In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib
import xgboost as xgb
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
from scipy.stats import skew
from scipy.stats.stats import pearsonr

%config InlineBackend.figure_format = 'png' #set 'png' here when working
%matplotlib inline
from sklearn.model_selection import KFold, cross_val_score
```

In [38]:
```
train = pd.read_csv(r'train.csv')
test = pd.read_csv(r'test.csv')
```

In [39]:
```
train
```

Out[39]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	Lan
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	
...	...	...	...	...	...	...	...	...	
1455	1456	60	RL	62.0	7917	Pave	NaN	Reg	
1456	1457	20	RL	85.0	13175	Pave	NaN	Reg	
1457	1458	70	RL	66.0	9042	Pave	NaN	Reg	
1458	1459	20	RL	68.0	9717	Pave	NaN	Reg	
1459	1460	20	RL	75.0	9937	Pave	NaN	Reg	

1460 rows × 81 columns

In [40]:
```
test
```

Out[40]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	Lan
0	1461	20	RH	80.0	11622	Pave	NaN	Reg	
1	1462	20	RL	81.0	14267	Pave	NaN	IR1	
2	1463	60	RL	74.0	13830	Pave	NaN	IR1	
3	1464	60	RL	78.0	9978	Pave	NaN	IR1	
4	1465	120	RL	43.0	5005	Pave	NaN	IR1	
...	...	...	...	...	...	...	...	...	
1454	2915	160	RM	21.0	1936	Pave	NaN	Reg	
1455	2916	160	RM	21.0	1894	Pave	NaN	Reg	
1456	2917	20	RL	160.0	20000	Pave	NaN	Reg	
1457	2918	85	RL	62.0	10441	Pave	NaN	Reg	
1458	2919	60	RL	74.0	9627	Pave	NaN	Reg	

1459 rows × 80 columns

In [41]:
```
all_data = pd.concat((train.loc[:,'MSSubClass':'SaleCondition'],test.loc[
```

In [42]:
```
all_data
```

Out[42]:

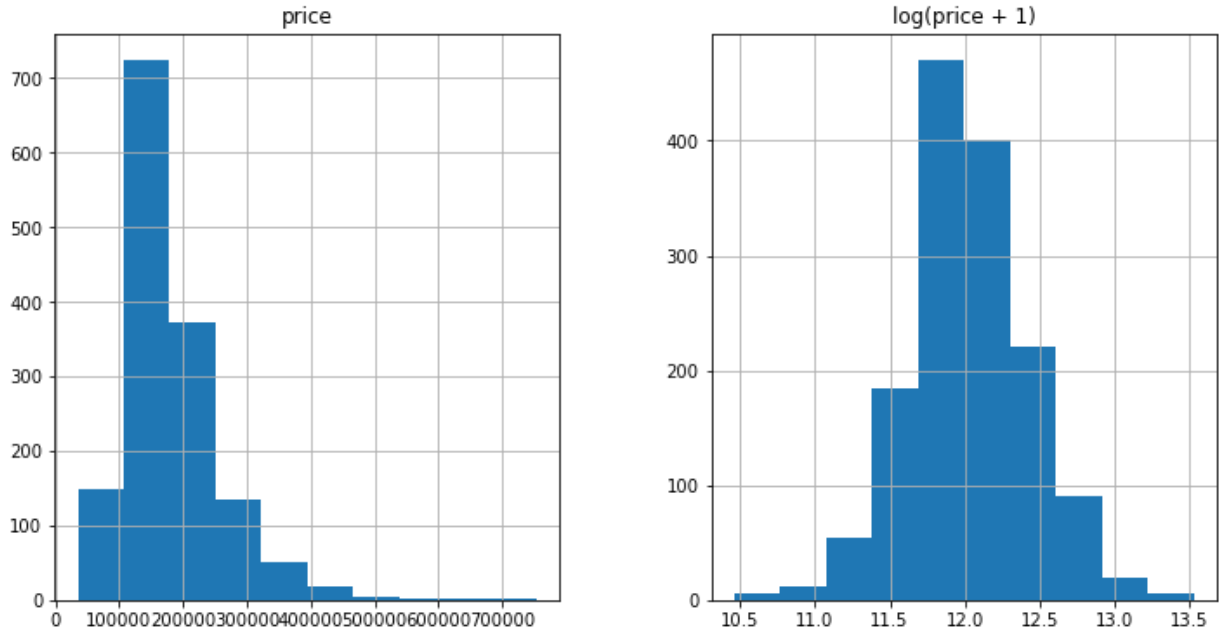
	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContc
0	60	RL	65.0	8450	Pave	NaN	Reg	
1	20	RL	80.0	9600	Pave	NaN	Reg	
2	60	RL	68.0	11250	Pave	NaN	IR1	
3	70	RL	60.0	9550	Pave	NaN	IR1	
4	60	RL	84.0	14260	Pave	NaN	IR1	
...	...	...	...	...	...	...	...	
1454	160	RM	21.0	1936	Pave	NaN	Reg	
1455	160	RM	21.0	1894	Pave	NaN	Reg	
1456	20	RL	160.0	20000	Pave	NaN	Reg	
1457	85	RL	62.0	10441	Pave	NaN	Reg	
1458	60	RL	74.0	9627	Pave	NaN	Reg	

2919 rows × 79 columns

In [43]:
```
# model regression . fit
# model lasso . fit
```

In [44]:
```
#features more normal
matplotlib.rcParams['figure.figsize'] = (12.0, 6.0)
prices = pd.DataFrame({"price":train["SalePrice"], "log(price + 1)":np.lo
prices.hist()
```

Out[44]:
```
array([[<AxesSubplot:title={'center':'price'}>,
        <AxesSubplot:title={'center':'log(price + 1)'}>]], dtype=object)
```



In [45]:
```
train["SalePrice"] = np.log1p(train["SalePrice"])
```

In [46]:
```
train["SalePrice"] #define target
```

Out[46]:
```
0       12.247699
1       12.109016
2       12.317171
3       11.849405
4       12.429220
          ...
1455    12.072547
1456    12.254868
1457    12.493133
1458    11.864469
1459    11.901590
Name: SalePrice, Length: 1460, dtype: float64
```

In [47]:
```
numeric_feats = all_data.dtypes[all_data.dtypes != "object"].index
```

In [48]:
```
numeric_feats
```

Out[48]:
```
Index(['MSSubClass', 'LotFrontage', 'LotArea', 'OverallQual', 'OverallCon
d',
       'YearBuilt', 'YearRemodAdd', 'MasVnrArea', 'BsmtFinSF1', 'BsmtFinS
F2',
       'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF'
,
       'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath', 'HalfBath
',
       'BedroomAbvGr', 'KitchenAbvGr', 'TotRmsAbvGrd', 'Fireplaces',
       'GarageYrBlt', 'GarageCars', 'GarageArea', 'WoodDeckSF', 'OpenPorc
hSF',
       'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'MiscVal'
,
       'MoSold', 'YrSold'],
      dtype='object')
```

In [49]:
```
skewed_feats = train[numeric_feats].apply(lambda x: skew(x.dropna())) #co
skewed_feats = skewed_feats[skewed_feats > 0.75]
skewed_feats = skewed_feats.index

