Week 3. Running a Lasso Regression Analysis

June 30, 2016

In [1]:
```
#
# Created on Mon Dec 14 16:26:46 2015
#
# @author: jrose01
# Modified by: mcolosso
#
```

In [2]:
```
%matplotlib inline

from pandas import Series, DataFrame
import pandas as pd
import numpy as np
import os
import matplotlib.pylab as plt
from sklearn.cross_validation import train_test_split
from sklearn.linear_model import LassoLarsCV

#pd.set_option('display.float_format', lambda x:'%.3f'%x)
#pd.set_option('display.mpl_style', 'default')   # --deprecated
#plt.style.use('ggplot')   # Make the graphs a bit prettier
plt.rcParams['figure.figsize'] = (15, 5)
```

In [3]:
```
#os.chdir("C:/Users/MColosso/Documents/CURSOS/Wesleyan University/Machine Learning for Data Ana
```

In [4]:
```
#
# Data Engineering and Analysis
#
```

In [5]:
```
#Load the dataset

loans = pd.read_csv("./LendingClub.csv", low_memory = False)

# LendingClub.csv is a dataset taken from The LendingClub (https://www.lendingclub.com/)
# which is a peer-to-peer leading company that directly connects borrowers and potential
# lenders/investors
```

In [6]:
```
#
# Exploring the target column
#
```

In [7]:
```
# The target column (label column) of the dataset that we are interested in is called
# 'bad_loans'. In this column **1** means a risky (bad) loan **0** means a safe  loan.
#
# In order to make this more intuitive, we reassign the target to be:
# * ** 1 ** as a safe  loan,
```

```
# * ** 0 ** as a risky (bad) loan.
#
# We put this in a new column called `safe_loans`.

loans['safe_loans'] = loans['bad_loans'].apply(lambda x : 1 if x == 0 else 0)
loans.drop('bad_loans', axis = 1, inplace = True)
```

In [8]:
```
# Select features to handle

predictors = ['grade',                     # grade of the loan
              'sub_grade',                 # sub-grade of the loan
              'short_emp',                 # one year or less of employment
              'emp_length_num',            # number of years of employment
              'home_ownership',            # home_ownership status: own, mortgage or rent
              'dti',                       # debt to income ratio
              'purpose',                   # the purpose of the loan
              'term',                      # the term of the loan
              'last_delinq_none',          # has borrower had a delinquincy
              'last_major_derog_none',     # has borrower had 90 day or worse rating
              'revol_util',                # percent of available credit being used
              'total_rec_late_fee',        # total late fees received to day
             ]

target = 'safe_loans'                      # prediction target (y) (+1 means safe, 0 is risky)

# Extract the predictors and target columns
loans = loans[predictors + [target]]

# Delete rows where any or all of the data are missing
data_clean = loans.dropna()
```

In [9]:
```
# Convert categorical variables into binary variables

data_clean = pd.get_dummies(data_clean, prefix_sep = '=')
```

In [10]:
```
(data_clean.describe()).T
```

Out[10]:

	count	mean	std	min	25%	50%	\
short_emp	122607.0	0.123672	0.329208	0.0	0.00	0.00	
emp_length_num	122607.0	6.370256	3.736014	0.0	3.00	6.00	
dti	122607.0	15.496888	7.497442	0.0	9.88	15.26	
last_delinq_none	122607.0	0.588115	0.492177	0.0	0.00	1.00	
last_major_derog_none	122607.0	0.873906	0.331957	0.0	1.00	1.00	
revol_util	122607.0	53.716307	25.723881	0.0	34.80	55.70	
total_rec_late_fee	122607.0	0.742344	5.363268	0.0	0.00	0.00	
safe_loans	122607.0	0.811185	0.391363	0.0	1.00	1.00	
grade=A	122607.0	0.181996	0.385843	0.0	0.00	0.00	
grade=B	122607.0	0.303180	0.459634	0.0	0.00	0.00	
grade=C	122607.0	0.244276	0.429659	0.0	0.00	0.00	
grade=D	122607.0	0.156394	0.363230	0.0	0.00	0.00	
grade=E	122607.0	0.073324	0.260668	0.0	0.00	0.00	
grade=F	122607.0	0.032070	0.176187	0.0	0.00	0.00	
grade=G	122607.0	0.008760	0.093183	0.0	0.00	0.00	
sub_grade=A1	122607.0	0.024362	0.154172	0.0	0.00	0.00	
sub_grade=A2	122607.0	0.027339	0.163071	0.0	0.00	0.00	

