6.6.2 The Lasso

May 18, 2018

```
In [1]: # conventional way to import pandas
        import pandas as pd
        # conventional way to import seaborn
        import seaborn as sns
        # conventional way to import numpy
        import numpy as np

        from sklearn import metrics
        import matplotlib.pyplot as plt

        data = pd.read_csv("https://vincentarelbundock.github.io/Rdatasets/csv/ISLR/Hitters.csv

        data.head()
```

```
Out[1]:                    AtBat  Hits  HmRun  Runs  RBI  Walks  Years  CAtBat  CHits  \
        -Andy Allanson       293    66      1    30   29     14      1     293     66
        -Alan Ashby          315    81      7    24   38     39     14    3449    835
        -Alvin Davis         479   130     18    66   72     76      3    1624    457
        -Andre Dawson        496   141     20    65   78     37     11    5628   1575
        -Andres Galarraga    321    87     10    39   42     30      2     396    101

                           CHmRun  CRuns  CRBI  CWalks League Division  PutOuts  \
        -Andy Allanson          1     30    29      14      A        E      446
        -Alan Ashby            69    321   414     375      N        W      632
        -Alvin Davis           63    224   266     263      A        W      880
        -Andre Dawson         225    828   838     354      N        E      200
        -Andres Galarraga      12     48    46      33      N        E      805

                           Assists  Errors  Salary NewLeague
        -Andy Allanson          33      20     NaN         A
        -Alan Ashby             43      10   475.0         N
        -Alvin Davis            82      14   480.0         A
        -Andre Dawson           11       3   500.0         N
        -Andres Galarraga       40       4    91.5         N
```

After listing the data we can see that some have missing data for their Salary. Next drop all the rows that contain NaN data.

```
In [2]: data = data.dropna()
        data.index.name = 'Player'
        data.head()
```

```
Out[2]:                    AtBat  Hits  HmRun  Runs  RBI  Walks  Years  CAtBat  CHits  \
        Player
        -Alan Ashby          315    81      7    24   38     39     14    3449    835
        -Alvin Davis         479   130     18    66   72     76      3    1624    457
        -Andre Dawson        496   141     20    65   78     37     11    5628   1575
        -Andres Galarraga    321    87     10    39   42     30      2     396    101
        -Alfredo Griffin     594   169      4    74   51     35     11    4408   1133

                           CHmRun  CRuns  CRBI  CWalks League Division  PutOuts  \
        Player
        -Alan Ashby            69    321   414     375      N        W      632
        -Alvin Davis           63    224   266     263      A        W      880
        -Andre Dawson         225    828   838     354      N        E      200
        -Andres Galarraga      12     48    46      33      N        E      805
        -Alfredo Griffin       19    501   336     194      A        W      282

                           Assists  Errors  Salary NewLeague
        Player
        -Alan Ashby             43      10   475.0         N
        -Alvin Davis            82      14   480.0         A
        -Andre Dawson           11       3   500.0         N
        -Andres Galarraga       40       4    91.5         N
        -Alfredo Griffin       421      25   750.0         A
```

In the lab they use a R funktion that we don't have in python. That funktion automatically transforms any qualitative variables into dummy variables. But in python we will have to do this by hand using the following code and display infomation about the variables we converted from strings to numbers.

```
In [3]: dummieVariables = pd.get_dummies(data[['League', 'Division', 'NewLeague']])
        dummieVariables.info()
        print(dummieVariables.head())
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 263 entries, -Alan Ashby to -Willie Wilson
Data columns (total 6 columns):
League_A       263 non-null uint8
League_N       263 non-null uint8
Division_E     263 non-null uint8
Division_W     263 non-null uint8
NewLeague_A    263 non-null uint8
NewLeague_N    263 non-null uint8
dtypes: uint8(6)
memory usage: 3.6+ KB
                   League_A  League_N  Division_E  Division_W  NewLeague_A  \
Player
-Alan Ashby                0          1           0           1            0
-Alvin Davis               1          0           0           1            1
-Andre Dawson              0          1           1           0            0
-Andres Galarraga          0          1           1           0            0
-Alfredo Griffin           1          0           0           1            1

                   NewLeague_N
Player
-Alan Ashby                  1
-Alvin Davis                 0
-Andre Dawson                1
-Andres Galarraga            1
-Alfredo Griffin             0
```

Next we must remove the columns with our independent variable (Salary), and columns for which we created dummy variables and reinterduce them into our predictors. This is do so the data fit the data in the book.

```
In [4]: y = data.Salary
        X_ = data.drop(['Salary', 'League', 'Division', 'NewLeague'], axis=1).astype('float64'

        X = pd.concat([X_, dummieVariables[['League_N', 'Division_W', 'NewLeague_N']]], axis=1
        X.head()
```

```
Out[4]:                    AtBat   Hits  HmRun  Runs   RBI  Walks  Years  CAtBat  \
        Player
        -Alan Ashby        315.0   81.0    7.0  24.0  38.0   39.0   14.0  3449.0
        -Alvin Davis       479.0  130.0   18.0  66.0  72.0   76.0    3.0  1624.0
        -Andre Dawson      496.0  141.0   20.0  65.0  78.0   37.0   11.0  5628.0
        -Andres Galarraga  321.0   87.0   10.0  39.0  42.0   30.0    2.0   396.0
        -Alfredo Griffin   594.0  169.0    4.0  74.0  51.0   35.0   11.0  4408.0

                            CHits  CHmRun  CRuns   CRBI  CWalks  PutOuts  Assists  \
        Player
        -Alan Ashby         835.0    69.0  321.0  414.0   375.0    632.0     43.0
        -Alvin Davis        457.0    63.0  224.0  266.0   263.0    880.0     82.0
        -Andre Dawson      1575.0   225.0  828.0  838.0   354.0    200.0     11.0
        -Andres Galarraga   101.0    12.0   48.0   46.0    33.0    805.0     40.0
        -Alfredo Griffin   1133.0    19.0  501.0  336.0   194.0    282.0    421.0

                           Errors  League_N  Division_W  NewLeague_N
        Player
        -Alan Ashby          10.0         1           1            1
        -Alvin Davis         14.0         0           1            0
        -Andre Dawson         3.0         1           0            1
        -Andres Galarraga     4.0         1           0            1
        -Alfredo Griffin     25.0         0           1            0
```

Now we spilt the data into a tranining and testing set like the book.

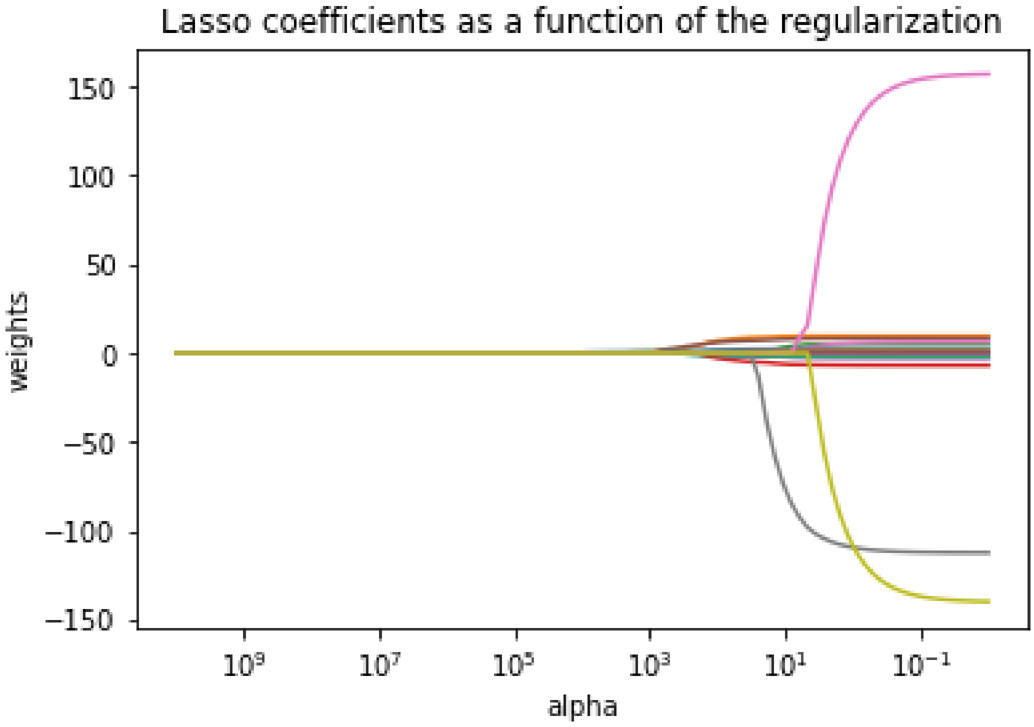
```
In [5]: from sklearn.model_selection import train_test_split
        X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=10)
```

```
In [6]: ##Ploting code from scikit-learn.org just changed to use lasso.
        ## http://scikit-learn.org/stable/auto_examples/linear_model/plot_ridge_path.html#sphx
        from sklearn.preprocessing import scale
        from sklearn.linear_model import Lasso

        alphas = 10**np.linspace(10,-2,100)*0.5
        lasso = Lasso(max_iter=100000)
        coefs = []

        for a in alphas*2:
            lasso.set_params(alpha=a)
            lasso.fit(X_train, y_train)
            coefs.append(lasso.coef_)

        ax = plt.gca()
        ax.plot(alphas*2, coefs)
        ax.set_xscale('log')
        ax.set_xlim(ax.get_xlim()[::-1])  # reverse axis
        plt.axis('tight')
        plt.xlabel('alpha')
        plt.ylabel('weights')
        plt.title('Lasso coefficients as a function of the regularization');
```



Next we try to get the best lamda using cross validation.

```
In [7]: from sklearn.linear_model import LassoCV
        alphas = 10**np.linspace(100,-20,100)
        lassoCV = LassoCV(alphas=alphas)
        lassoCV.fit(X_train, y_train)
        lassoCV.alpha_
```

```
C:\Users\entvex\Anaconda3\lib\site-packages\sklearn\linear_model\coordinate_descent.py:491: Co
  ConvergenceWarning)
```

```
Out[7]: 1072.2672220103254
```

```
In [8]: pd.Series(lassoCV.coef_.flatten(), index=X.columns)
```

```
Out[8]: AtBat          0.729684
        Hits           0.000000
        HmRun         -0.000000
        Runs           0.000000
        RBI            0.000000
        Walks          0.000000
        Years         -0.000000
        CAtBat        -0.241005
        CHits          0.705141
        CHmRun         0.000000
        CRuns          0.579498
        CRBI           0.442905
        CWalks        -0.000000
        PutOuts        0.232895
        Assists        0.000000
        Errors        -0.000000
        League_N       0.000000
        Division_W    -0.000000
        NewLeague_N    0.000000
        dtype: float64
```