In [3]:
```
from numpy import genfromtxt
from sklearn.cross_validation import KFold
from sklearn.linear_model import LinearRegression, Lasso, Ridge
import numpy as np
import pylab as pl
from sklearn import linear_model
import math

path=r'C:\Users\pegah\Desktop\univ\Courses\data mining\HW4\P7\blogData_train.csv'
data_train = genfromtxt(path, delimiter=',')

path2=r'C:\Users\pegah\Desktop\univ\Courses\data mining\HW4\P7\blogData_test-2012
data_test=genfromtxt(path2, delimiter=',')

x_train=data_train[:,0:data_train.shape[1]-1]
y_train=data_train[:,data_train.shape[1]-1]
x_test=data_test[:,0:data_test.shape[1]-1]
y_test=data_test[:,data_test.shape[1]-1]
# In order to do multiple regression we need to add a column of 1s for x
x_train = np.array([np.concatenate((v,[1])) for v in x_train])
x_test= np.array([np.concatenate((v,[1])) for v in x_test])
```

In [43]:
```
#fitting and cross validation
n=10
for name,met in [
        ('linear regression', LinearRegression()),
        ('lasso', Lasso()),
        ('ridge', Ridge()),
        ]:
    met.fit(x_train,y_train)
    # p = np.array([met.predict(xi) for xi in x])
    p = met.predict(x_train)
    e = abs(p-y_train)
    total_error = np.dot(e,e)
    rmse_train = np.sqrt(total_error/len(p))

    kf = KFold(len(x_train), n_folds=n)
    err = 0
    for train,test in kf:
        met.fit(x_train[train],y_train[train])
        y_pred = met.predict(x_train[test])
        e = abs(y_pred-y_train[test])
        rmse = np.sqrt(np.dot(e,e)/len(y_train[test]))
        err+=rmse
    y_pred=met.predict(x_test)
    error= abs(y_pred-y_test)
    total_error = np.dot(error,error)
    lsr_rmse = np.sqrt(total_error/len(y_pred))

    rmse_10cv = err/n
    print('Method: %s' %name)
    print('RMSE on training: %.4f' %rmse_train)
    print('RMSE on 10-fold CV: %.4f' %rmse_10cv)
    print('RMSE of prediction: %.4f' %lsr_rmse)
    print('\n')
```

```
Method: linear regression
RMSE on training: 30.0526
RMSE on 10-fold CV: 237.8835
RMSE of prediction: 40.3691



C:\Users\pegah\Anaconda3\lib\site-packages\sklearn\linear_model\coordinate_desc
ent.py:491: ConvergenceWarning: Objective did not converge. You might want to i
ncrease the number of iterations. Fitting data with very small alpha may cause
 precision problems.
  ConvergenceWarning)

Method: lasso
RMSE on training: 30.1980
RMSE on 10-fold CV: 25.9631
RMSE of prediction: 40.8184


Method: ridge
RMSE on training: 30.0544
RMSE on 10-fold CV: 26.2895
RMSE of prediction: 40.3761
```

In [44]:
```
#Lasso hyperparameter optimization using cross_val
lasso=linear_model.Lasso()
#no cross_val
model_init=lasso.fit(x_train,y_train)
yp_init=lasso.predict(x_test)
error_init=abs(yp_init-y_test)
rmse_init=np.sqrt(np.dot(error_init,error_init)/len(y_test))
par_init=lasso.get_params()            #par_init['alpha']
```

```
C:\Users\pegah\Anaconda3\lib\site-packages\sklearn\linear_model\coordinate_desc
ent.py:491: ConvergenceWarning: Objective did not converge. You might want to i
ncrease the number of iterations. Fitting data with very small alpha may cause
 precision problems.
  ConvergenceWarning)
```

In [45]:
```
#cross_val
kf = KFold(len(x_train), n_folds=10)
err = 0
models={}
i=0
best_rmse=math.inf
alpha_set=np.arange(1,2.1,0.1)
for train,test in kf:
    lasso=linear_model.Lasso(alpha=alpha_set[i])
    models[i]=lasso.fit(x_train[train],y_train[train])
    yp_cv = lasso.predict(x_train[test])
    error_cv = abs(yp_cv-y_train[test])
    rmse_cv = np.sqrt(np.dot(error_cv,error_cv)/len(y_train[test]))
    if rmse_cv<best_rmse:
        best_alpha= lasso.get_params()['alpha']
        best_rmse=rmse_cv
    i+=1
```

```
C:\Users\pegah\Anaconda3\lib\site-packages\sklearn\linear_model\coordinate_desc
ent.py:491: ConvergenceWarning: Objective did not converge. You might want to i
ncrease the number of iterations. Fitting data with very small alpha may cause
 precision problems.
  ConvergenceWarning)
C:\Users\pegah\Anaconda3\lib\site-packages\sklearn\linear_model\coordinate_desc
ent.py:491: ConvergenceWarning: Objective did not converge. You might want to i
ncrease the number of iterations. Fitting data with very small alpha may cause
 precision problems.
  ConvergenceWarning)
C:\Users\pegah\Anaconda3\lib\site-packages\sklearn\linear_model\coordinate_desc
ent.py:491: ConvergenceWarning: Objective did not converge. You might want to i
ncrease the number of iterations. Fitting data with very small alpha may cause
 precision problems.
  ConvergenceWarning)
C:\Users\pegah\Anaconda3\lib\site-packages\sklearn\linear_model\coordinate_desc
ent.py:491: ConvergenceWarning: Objective did not converge. You might want to i
ncrease the number of iterations. Fitting data with very small alpha may cause
 precision problems.
  ConvergenceWarning)
```

In [46]:
```
lasso=linear_model.Lasso(alpha=best_alpha)
best_model=lasso.fit(x_train,y_train)
yp=lasso.predict(x_test)
error= abs(yp-y_test)
total_error = np.dot(error,error)
rmse = np.sqrt(total_error/len(y_pred))
```

```
C:\Users\pegah\Anaconda3\lib\site-packages\sklearn\linear_model\coordinate_desc
ent.py:491: ConvergenceWarning: Objective did not converge. You might want to i
ncrease the number of iterations. Fitting data with very small alpha may cause
 precision problems.
  ConvergenceWarning)
```

In [47]:
```
print('test rmse without Lasso=',rmse_init)
print('CrossValidation rmse=',rmse_cv)
print('model with best lambda rmse=',rmse)
print('best lambda=',best_alpha)
```

```
test rmse without Lasso= 40.8368239929
CrossValidation rmse= 14.090779857
model with best lambda rmse= 40.9179298093
best lambda= 1.9
```

In [52]:
```
weights=lasso.coef_
```

In [106]:
```
imp_weigths_index=sorted(range(len(weights)), key=lambda k: weights[k],reverse=Tr
imp_weigths=sorted(weights,reverse=True)
```

In [107]:
```
imp_weigths_index
```

Out[107]:
```
[5,
 9,
 51,
 1,
 4,
 54,
 0,
 8,
 13,
 3,
 61,
 2,
 6,
 7,
 10,
 11,
 12,
 14,
 17,
```

In [108]:
```
imp_weigths
```

Out[108]:
```
[0.61912399138060104,
 0.18849619431082415,
 0.18185289530295984,
 0.12746181960043274,
 0.12718710836520086,
 0.039437523594102589,
 0.011089773240268086,
 0.0065768932816394699,
 0.0017982449545209372,
 0.0010944460941809686,
 0.00017639869675468973,
 -0.0,
 0.0,
 -0.0,
 0.0,
 0.0,
 0.0,
 0.0,
 -0.0,
```

In [ ]:
```
```