sub_grade=A3	122607.0	0.032258	0.176684	0.0	0.00	0.00
sub_grade=A4	122607.0	0.048880	0.215617	0.0	0.00	0.00
sub_grade=A5	122607.0	0.049157	0.216197	0.0	0.00	0.00
sub_grade=B1	122607.0	0.047607	0.212935	0.0	0.00	0.00
sub_grade=B2	122607.0	0.057876	0.233510	0.0	0.00	0.00
sub_grade=B3	122607.0	0.073699	0.261281	0.0	0.00	0.00
sub_grade=B4	122607.0	0.067525	0.250930	0.0	0.00	0.00
sub_grade=B5	122607.0	0.056473	0.230834	0.0	0.00	0.00
sub_grade=C1	122607.0	0.057648	0.233077	0.0	0.00	0.00
sub_grade=C2	122607.0	0.054858	0.227704	0.0	0.00	0.00
sub_grade=C3	122607.0	0.046408	0.210369	0.0	0.00	0.00
sub_grade=C4	122607.0	0.044059	0.205228	0.0	0.00	0.00
sub_grade=C5	122607.0	0.041303	0.198990	0.0	0.00	0.00
...	...	...	...	...	...	...
sub_grade=E4	122607.0	0.012895	0.112821	0.0	0.00	0.00
sub_grade=E5	122607.0	0.011092	0.104735	0.0	0.00	0.00
sub_grade=F1	122607.0	0.009013	0.094506	0.0	0.00	0.00
sub_grade=F2	122607.0	0.007585	0.086763	0.0	0.00	0.00
sub_grade=F3	122607.0	0.006280	0.078999	0.0	0.00	0.00
sub_grade=F4	122607.0	0.005130	0.071442	0.0	0.00	0.00
sub_grade=F5	122607.0	0.004062	0.063603	0.0	0.00	0.00
sub_grade=G1	122607.0	0.003018	0.054852	0.0	0.00	0.00
sub_grade=G2	122607.0	0.001966	0.044292	0.0	0.00	0.00
sub_grade=G3	122607.0	0.001362	0.036881	0.0	0.00	0.00
sub_grade=G4	122607.0	0.001240	0.035188	0.0	0.00	0.00
sub_grade=G5	122607.0	0.001174	0.034251	0.0	0.00	0.00
home_ownership=MORTGAGE	122607.0	0.483170	0.499719	0.0	0.00	0.00
home_ownership=OTHER	122607.0	0.001460	0.038182	0.0	0.00	0.00
home_ownership=OWN	122607.0	0.081097	0.272984	0.0	0.00	0.00
home_ownership=RENT	122607.0	0.434274	0.495663	0.0	0.00	0.00
purpose=car	122607.0	0.019371	0.137825	0.0	0.00	0.00
purpose=credit_card	122607.0	0.179843	0.384058	0.0	0.00	0.00
purpose=debt_consolidation	122607.0	0.556518	0.496797	0.0	0.00	1.00
purpose=home_improvement	122607.0	0.061522	0.240286	0.0	0.00	0.00
purpose=house	122607.0	0.008197	0.090165	0.0	0.00	0.00
purpose=major_purchase	122607.0	0.031621	0.174991	0.0	0.00	0.00
purpose=medical	122607.0	0.013107	0.113733	0.0	0.00	0.00
purpose=moving	122607.0	0.009624	0.097630	0.0	0.00	0.00
purpose=other	122607.0	0.074115	0.261959	0.0	0.00	0.00
purpose=small_business	122607.0	0.026622	0.160976	0.0	0.00	0.00
purpose=vacation	122607.0	0.007014	0.083457	0.0	0.00	0.00
purpose=wedding	122607.0	0.012446	0.110867	0.0	0.00	0.00
term= 36 months	122607.0	0.797679	0.401732	0.0	1.00	1.00
term= 60 months	122607.0	0.202321	0.401732	0.0	0.00	0.00