all_data[skewed_feats] = np.log1p(all_data[skewed_feats])
```

In [50]:
```
all_data[skewed_feats]
```

Out[50]:

	MSSubClass	LotFrontage	LotArea	MasVnrArea	BsmtFinSF1	BsmtFinSF2	BsmtU
0	4.110874	4.189655	9.042040	5.283204	6.561031	0.0	5.01
1	3.044522	4.394449	9.169623	0.000000	6.886532	0.0	5.65
2	4.110874	4.234107	9.328212	5.093750	6.188264	0.0	6.07
3	4.262680	4.110874	9.164401	0.000000	5.379897	0.0	6.29
4	4.110874	4.442651	9.565284	5.860786	6.486161	0.0	6.19
...	...	...	...	...	...	...	
1454	5.081404	3.091042	7.568896	0.000000	0.000000	0.0	6.30
1455	5.081404	3.091042	7.546974	0.000000	5.533389	0.0	5.68
1456	3.044522	5.081404	9.903538	0.000000	7.110696	0.0	0.00
1457	4.454347	4.143135	9.253591	0.000000	5.823046	0.0	6.35
1458	4.110874	4.317488	9.172431	4.553877	6.632002	0.0	5.47

2919 rows × 21 columns

In [51]:
```
all_data = pd.get_dummies(all_data)
```

In [52]:
```
#filling NA's with the mean of the column:
all_data = all_data.fillna(all_data.mean())
```

In [53]:
```
#creating matrices for sklearn:
X_train = all_data[:train.shape[0]]
X_test = all_data[train.shape[0]:]
y = train.SalePrice
```

In [54]:
```
all_data
```

Out[54]:

	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRem
0	4.110874	4.189655	9.042040	7	5	2003	
1	3.044522	4.394449	9.169623	6	8	1976	
2	4.110874	4.234107	9.328212	7	5	2001	
3	4.262680	4.110874	9.164401	7	5	1915	
4	4.110874	4.442651	9.565284	8	5	2000	
...	...	...	...	...	...	...	
1454	5.081404	3.091042	7.568896	4	7	1970	
1455	5.081404	3.091042	7.546974	4	5	1970	
1456	3.044522	5.081404	9.903538	5	7	1960	
1457	4.454347	4.143135	9.253591	5	5	1992	
1458	4.110874	4.317488	9.172431	7	5	1993	

2919 rows × 288 columns

In [55]:
```
from sklearn.linear_model import Ridge, RidgeCV, ElasticNet, LassoCV, Las
from sklearn.model_selection import cross_val_score

