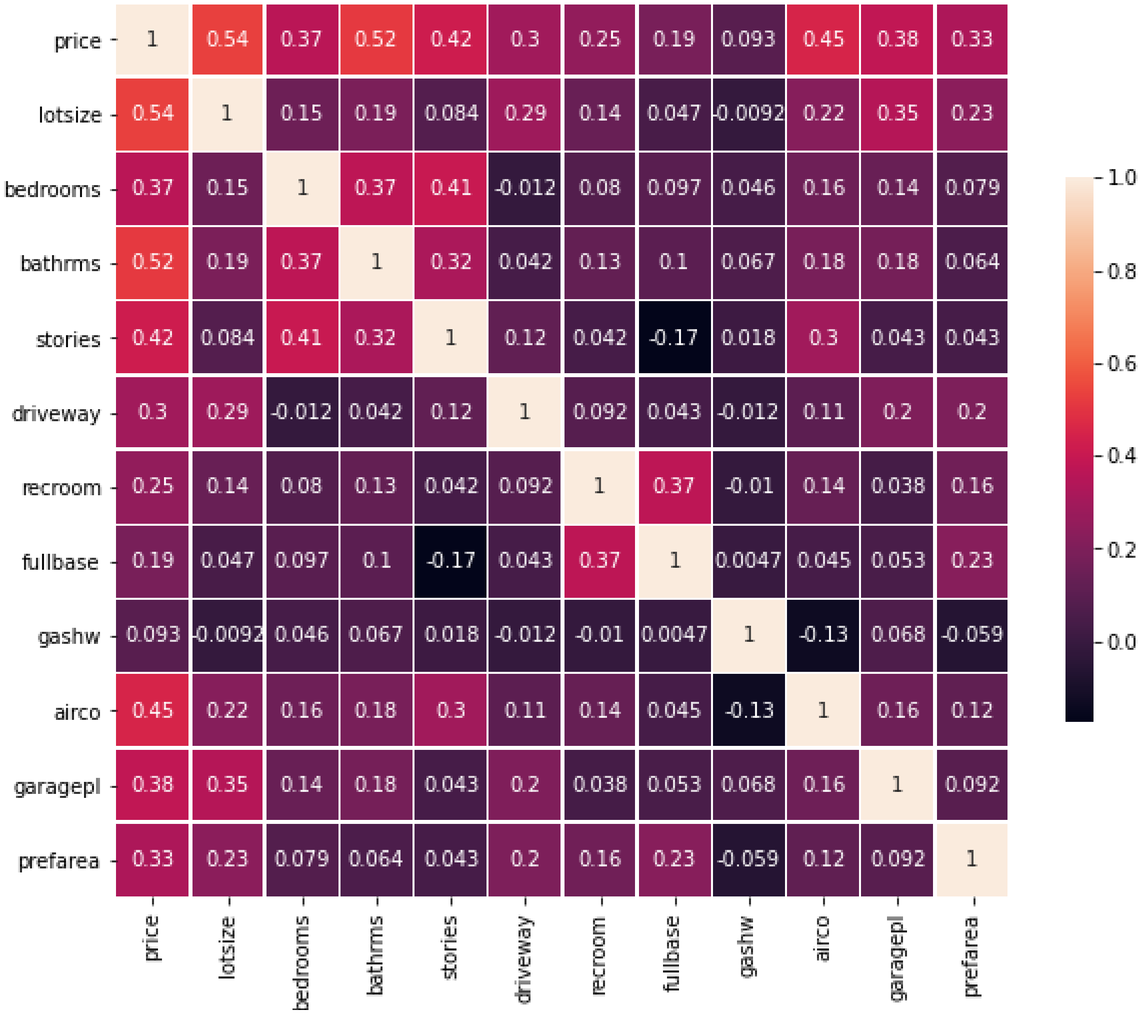
```
     transformation using min-max scaler

     array([[0.28865979, 0.4       , 0.        , ..., 0.        , 0.33333333,
             0.        ],
            [0.16151203, 0.2       , 0.        , ..., 0.        , 0.        ,
             0.        ],
            [0.09690722, 0.4       , 0.        , ..., 0.        , 0.        ,
             0.        ],
            ...,
            [0.29896907, 0.4       , 0.33333333, ..., 1.        , 0.33333333,
             0.        ],
            [0.29896907, 0.4       , 0.33333333, ..., 1.        , 0.33333333,
             0.        ],
            [0.29896907, 0.4       , 0.        , ..., 1.        , 0.33333333,
             0.        ]])
```

```
# Create  a correlation matrix using Seaborn and Matplotlib
import seaborn as sns

# Set up the matplotlib figure
size = max(10, len(data.columns)/2.)
f, ax = plt.subplots(figsize=(size, size))

# Draw the heatmap with the mask and correct aspect ratio
sns.heatmap(data.corr(), annot=True, square=True, linewidths=.5, cbar_kws={"shrink": 0.5},
<matplotlib.axes._subplots.AxesSubplot at 0x7fd2af7dce48>
```



	price	lotsize	bedrooms	bathrms	stories	driveway	recroom	fullbase	gashw	airco	garagepl	prefarea
price	1	0.54	0.37	0.52	0.42	0.3	0.25	0.19	0.093	0.45	0.38	0.33
lotsize	0.54	1	0.15	0.19	0.084	0.29	0.14	0.047	-0.0092	0.22	0.35	0.23
bedrooms	0.37	0.15	1	0.37	0.41	-0.012	0.08	0.097	0.046	0.16	0.14	0.079
bathrms	0.52	0.19	0.37	1	0.32	0.042	0.13	0.1	0.067	0.18	0.18	0.064
stories	0.42	0.084	0.41	0.32	1	0.12	0.042	-0.17	0.018	0.3	0.043	0.043
driveway	0.3	0.29	-0.012	0.042	0.12	1	0.092	0.043	-0.012	0.11	0.2	0.2
recroom	0.25	0.14	0.08	0.13	0.042	0.092	1	0.37	-0.01	0.14	0.038	0.16
fullbase	0.19	0.047	0.097	0.1	-0.17	0.043	0.37	1	0.0047	0.045	0.053	0.23
gashw	0.093	-0.0092	0.046	0.067	0.018	-0.012	-0.01	0.0047	1	-0.13	0.068	-0.059
airco	0.45	0.22	0.16	0.18	0.3	0.11	0.14	0.045	-0.13	1	0.16	0.12
garagepl	0.38	0.35	0.14	0.18	0.043	0.2	0.038	0.053	0.068	0.16	1	0.092
prefarea	0.33	0.23	0.079	0.064	0.043	0.2	0.16	0.23	-0.059	0.12	0.092	1