	75%	max
short_emp	0.00	1.00
emp_length_num	11.00	11.00
dti	20.85	39.88
last_delinq_none	1.00	1.00
last_major_derog_none	1.00	1.00
revol_util	74.30	150.70
total_rec_late_fee	0.00	208.82
safe_loans	1.00	1.00

grade=A	0.00	1.00
grade=B	1.00	1.00
grade=C	0.00	1.00
grade=D	0.00	1.00
grade=E	0.00	1.00
grade=F	0.00	1.00
grade=G	0.00	1.00
sub_grade=A1	0.00	1.00
sub_grade=A2	0.00	1.00
sub_grade=A3	0.00	1.00
sub_grade=A4	0.00	1.00
sub_grade=A5	0.00	1.00
sub_grade=B1	0.00	1.00
sub_grade=B2	0.00	1.00
sub_grade=B3	0.00	1.00
sub_grade=B4	0.00	1.00
sub_grade=B5	0.00	1.00
sub_grade=C1	0.00	1.00
sub_grade=C2	0.00	1.00
sub_grade=C3	0.00	1.00
sub_grade=C4	0.00	1.00
sub_grade=C5	0.00	1.00
...	...	...
sub_grade=E4	0.00	1.00
sub_grade=E5	0.00	1.00
sub_grade=F1	0.00	1.00
sub_grade=F2	0.00	1.00
sub_grade=F3	0.00	1.00
sub_grade=F4	0.00	1.00
sub_grade=F5	0.00	1.00
sub_grade=G1	0.00	1.00
sub_grade=G2	0.00	1.00
sub_grade=G3	0.00	1.00
sub_grade=G4	0.00	1.00
sub_grade=G5	0.00	1.00
home_ownership=MORTGAGE	1.00	1.00
home_ownership=OTHER	0.00	1.00
home_ownership=OWN	0.00	1.00
home_ownership=RENT	1.00	1.00
purpose=car	0.00	1.00
purpose=credit_card	0.00	1.00
purpose=debt_consolidation	1.00	1.00
purpose=home_improvement	0.00	1.00
purpose=house	0.00	1.00
purpose=major_purchase	0.00	1.00
purpose=medical	0.00	1.00
purpose=moving	0.00	1.00
purpose=other	0.00	1.00
purpose=small_business	0.00	1.00
purpose=vacation	0.00	1.00
purpose=wedding	0.00	1.00
term= 36 months	1.00	1.00
term= 60 months	0.00	1.00

```
[68 rows x 8 columns]
```

In [11]:
```
# Extract new features names

features = data_clean.columns.values
features = features[features != target]
```

In [12]:
```
#
# Modeling and Prediction
#
```

In [13]:
```
predvar    = data_clean[features]
predictors = predvar.copy()
target     = data_clean.safe_loans
```

In [14]:
```
# Standardize predictors to have mean=0 and sd=1

from sklearn import preprocessing
for attr in predictors.columns.values:
    predictors[attr] = preprocessing.scale(predictors[attr].astype('float64'))
```

