1. Ridge, RASSO 코드 돌려보기

Import Data

import numpy as np import pandas as pd from sklearn.model_selection import train_test_split from sklearn.linear_model import LinearRegression, Ridge, Lasso import matplotlib.pyplot as plt from sklearn.metrics import mean_squared_error

In [2]:
```
data = pd.read_csv('data.csv')
```

In [3]:
```
data.head()
#Age: 나이
#Experience: 경력
#Income: 수입
#Family: 가족단위
#CCAvg: 월 카드 사용량
```

Out[3]:

	Age	Experience	Income	Family	CCAvg
0	25	1	49	4	1.6
1	45	19	34	3	1.5
2	39	15	11	1	1.0
3	35	9	100	1	2.7
4	35	8	45	4	1.0

종속변수

In [4]:
```
#결측치 확인(없음)
data.isnull().sum()
```
결측치 유무 확인!

Out[4]:
```
Age           0
Experience    0
Income        0
Family        0
CCAvg         0
dtype: int64
```

In [5]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2500 entries, 0 to 2499
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Age         2500 non-null   int64
 1   Experience  2500 non-null   int64
 2   Income      2500 non-null   int64
 3   Family      2500 non-null   int64
 4   CCAvg       2500 non-null   float64
dtypes: float64(1), int64(4)
memory usage: 97.8 KB
```

test 셋은 몇?

결과데이터

종속변수

In [6]:
```
y = data['Income']
X = data.drop(['Income'], axis = 1)
x_train, x_test, y_train, y_test = train_test_split(X, y, train_size = 0.7, random_state = 1000)
```
독립변수

train, test set 생성 (과적합 방지!)

입력데이터

train 70%
test 30%

1. Linear Regression

In [7]:
```
reg = LinearRegression()
results1 = reg.fit(x_train, y_train)
```

In [8]:
```
reg.coef_
```

Out[8]: array([-3.07793956, 2.89401562, -3.37220023, 16.09065086])

2. Ridge Regression

In [9]:
```
rreg = Ridge(alpha = 0) # alpha = Lambda
rreg.fit(x_train, y_train)
```

Out[9]: Ridge(alpha=0)

In [10]:
```
rreg.coef_
```

Out[10]: array([-3.07793956, 2.89401562, -3.37220023, 16.09065086]) ⇒ λ=0으로하면 Least Square method와 같은결과! (제약X)

In [11]:
```
alpha = np.logspace(-3,3,7)
alpha
```
⇒ -3부터 3까지 log scale 간격으로 7개 출력!

Out[11]: array([1.e-03, 1.e-02, 1.e-01, 1.e+00, 1.e+01, 1.e+02, 1.e+03])

$= 0.001$ ... $= 1000$

In [12]:

횟수 λ
```
df = []
acc_table = []

for i, a in enumerate(alpha):
    rreg = Ridge(alpha=a).fit(x_train, y_train)
    df.append(pd.Series(np.hstack([rreg.intercept_, rreg.coef_])))
    pred_y = rreg.predict(x_test)

df_ridge = pd.DataFrame(df, index = alpha).T
df_ridge
```
λ가 커질수록 →

Out[12]:

상수항

계수

	0.001	0.010	0.100	1.000	10.000	100.000	1000.000
0	132.296084	132.295649	132.291303	132.247877	131.817002	127.823048	105.704966
1	-3.077937	-3.077919	-3.077732	-3.075864	-3.057321	-2.884607	-1.883048
2	2.894014	2.893995	2.893806	2.891920	2.873198	2.698718	1.681685
3	-3.372199	-3.372192	-3.372122	-3.371422	-3.364435	-3.295822	-2.731156
4	16.090648	16.090622	16.090363	16.087768	16.061871	15.807207	13.634454

→ 계수가 shrink된다. (0이되진않음)

3. Lasso Regression

In [13]:
```
lreg = Lasso(alpha = 0 ) # alpha = Lambda
lreg.fit(x_train, y_train)
```

In [14]:

```
lreg.coef_
```

Out[14]:

```
array([-3.07790231,  2.8939786 , -3.37220244, 16.09065156])
```

⇒ λ=0일 때 제약없을 때의 결과와 같다.

In [15]:

```
df = []
acc_table = []

for i, a in enumerate(alpha):
        lreg = Lasso(alpha=a).fit(x_train, y_train)
        df.append(pd.Series(np.hstack([lreg.intercept_, lreg.coef_])))
        pred_y = lreg.predict(x_test)

df_lasso = pd.DataFrame(df, index = alpha).T
df_lasso
```

```
C:\Users\Jeong NeulPum\anaconda3\lib\site-packages\sklearn\linear_model\_coordinate_descent.py:530: ConvergenceWarning: Objective did not converge. You might want to
ease the number of iterations. Duality gap: 3094.37938440009, tolerance: 373.8484092000001
  model = cd_fast.enet_coordinate_descent(
```

Out[15]:

	0.001	0.010	0.100	1.000	10.000	100.000	1000.000
0	132.261976	131.960877	128.945930	98.937749	54.569493	73.876	73.876
1	-3.076625	-3.065044	-2.949074	-1.794975	-0.134206	-0.000	-0.000
2	2.892703	2.881139	2.765340	1.612913	-0.000000	-0.000	-0.000
3	-3.371595	-3.366136	-3.311548	-2.765340	-0.000000	-0.000	-0.000
4	16.090400	16.088142	16.065558	15.839618	13.184919	0.000	0.000

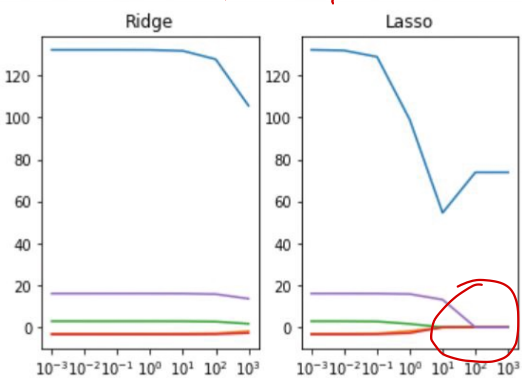
⇒ 마찬가지로 λ가 커질수록 계수가 shrinkage
(0이 나타나는 경우도 있다!)

In [16]:

```
import matplotlib.pyplot as plt

ax1 = plt.subplot(121)
plt.semilogx(df_ridge.T)
plt.xticks(alpha)
plt.title("Ridge")

ax2 = plt.subplot(122)
plt.semilogx(df_lasso.T)
plt.xticks(alpha)
plt.title("Lasso")

plt.show()
```

→ 급격적으로 λ가 커질수록 shrinkage.

y축: 변수들의 계수



0에 도달하면 유지
⇒ selection process 수행

x축: λ (제약의 정도)

2. **Exercise. 3.29**

Suppose we fit a ridge regression with a given shrinkage parameter $\lambda \in \mathbb{R}^+$ on a single variable x_1. (Notice that x_1 is a $N \times 1$ vector.)

1. (Essential) Show that the coefficient must be $\frac{X^T y}{X^T X + \lambda}$ where $X = x_1$.
2. (Essential) We now include an exact copy $x_2 = x_1$, so our new design matrix would be $X = [x_1 | x_2]$. Using this matrix, re-fit our ridge regression. Show that both coefficients are identical, and derive their value.
3. (Extra) Show in general that if m copies of a variable x_j, are included in a ridge regression, so X would be $[x_1 | x_2 | \cdots | x_m]$, their coefficients are all the same.

1. $$\hat{\beta}^{ridge} = \underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j\right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\} \Rightarrow \text{ESL (3.41)}$$

$$= \underset{\beta}{\operatorname{argmin}} \left\{ \|y - X\beta\|^2 + \lambda \|\beta\|^2 \right\} \Rightarrow \text{ESL (3.43) (matrix form)}$$

$\Rightarrow$ β에 대해 미분하면 $-2X^T(y - X\hat{\beta}) + 2\lambda\hat{\beta} = 0$

$\Rightarrow X^T y = X^T X \hat{\beta} + \lambda I \hat{\beta} = (X^T X + \lambda I)\hat{\beta}$

p×p Identity matrix

$$= (X^T X + \lambda I)^{-1} X^T y$$

$$= \boxed{\frac{X^T y}{X^T X + \lambda}}$$

(∵ p=1 이므로 분모가 스칼라값)

2. $p=2$일때!

$$\hat{\beta}^{ridge} = \underset{\beta_1,\beta_2}{\operatorname{argmin}} \{ \| X\beta_1 + X\beta_2 - Y \|^2 + \lambda \|\beta_1\|^2 + \lambda \|\beta_2\|^2 \}$$ ⇒ 편미분!

$$\begin{cases} 2X^T(X\beta_1 + X\beta_2 - Y) + 2\lambda\beta_1 = 0 \\ 2X^T(X\beta_1 + X\beta_2 - Y) + 2\lambda\beta_2 = 0 \end{cases} \Rightarrow -2\lambda\beta_1 = -2\lambda\beta_2 \Rightarrow \boxed{\beta_1 = \beta_2}$$

$$\Rightarrow 2X^T(X\beta_1 + X\beta_1 - Y) + 2\lambda\beta_1 = 0$$

$$\Rightarrow 2X^TX\beta_1 - X^TY + \lambda\beta_1 = 0$$

$$\Rightarrow \beta_1(2X^TX + \lambda) = X^TY$$

$$\Rightarrow \boxed{\beta_1 = \beta_2 = \frac{X^TY}{2X^TX + \lambda}}$$

3. $p=m$일때!

$$\hat{\beta}^{ridge} = \underset{\beta_1,\beta_2,\cdots,\beta_m}{\operatorname{argmin}} \{ \| X\beta_1 + X\beta_2 + \cdots + X\beta_m - Y \|^2 + \lambda \|\beta_1\|^2 + \cdots + \lambda \|\beta_m\|^2 \}$$

⇒ 편미분!

$$\begin{cases} 2X^T(X\beta_1 + X\beta_2 + \cdots + X\beta_m - Y) + 2\lambda\beta_1 = 0 \\ 2X^T(X\beta_1 + X\beta_2 + \cdots + X\beta_m - Y) + 2\lambda\beta_2 = 0 \\ \vdots \\ 2X^T(X\beta_1 + X\beta_2 + \cdots + X\beta_m - Y) + 2\lambda\beta_m = 0 \end{cases} \Rightarrow -2\lambda\beta_1 = -2\lambda\beta_2 = \cdots = -2\lambda\beta_m$$

$$\Rightarrow \boxed{\beta_1 = \beta_2 = \cdots = \beta_m}$$

⇒ 대입해보면 $\beta_1(mX^TX + \lambda) = X^TY$

$$\Rightarrow \beta_1 = \beta_2 = \cdots = \beta_m = \frac{X^TY}{mX^TX + \lambda}$$