def rmse_cv(model):
    rmse= np.sqrt(-cross_val_score(model, X_train, y, scoring="neg_mean_s
    return(rmse)
```

In [56]:
```
model_ridge = Ridge()
```

In [57]:
```
model_ridge
```

Out[57]:
```
Ridge()
```

In [58]:
```
alphas=[0.1]
cv_ridge = [rmse_cv(Ridge(alpha =alpha)).mean()
            for alpha in alphas]
```

RMSE Value - Ridge

In [59]:
```
cv_ridge
```

Out[59]: [0.13777538277187865]

RMSE Value - Lasso

In [60]:
```
model_lasso = LassoCV(alphas = [0.1]).fit(X_train, y)
```

In [61]:
```
rmse_cv(model_lasso).mean()
```

Out[61]: 0.20921930047608214

Obseerved that ridge performed better for alpha

Optimizing Alpha

In [62]:
```
# For Ridge
alphas=np.arange(5, 15, 1).tolist() #running on alpha from 5 to 15 at a 1
cv_ridge = [rmse_cv(Ridge(alpha =alpha)).mean()
            for alpha in alphas]
```

In [63]:
```
min_index=cv_ridge.index(min(cv_ridge)) #min cv see for index
cv_ridge[min_index] #this is minimum cv score we could achieve
```

Out[63]: 0.12733734668670776

In [64]:
```
alphas[min_index] #min CV achieved at alpha = 10
```

Out[64]: 10

In [65]:
```
# For Lasso
alphas=np.arange(0, 0.001, 0.0001).tolist() #running on alpha from 0 to 0
lasso_cv=[]
alpha_curr=[]
for i in alphas:
    model_lasso_cv = rmse_cv(LassoCV(alphas=[i]).fit(X_train, y)).mean()
    alpha_curr.append(i)
    lasso_cv.append(model_lasso_cv)
```

```
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 4.859170575181508, tolerance: 0.018912592760396085
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
```

```
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 4.257882622560601, tolerance: 0.01800219138548883
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 4.095357955603021, tolerance: 0.018373605848561597
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 4.829918807482244, tolerance: 0.019008081403702633
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 4.703140526782022, tolerance: 0.01881061188370518
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:1714: UserWarning: With alpha=0, this algo
rithm does not converge well. You are advised to use the LinearRegression
estimator
  model.fit(X, y)
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:647: UserWarning: Coordinate descent with
no regularization may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:647: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations, check the
scale of the features or consider increasing regularisation. Duality gap:
6.153e+00, tolerance: 2.328e-02 Linear regression models with null weight
for the l1 regularization term are more efficiently fitted using one of t
he solvers implemented in sklearn.linear_model.Ridge/RidgeCV instead.
  model = cd_fast.enet_coordinate_descent(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
```

```
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 3.4079296266881443, tolerance: 0.014608206095799353
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 3.3428077749517797, tolerance: 0.014601004078052359
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 3.4383502763372746, tolerance: 0.015531757542978097
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 3.827884824674669, tolerance: 0.015448874096282393
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 3.662900988799123, tolerance: 0.015453647710308875
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:1714: UserWarning: With alpha=0, this algo
rithm does not converge well. You are advised to use the LinearRegression
estimator
  model.fit(X, y)
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:647: UserWarning: Coordinate descent with
no regularization may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:647: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations, check the
```

```
scale of the features or consider increasing regularisation. Duality gap:
4.859e+00, tolerance: 1.891e-02 Linear regression models with null weight
for the l1 regularization term are more efficiently fitted using one of t
he solvers implemented in sklearn.linear_model.Ridge/RidgeCV instead.
  model = cd_fast.enet_coordinate_descent(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 3.2535264274607414, tolerance: 0.014421032323759136
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 2.9705961284435602, tolerance: 0.013878761882709149
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 3.086940408869907, tolerance: 0.014621128652612829
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 3.3486242125725028, tolerance: 0.01453514652569633
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 3.0780428069976438, tolerance: 0.014544304735976838
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:1714: UserWarning: With alpha=0, this algo
rithm does not converge well. You are advised to use the LinearRegression
estimator
```

```
  model.fit(X, y)
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:647: UserWarning: Coordinate descent with
no regularization may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:647: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations, check the
scale of the features or consider increasing regularisation. Duality gap:
4.258e+00, tolerance: 1.800e-02 Linear regression models with null weight
for the l1 regularization term are more efficiently fitted using one of t
he solvers implemented in sklearn.linear_model.Ridge/RidgeCV instead.
  model = cd_fast.enet_coordinate_descent(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 3.078947772285062, tolerance: 0.014791078401076713
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 3.01427064223833, tolerance: 0.014455328041034693
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 2.952829479429994, tolerance: 0.014416323383021182
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 3.0683216215797984, tolerance: 0.014911027581501767
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
```

```
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 2.977869365490676, tolerance: 0.014914232105412309
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:1714: UserWarning: With alpha=0, this algo
rithm does not converge well. You are advised to use the LinearRegression
estimator
  model.fit(X, y)
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:647: UserWarning: Coordinate descent with
no regularization may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:647: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations, check the
scale of the features or consider increasing regularisation. Duality gap:
4.095e+00, tolerance: 1.837e-02 Linear regression models with null weight
for the l1 regularization term are more efficiently fitted using one of t
he solvers implemented in sklearn.linear_model.Ridge/RidgeCV instead.
  model = cd_fast.enet_coordinate_descent(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 3.662546611421817, tolerance: 0.01542865087978492
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 3.7217488118484425, tolerance: 0.015089979013565014
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 3.1000647992446773, tolerance: 0.014491631155835854
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
```

```
converge. You might want to increase the number of iterations. Duality ga
p: 3.2917687071829107, tolerance: 0.01546918576152043
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 3.715077253629545, tolerance: 0.01555239160345648
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:1714: UserWarning: With alpha=0, this algo
rithm does not converge well. You are advised to use the LinearRegression
estimator
  model.fit(X, y)
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:647: UserWarning: Coordinate descent with
no regularization may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:647: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations, check the
scale of the features or consider increasing regularisation. Duality gap:
4.830e+00, tolerance: 1.901e-02 Linear regression models with null weight
for the l1 regularization term are more efficiently fitted using one of t
he solvers implemented in sklearn.linear_model.Ridge/RidgeCV instead.
  model = cd_fast.enet_coordinate_descent(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 3.5948797841396924, tolerance: 0.015228754219030277
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 3.507123236427276, tolerance: 0.014892726351484208
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
```

```
p: 2.8620210884815833, tolerance: 0.014286517661843702
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 3.3364463600519514, tolerance: 0.01528597962708133
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: UserWarning: Coordinate descent with
alpha=0 may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 3.5593462547941215, tolerance: 0.015536468891600251
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:1714: UserWarning: With alpha=0, this algo
rithm does not converge well. You are advised to use the LinearRegression
estimator
  model.fit(X, y)
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:647: UserWarning: Coordinate descent with
no regularization may lead to unexpected results and is discouraged.
  model = cd_fast.enet_coordinate_descent(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:647: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations, check the
scale of the features or consider increasing regularisation. Duality gap:
4.703e+00, tolerance: 1.881e-02 Linear regression models with null weight
for the l1 regularization term are more efficiently fitted using one of t
he solvers implemented in sklearn.linear_model.Ridge/RidgeCV instead.
  model = cd_fast.enet_coordinate_descent(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 0.5392151380338897, tolerance: 0.01800219138548883
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:647: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations, check the
scale of the features or consider increasing regularisation. Duality gap:
9.122e-01, tolerance: 2.328e-02
  model = cd_fast.enet_coordinate_descent(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
r_model/_coordinate_descent.py:633: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations. Duality ga
p: 0.769086169657208, tolerance: 0.015531757542978097
  model = cd_fast.enet_coordinate_descent_gram(
/Users/vipulsahni/opt/anaconda3/lib/python3.9/site-packages/sklearn/linea
```

```
r_model/_coordinate_descent.py:647: ConvergenceWarning: Objective did not
converge. You might want to increase the number of iterations, check the
scale of the features or consider increasing regularisation. Duality gap:
5.392e-01, tolerance: 1.800e-02
  model = cd_fast.enet_coordinate_descent(
```

In [66]:
```
min_index=lasso_cv.index(min(lasso_cv)) #min cv see for index
lasso_cv[min_index] #this is minimum cv score we could achieve with lasso
```

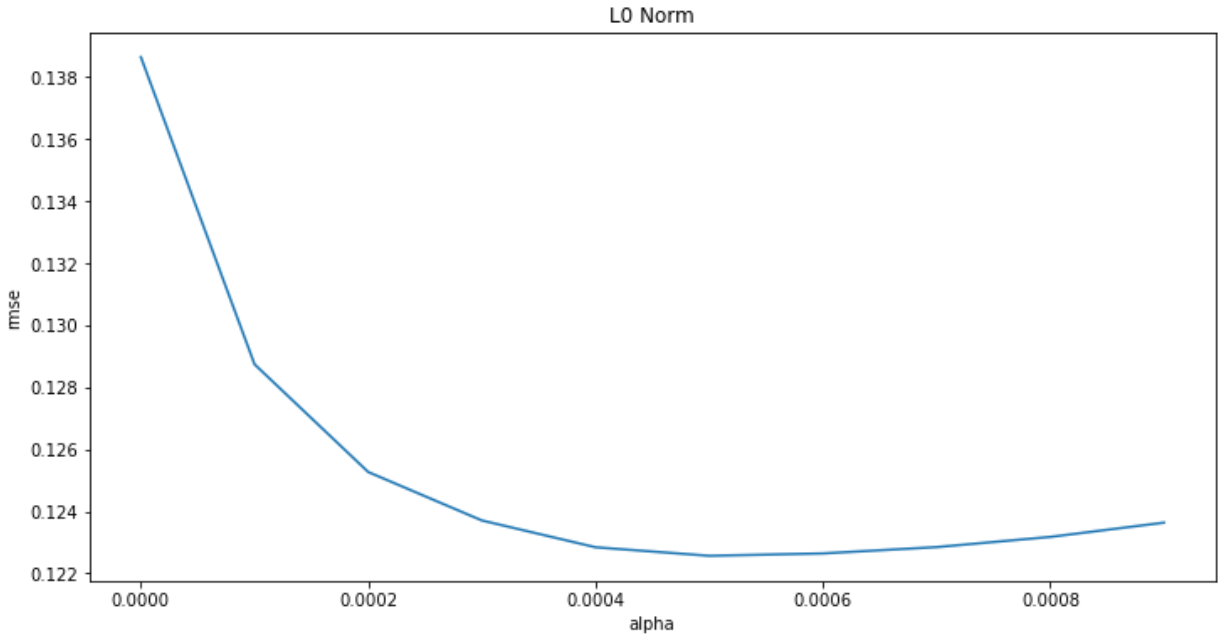
Out[66]: 0.12256735885048128

In [67]:
```
alpha_curr[min_index] #min CV achieved at alpha = 0.0005
```

Out[67]: 0.0005

In [68]:
```
#L0 Norm of the coefficients that lasso produces
lasso_cv = pd.Series(lasso_cv, index = alphas)
lasso_cv.plot(title = "L0 Norm")
plt.xlabel("alpha")
plt.ylabel("rmse")
```

Out[68]: Text(0, 0.5, 'rmse')



For Ridge, we were able to optimize alpha = 10, to get RMSE CV score of 0.127

For Lasso, alpha = 0.0005 to get RMSE CV score of 0.1225 - lower than Ridge

In [29]:
```
#Now checking predictions - Lasso
model_lasso = LassoCV(alphas = [0.0005]).fit(X_train, y)
model_lasso.predict(X_test)
```

Out[29]:
```
array([11.69490559, 11.92823243, 12.10183291, ..., 12.03779924,
       11.68640318, 12.33887195])
```

In [30]:
```
#Now checking predictions - Ridge
model_ridge = Ridge(alpha=0.1).fit(X_train, y) #0.1 because submitting to
```

In [31]:
```
pred_ridge=model_ridge.predict(X_test)
```

In [32]:
```
pred_ridge_df=pd.DataFrame(np.expm1(pred_ridge))
```

In [33]:
```
X_test['SalePrice']=pred_ridge_df[0]
X_test['Id'] = X_test.index
X_test['Id'] += 1461
```

```
/var/folders/6p/fvv3r7rj335gs5y3bm1f750m0000gn/T/ipykernel_18452/25514226
27.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-do
cs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  X_test['SalePrice']=pred_ridge_df[0]
/var/folders/6p/fvv3r7rj335gs5y3bm1f750m0000gn/T/ipykernel_18452/25514226
27.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-do
cs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  X_test['Id'] = X_test.index
/var/folders/6p/fvv3r7rj335gs5y3bm1f750m0000gn/T/ipykernel_18452/25514226
27.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-do
cs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  X_test['Id'] += 1461
```

Submission score=0.135

In [34]:
```
X_test[['Id','SalePrice']].to_csv('submission.csv') #0.135
```

Add features to ridge regression | Ensembling and Stacking

In [69]:
```
model_ridge = Ridge(alpha=10).fit(X_train, y)
model_lasso = LassoCV(alphas = [0.0005]).fit(X_train, y)
X_train['Lasso_val'] = model_lasso.predict(X_train)
X_train['Ridge_val'] = model_ridge.predict(X_train.loc[:,'MSSubClass':'Sa
```

```
/var/folders/6p/fvv3r7rj335gs5y3bm1f750m0000gn/T/ipykernel_18452/3350432349.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  X_train['Lasso_val'] = model_lasso.predict(X_train)
/var/folders/6p/fvv3r7rj335gs5y3bm1f750m0000gn/T/ipykernel_18452/3350432349.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  X_train['Ridge_val'] = model_ridge.predict(X_train.loc[:,'MSSubClass':'SaleCondition_Partial'])
```

In [70]:
```
X_train
```

Out[70]:

	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRem
0	4.110874	4.189655	9.042040	7	5	2003	
1	3.044522	4.394449	9.169623	6	8	1976	
2	4.110874	4.234107	9.328212	7	5	2001	
3	4.262680	4.110874	9.164401	7	5	1915	
4	4.110874	4.442651	9.565284	8	5	2000	
...	...	...	...	...	...	...	
1455	4.110874	4.143135	8.976894	6	5	1999	
1456	3.044522	4.454347	9.486152	6	6	1978	
1457	4.262680	4.204693	9.109746	7	9	1941	
1458	3.044522	4.234107	9.181735	5	6	1950	
1459	3.044522	4.330733	9.204121	5	6	1965	

1460 rows × 290 columns

In [165…
```
X_train.head()
```

Out[165]:

	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemod
0	4.110874	4.189655	9.042040	7	5	2003	2
1	3.044522	4.394449	9.169623	6	8	1976	1
2	4.110874	4.234107	9.328212	7	5	2001	2
3	4.262680	4.110874	9.164401	7	5	1915	1
4	4.110874	4.442651	9.565284	8	5	2000	2

5 rows × 290 columns

In [168…
```
# Check Ridge train predictions
model_ridge = Ridge(alpha=10).fit(X_train, y)
```

In [175…
```
rmse_cv(model_ridge).mean() #RMSE is improved, earlier it was 0.127
```

Out[175]: 0.12267213549556055

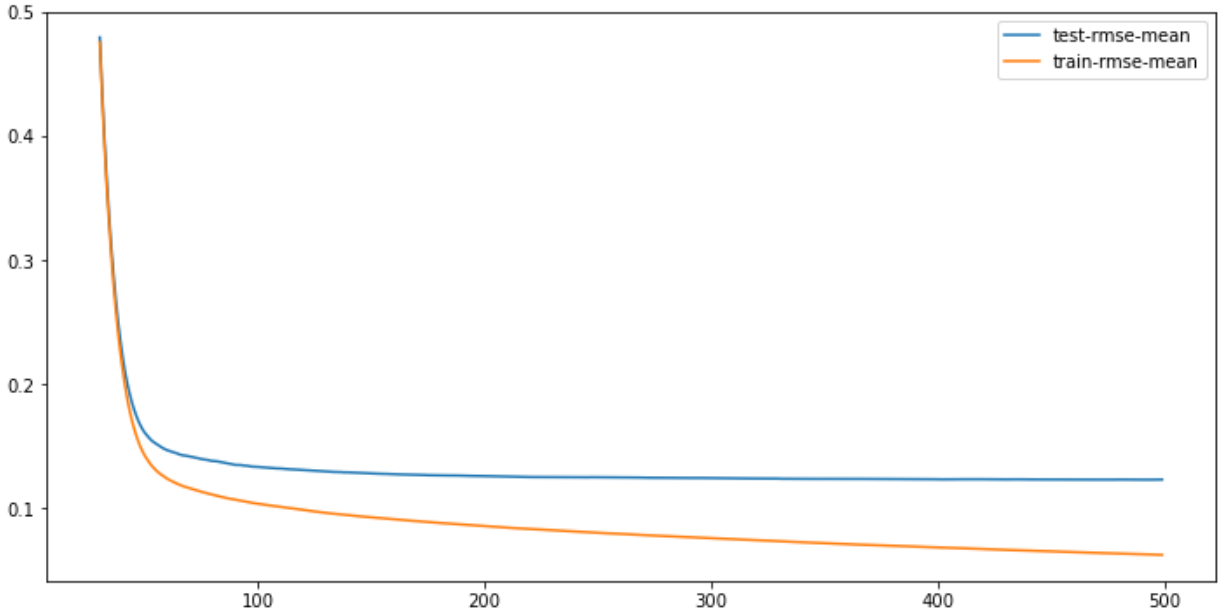
XG Boost

In [198…
```
dtrain = xgb.DMatrix(X_train.loc[:,'MSSubClass':'SaleCondition_Partial'],
dtest = xgb.DMatrix(X_test)

params = {"max_depth":2, "eta":0.1}
model = xgb.cv(params, dtrain,  num_boost_round=500, early_stopping_round
```

In [199…
```
model.loc[30:,["test-rmse-mean", "train-rmse-mean"]].plot()
```

Out[199]: <AxesSubplot:>



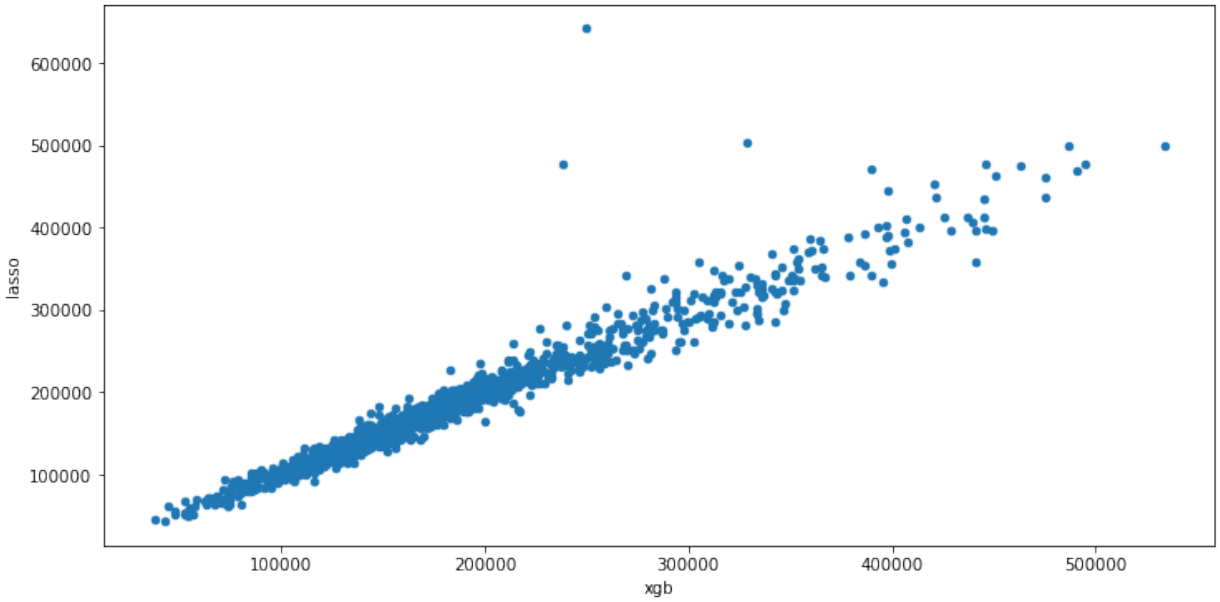
In [200…
```
model_xgb = xgb.XGBRegressor(n_estimators=360, max_depth=2, learning_rate
model_xgb.fit(X_train.loc[:,'MSSubClass':'SaleCondition_Partial'], y)
```

Out[200]:
```
XGBRegressor(base_score=0.5, booster='gbtree', callbacks=None,
             colsample_bylevel=1, colsample_bynode=1, colsample_bytree=1
,
             early_stopping_rounds=None, enable_categorical=False,
             eval_metric=None, gamma=0, gpu_id=-1, grow_policy='depthwis
e',
             importance_type=None, interaction_constraints='',
             learning_rate=0.1, max_bin=256, max_cat_to_onehot=4,
             max_delta_step=0, max_depth=2, max_leaves=0, min_child_weig
ht=1,
             missing=nan, monotone_constraints='()', n_estimators=360, n
_jobs=0,
             num_parallel_tree=1, predictor='auto', random_state=0, reg_
alpha=0,
             reg_lambda=1, ...)
```

In [204…
```
model_lasso = LassoCV(alphas = [0.0005]).fit(X_train.loc[:,'MSSubClass':'
xgb_preds = np.expm1(model_xgb.predict(X_test))
lasso_preds = np.expm1(model_lasso.predict(X_test))
```

In [205…
```
predictions = pd.DataFrame({"xgb":xgb_preds, "lasso":lasso_preds})
predictions.plot(x = "xgb", y = "lasso", kind = "scatter")
```

Out[205]: <AxesSubplot:xlabel='xgb', ylabel='lasso'>



Using only xgb_preds, model is overfitting. On training RMSE is ~0.1, but on test it is 0.13

In [212…
```
rmse_cv(model_xgb).mean() #RMSE is improved, earlier it was 0.127
```

Out[212]:
```
0.10913110265345634
```

In [206…
```
preds = 0.7*lasso_preds + 0.3*xgb_preds
```

In [214…
```
solution = pd.DataFrame({"id":test.Id, "SalePrice":preds})
# solution.to_csv("ridge_sol.csv", index = False)
```

In [215…
```
solution.to_csv('submission.csv') #Score: 0.12299
```

Improve upon this XGboost model

Feature Engineering

In [217…
```
all_data['GarageYrBltn'] = abs(all_data['YrSold'] - all_data['GarageYrBlt
all_data['YearRemodAddn'] = abs(all_data['YrSold'] - all_data['YearRemodA
all_data['YearBuiltn'] = abs(all_data['YrSold'] - all_data['YearBuilt'])
all_data['SF'] = all_data['1stFlrSF']+all_data['2ndFlrSF']+all_data['Tota
```

```
/var/folders/6p/fvv3r7rj335gs5y3bm1f750m0000gn/T/ipykernel_86807/35110614
70.py:1: PerformanceWarning: DataFrame is highly fragmented.  This is usu
ally the result of calling `frame.insert` many times, which has poor perf
ormance.  Consider joining all columns at once using pd.concat(axis=1) in
stead. To get a de-fragmented frame, use `newframe = frame.copy()`
  all_data['GarageYrBltn'] = abs(all_data['YrSold'] - all_data['GarageYrB
lt'])
/var/folders/6p/fvv3r7rj335gs5y3bm1f750m0000gn/T/ipykernel_86807/35110614
70.py:2: PerformanceWarning: DataFrame is highly fragmented.  This is usu
ally the result of calling `frame.insert` many times, which has poor perf
ormance.  Consider joining all columns at once using pd.concat(axis=1) in
stead. To get a de-fragmented frame, use `newframe = frame.copy()`
  all_data['YearRemodAddn'] = abs(all_data['YrSold'] - all_data['YearRemo
dAdd'])
/var/folders/6p/fvv3r7rj335gs5y3bm1f750m0000gn/T/ipykernel_86807/35110614
70.py:3: PerformanceWarning: DataFrame is highly fragmented.  This is usu
ally the result of calling `frame.insert` many times, which has poor perf
ormance.  Consider joining all columns at once using pd.concat(axis=1) in
stead. To get a de-fragmented frame, use `newframe = frame.copy()`
  all_data['YearBuiltn'] = abs(all_data['YrSold'] - all_data['YearBuilt']
)
/var/folders/6p/fvv3r7rj335gs5y3bm1f750m0000gn/T/ipykernel_86807/35110614
70.py:4: PerformanceWarning: DataFrame is highly fragmented.  This is usu
ally the result of calling `frame.insert` many times, which has poor perf
ormance.  Consider joining all columns at once using pd.concat(axis=1) in
stead. To get a de-fragmented frame, use `newframe = frame.copy()`
  all_data['SF'] = all_data['1stFlrSF']+all_data['2ndFlrSF']+all_data['To
talBsmtSF']+all_data['GrLivArea']+all_data['HalfBath']+all_data['FullBath
']
```

In [218…
```
all_data
```

Out[218]:

	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRer
0	4.110874	4.189655	9.042040	7	5	2003	
1	3.044522	4.394449	9.169623	6	8	1976	
2	4.110874	4.234107	9.328212	7	5	2001	
3	4.262680	4.110874	9.164401	7	5	1915	
4	4.110874	4.442651	9.565284	8	5	2000	
...	...	...	...	...	...	...	
1454	5.081404	3.091042	7.568896	4	7	1970	
1455	5.081404	3.091042	7.546974	4	5	1970	
1456	3.044522	5.081404	9.903538	5	7	1960	
1457	4.454347	4.143135	9.253591	5	5	1992	
1458	4.110874	4.317488	9.172431	7	5	1993	

2919 rows × 292 columns

In [223…
```
scaler = StandardScaler()
scaler.fit(all_data)
all_data = pd.DataFrame(scaler.transform(all_data), index = all_data.inde
```

In [224…
```
all_data
```

Out[224]:

	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearR
0	0.419418	-0.020358	-0.103719	0.646183	-0.507284	1.046258	
1	-1.120845	0.619103	0.146544	-0.063185	2.188279	0.154764	
2	0.419418	0.118440	0.457629	0.646183	-0.507284	0.980221	
3	0.638691	-0.266348	0.136301	0.646183	-0.507284	-1.859351	
4	0.419418	0.769612	0.922662	1.355551	-0.507284	0.947203	
...	...	...	...	...	...	...	
1454	1.821276	-3.450727	-2.993401	-1.481920	1.289758	-0.043346	
1455	1.821276	-3.450727	-3.036401	-1.481920	-0.507284	-0.043346	
1456	-1.120845	2.764091	1.586172	-0.772552	1.289758	-0.373528	
1457	0.915540	-0.165615	0.311255	-0.772552	-0.507284	0.683057	
1458	0.419418	0.378796	0.152052	0.646183	-0.507284	0.716075	

2919 rows × 292 columns

In [225…
```
df_train = all_data.iloc[:1460,:]
df_test = all_data.iloc[1460:,:]
final = pd.concat([df_train,y], axis = 1)
```

In [226…
```
final
```

Out[226]:

	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearR
0	0.419418	-0.020358	-0.103719	0.646183	-0.507284	1.046258	
1	-1.120845	0.619103	0.146544	-0.063185	2.188279	0.154764	-
2	0.419418	0.118440	0.457629	0.646183	-0.507284	0.980221	
3	0.638691	-0.266348	0.136301	0.646183	-0.507284	-1.859351	-
4	0.419418	0.769612	0.922662	1.355551	-0.507284	0.947203	
...	...	...	...	...	...	...	
1455	0.419418	-0.165615	-0.231508	-0.063185	-0.507284	0.914184	
1456	-1.120845	0.806133	0.767440	-0.063185	0.391237	0.220801	
1457	0.638691	0.026597	0.029092	0.646183	3.086800	-1.000876	
1458	-1.120845	0.118440	0.170303	-0.772552	0.391237	-0.703711	
1459	-1.120845	0.420154	0.214215	-0.772552	0.391237	-0.208437	-

1460 rows × 293 columns

In [231…
```
from sklearn.linear_model import BayesianRidge, HuberRegressor, Ridge, Or
from sklearn.ensemble import GradientBoostingRegressor
from catboost import CatBoostRegressor
```

In [270…
```
baseline_model = GradientBoostingRegressor()
baseline_model.fit(df_train, y)
```

Out[270]:
```
GradientBoostingRegressor()
```

In [287…
```
br_params = {
    'n_iter': 304,
    'tol': 0.16864712769300896,
    'alpha_1': 5.589616542154059e-07,
    'alpha_2': 9.799343618469923,
    'lambda_1': 1.7735725582463822,
    'lambda_2': 3.616928181181732e-06
}


ridge_params = {
    'alpha': 10
}
```

In [295…
```
models = {'gbr':GradientBoostingRegressor(),
          'br':BayesianRidge(**br_params),
          'ridge':Ridge(**ridge_params),
          'catboost':CatBoostRegressor(loss_function='RMSE',n_estimators=
```

In [274…
```
for name, model in models.items():
    model.fit(df_train, y)
```

In [275…
```
results = {}

kf = KFold(n_splits=10)

for name, model in models.items():
    result = np.exp(np.sqrt(-cross_val_score(model, df_train, y, scoring=
    results[name] = result
```

In [276…
```
for name, result in results.items():
    print("----------\n" + name)
    print(np.mean(result))
    print(np.std(result))
```

```
----------
gbr
1.1334937793062636
0.02041514019070494
----------
br
1.136720968542448
0.02708918872877562
----------
ridge
1.1377247226437417
0.025916296541081737
----------
catboost
1.1237199881446363
0.02114096999987419
```

In [291…
```
y_pred = (
    0.0 * np.exp(models['gbr'].predict(df_test)) +
    0.0 * np.exp(models['br'].predict(df_test)) +
    0.0 * np.exp(models['ridge'].predict(df_test))+
    1 * np.exp(models['catboost'].predict(df_test)))
```

In [292…
```
solution = pd.DataFrame({"id":test.Id, "SalePrice":y_pred})
# solution.to_csv("ridge_sol.csv", index = False)
```

In [293…
```
solution.to_csv('submission.csv',index=False) #Score: 0.12299
```

In [294…
```
solution
```

Out[294]:

	id	SalePrice
0	1461	125953.825089
1	1462	161424.964322
2	1463	187908.591855
3	1464	196804.694788
4	1465	183519.623195
...	...	...
1454	2915	80918.325435
1455	2916	78242.653140
1456	2917	160394.307425
1457	2918	113555.787541
1458	2919	224401.442934

1459 rows × 2 columns

Using ensemble model with some feature engineering gives us better result at public leaderboard 0.124, than by using just xgboost 0.132