In [15]:
```
#Split into training and testing sets

pred_train, pred_test, tar_train, tar_test = train_test_split(predictors, target,
                                                              test_size = .4,
                                                              random_state = 123)


print('pred_train.shape', pred_train.shape)
print('pred_test.shape',  pred_test.shape)
print('tar_train.shape',  tar_train.shape)
print('tar_test.shape',   tar_test.shape)
```

```
pred_train.shape (73564, 67)
pred_test.shape (49043, 67)
tar_train.shape (73564,)
tar_test.shape (49043,)
```

In [16]:
```
# Specify the lasso regression model

model = LassoLarsCV(cv = 10, precompute = False).fit(pred_train, tar_train)
```

In [17]:
```
# Print variable names and regression coefficients

pd.DataFrame([dict(zip(predictors.columns, model.coef_))], index=['coef']).T
```

Out[17]:

	coef
dti	-0.031080
emp_length_num	0.000000
grade=A	0.036706
grade=B	0.015090
grade=C	0.000000
grade=D	-0.014125
grade=E	-0.017393
grade=F	-0.015787
grade=G	-0.009659
home_ownership=MORTGAGE	0.010588

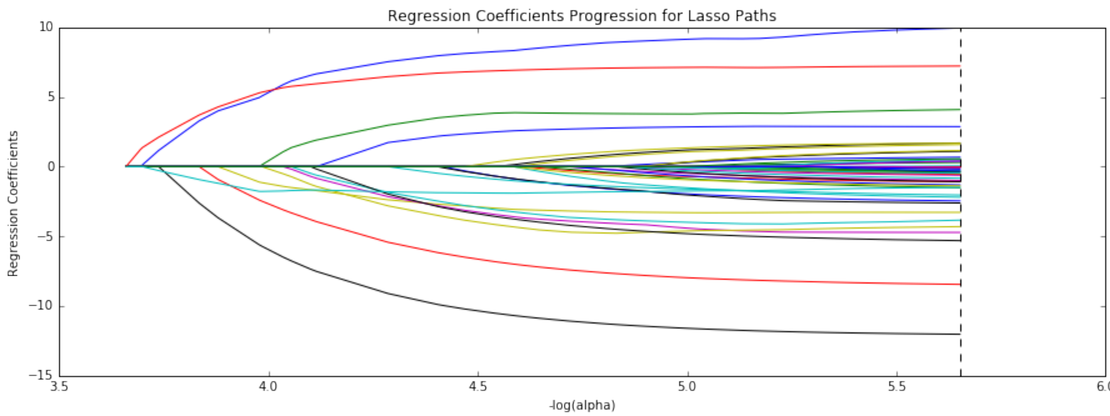
```
home_ownership=OTHER       -0.001163
home_ownership=OWN          0.000000
home_ownership=RENT        -0.007503
last_delinq_none           -0.008092
last_major_derog_none      -0.002272
purpose=car                 0.000000
purpose=credit_card         0.006216
purpose=debt_consolidation  0.000000
purpose=home_improvement   -0.001184
purpose=house               0.000000
purpose=major_purchase      0.000000
purpose=medical            -0.003115
purpose=moving             -0.000223
purpose=other              -0.005091
purpose=small_business     -0.019715
purpose=vacation           -0.000638
purpose=wedding             0.002051
revol_util                 -0.012068
short_emp                  -0.009104
sub_grade=A1                0.002509
...                              ...
sub_grade=B4                0.000000
sub_grade=B5               -0.001996
sub_grade=C1                0.001869
sub_grade=C2                0.001481
sub_grade=C3                0.000000
sub_grade=C4               -0.001447
sub_grade=C5                0.000000
sub_grade=D1                0.000000
sub_grade=D2                0.000000
sub_grade=D3               -0.000327
sub_grade=D4               -0.003696
sub_grade=D5               -0.001762
sub_grade=E1                0.004096
sub_grade=E2                0.000000
sub_grade=E3               -0.001249
sub_grade=E4               -0.003977
sub_grade=E5               -0.001715
sub_grade=F1                0.000000
sub_grade=F2                0.000000
sub_grade=F3               -0.003955
sub_grade=F4               -0.004832
sub_grade=F5               -0.005503
sub_grade=G1               -0.000474
sub_grade=G2               -0.001004
sub_grade=G3                0.000000
sub_grade=G4                0.004561
sub_grade=G5                0.000000
term= 36 months             0.026618
term= 60 months            -0.005477
total_rec_late_fee         -0.044206

[67 rows x 1 columns]
```

In [18]:
```
# Plot coefficient progression

m_log_alphas = -np.log10(model.alphas_)
ax = plt.gca()
plt.plot(m_log_alphas, model.coef_path_.T)
plt.axvline(-np.log10(model.alpha_), linestyle = '--', color = 'k',
            label = 'alpha CV')
plt.ylabel('Regression Coefficients')
plt.xlabel('-log(alpha)')
plt.title('Regression Coefficients Progression for Lasso Paths')
```

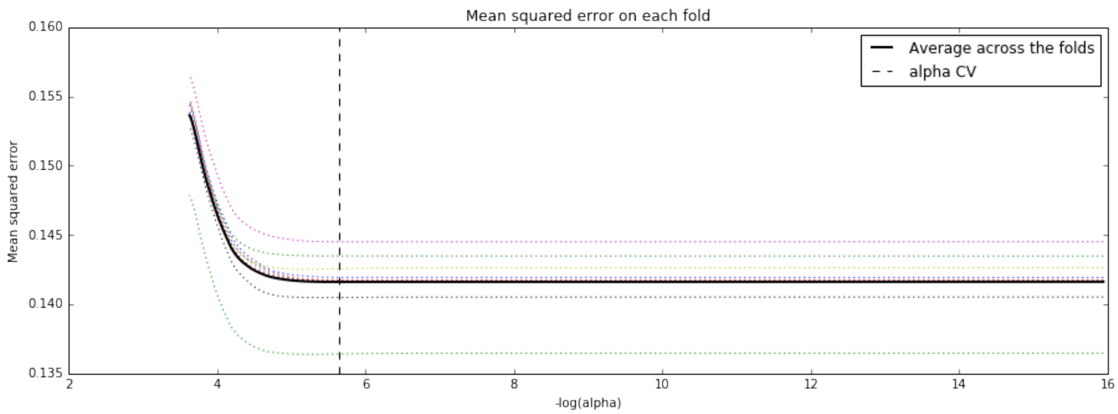
Out[18]: <matplotlib.text.Text at 0x17aaab94390>



In [19]:
```
# Plot mean square error for each fold

m_log_alphascv = -np.log10(model.cv_alphas_)
plt.figure()
plt.plot(m_log_alphascv, model.cv_mse_path_, ':')
plt.plot(m_log_alphascv, model.cv_mse_path_.mean(axis = -1), 'k',
         label = 'Average across the folds', linewidth = 2)
plt.axvline(-np.log10(model.alpha_), linestyle = '--', color = 'k',
            label = 'alpha CV')
plt.legend()
plt.xlabel('-log(alpha)')
plt.ylabel('Mean squared error')
plt.title('Mean squared error on each fold')
```

Out[19]: <matplotlib.text.Text at 0x17aac05cef0>



In [20]:
```
# MSE from training and test data

from sklearn.metrics import mean_squared_error
train_error = mean_squared_error(tar_train, model.predict(pred_train))
test_error = mean_squared_error(tar_test, model.predict(pred_test))
print ('training data MSE')
print(train_error)
print ('test data MSE')
print(test_error)
```

```
training data MSE
0.141354906717
test data MSE
0.140656085708
```

In [21]:
```
# R-square from training and test data

rsquared_train = model.score(pred_train, tar_train)
rsquared_test  = model.score(pred_test, tar_test)
print ('training data R-square')
print(rsquared_train)
print ('test data R-square')
print(rsquared_test)
```

```
training data R-square
0.0799940399148
test data R-square
0.0772929635462
```

In [ ]: