Data driven sampling (Lecture 6)

Regression

Hongwei YUAN

University of Macau

Model of regression

We first assume that the regression problem is one dimensional. We consider

$$Y = f(X) + \epsilon$$

where

- Y is called the output, X is the input, ϵ is noise term due to meaurement errors.
- f is unknown function to be learnt.
- We aim to learn f by observed data $(x_1, y_1), ..., (x_n, y_n)$.

Model of regression

We write

$$Y = f(X) + \epsilon$$

where

- f is unknown function to be learnt.
- In Statistics, one often assumes $f(x) = g(x, \theta)$ with g being known but θ unknown, e.g. $g(x, \theta) = \theta x$.
- When $f(x) = g(x, \theta)$, the problem turns to be finding θ by the observed data.
- The noise ϵ is often assumed to have a normal distribution $N(0, \sigma^2)$, usually the variance σ^2 is known.

Model of regression

We consider the regression:

$$Y = g(X, \theta) + \epsilon$$

where we have observed data $(x_1, y_1), ..., (x_n, y_n)$, and ϵ has the distribution $N(0, \sigma^2)$.

- for an observed data (x, y), the distribution is

$$p(x, y, \theta) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(y-g(x,\theta))^2}{2\sigma^2}} p(x),$$

  where $p(x)$ is the distribution that the data x satisfies.

- the likelihood function is

$$L_n(\theta) = \prod_{i=1}^{n} p(x_i, y_i, \theta) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^n \prod_{i=1}^{n} e^{-\frac{(y_i-g(x_i,\theta)^2}{2\sigma^2}} p(x_i).$$

Model of regression

- The log likelihood function is

$$l_n(\theta) = -\frac{1}{2}\sum_{i=1}^{n}(y_i - g(x_i, \theta))^2 + C + \sum_{i=1}^{n}\log p(x_i),$$

where $C = -n\log(\sqrt{2\pi}\sigma)$.

- Because $C + \sum_{i=1}^{n}\log p(x_i)$ does not depend on the θ, so

$$\begin{aligned}\arg\max_{\theta} l_n(\theta) &= \arg\max_{\theta}\left[-\frac{1}{2}\sum_{i=1}^{n}(y_i - g(x_i, \theta))^2\right] \\ &= \arg\min_{\theta}\left[\frac{1}{2}\sum_{i=1}^{n}(y_i - g(x_i, \theta))^2\right]\end{aligned}$$

Hence, the estimator of θ is

$$\hat{\theta}_n = \operatorname{argmin}_{\theta}\left[\frac{1}{2}\sum_{i=1}^{n}(y_i - g(x_i, \theta))^2\right].$$

$Y = g(X,\theta) + \varepsilon, \quad \varepsilon \sim N(0,\sigma^2)$

$p(x,y;\theta) = p(x)\cdot p(y|x;\theta) = p(x)\cdot \frac{1}{\sqrt{2\pi}\sigma}\cdot e^{-\frac{(y-g(x,\theta))^2}{2\sigma^2}}$

$Y|_{X=x} = g(x,\theta)+\varepsilon \sim N(g(x,\theta),\sigma^2)$

$L_n(\theta) = \prod_{i=1}^{n} p(x_i,y_i,\theta), \quad l_n(\theta) = \log L_n(\theta)$

$\hat{\theta}_n := \arg\max_\theta L_n(\theta) = \arg\max_\theta l_n(\theta)$

$= \arg\max_\theta -\frac{1}{2}\sum_{i=1}^{n}(y_i - g(x_i,\theta))^2 =: -\bar{l}_n(\theta)$

$\bar{l}_n'(\theta) = 0 \Rightarrow \hat{\theta}_n = ?$

Example. $g(x,\theta) = \theta_1 x + \theta_0$

$\bar{l}_n(\theta_1,\theta_0) = -\frac{1}{2}\sum_{i=1}^{n}(y_i - \theta_1 x_i - \theta_0)^2$

$0 = \frac{\partial}{\partial\theta_1}\bar{l}_n = \sum_{i=1}^{n} x_i y_i - \theta_1 x_i^2 - \theta_0 x_i = n\cdot\overline{xy} - n\cdot\theta_1\overline{x^2} - n\cdot\theta_0\bar{x}$

$0 = \frac{\partial}{\partial\theta_0}\bar{l}_n = \sum_{i=1}^{n}(y_i - \theta_1 x_i - \theta_0)$

$= n\cdot\bar{y} - \theta_1\cdot n\cdot\bar{x} - n\cdot\theta_0$

$\Rightarrow \begin{cases} \overline{xy} - \overline{x^2}\cdot\theta_1 - \bar{x}\cdot\theta_0 = 0 \cdots ① \\ \bar{y} - \bar{x}\cdot\theta_1 - \theta_0 = 0 \cdots ② \end{cases}$

$① - ②\cdot\bar{x} \Rightarrow \overline{xy} - \overline{x^2}\cdot\theta_1 - \bar{y}\cdot\bar{x} + \bar{x}^2\theta_1 = 0$

$\Rightarrow \theta_1 = \boxed{\frac{\overline{xy} - \bar{x}\cdot\bar{y}}{\overline{x^2} - \bar{x}^2}}$

$\theta_0 = \bar{y} - \bar{x}\cdot\theta_1 = \bar{y} - \bar{x}\cdot$

One dimensional linear regression

- Suppose $g(x, \theta) = \theta_1 x + \theta_0$ with $\theta_1 \in \mathbb{R}$ and $\theta_0 \in \mathbb{R}$ being unknown parameters to be estimated, therefore,

$$\hat{\theta}_n = \arg \min_{\theta=(\theta_0,\theta_1)\in\mathbb{R}^2} \left[\frac{1}{2} \sum_{i=1}^{n} (y_i - \theta_1 x_i - \theta_0)^2 \right]. \tag{1}$$

- (Class Exercise) Show that $\hat{\theta}_n = (\hat{\theta}_{n,1}, \hat{\theta}_{n,0})$ with the form:

$$\hat{\theta}_{n,1} = \frac{\overline{xy} - \bar{x}\bar{y}}{\overline{x^2} - (\bar{x})^2},$$

$$\hat{\theta}_{n,0} = \bar{y} - \left(\frac{\overline{xy} - \bar{x}\bar{y}}{\overline{x^2} - (\bar{x})^2} \right) \bar{x},$$

where $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$, $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$, $\overline{xy} = \frac{1}{n}\sum_{i=1}^{n} x_i y_i$, $\overline{x^2} = \frac{1}{n}\sum_{i=1}^{n} x_i^2$.

One dimensional polynomial regression

Suppose here is K dimensional

Suppose $g(x,\theta) = \theta_K x^K + ... + \theta_1 x + \theta_0$ with $\theta_K \in \mathbb{R}$,..., $\theta_1 \in \mathbb{R}$ and $\theta_0 \in \mathbb{R}$ being unknown parameters to be estimated, therefore,

$$\hat{\theta} = \arg \min_{\underbrace{\theta=(\theta_0,...,\theta_K)\in\mathbb{R}^{K+1}}_{\theta\in\mathbb{R}^{K+1}}} \left[\frac{1}{2} \sum_{i=1}^{n} (y_i - \theta_K x_i^K - ... - \theta_1 x_i - \theta_0)^2 \right]. \quad (2)$$

Denote $f_n(\theta_0, ..., \theta_K) = \frac{1}{2} \sum_{i=1}^{n} (y_i - \theta_K x_i^K - ... - \theta_1 x_i - \theta_0)^2$, to find the minimizer of f_n, we differentiate it with respect to $\theta_0, ..., \theta_K$ and let these partial derivatives be 0, i.e.

$$\frac{\partial f_n(\theta)}{\partial \theta_k} = 0, \quad k = 0, ..., K,$$

which read as

$$\sum_{i=1}^{n} (y_i - \theta_K x_i^K - ... - \theta_1 x_i - \theta_0) x_i^k = 0, \quad k = 0, ..., K. \quad (3)$$

One dimensional polynomial regression

Write

$$\mathbf{X} = \begin{bmatrix} 1 & x_1 & x_1^2 & ... & x_1^K \\ 1 & x_2 & x_2^2 & ... & x_2^K \\ ... & ... & ... & ... & \\ 1 & x_n & x_n^2 & ... & x_n^K \end{bmatrix}$$

Assignment 1: Verify that (3) can be rewritten as a vector equation as

$$\mathbf{X}^\top \mathbf{y} = \mathbf{X}^\top \mathbf{X}\theta.$$

where $\mathbf{y} = [y_1, ..., y_n]^\top$ and $\theta = [\theta_0, ..., \theta_K]^\top$.

One dimensional polynomial regression

$\exists$ invertible matrix P and $D = \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_{K+1} \end{pmatrix}$ s.t. $X^TX = P^{-1}DP$ $\lambda_1, \dots, \lambda_{K+1}$ are eigenvalues of X^TX

X^TX is real symmetric $\Leftrightarrow \lambda_i \neq 0$, for all i

If the matrix $\mathbf{X}^\top\mathbf{X} \in \mathbb{R}^{(K+1)\times(K+1)}$ is invertible, then

$$\hat{\theta} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}.$$

if $X^TX \in \mathbb{R}^{(K+1)\times(K+1)}$ is not invertible

Choose $\lambda \neq \lambda_i$

$X^Ty - \lambda\theta = (X^TX - \lambda I)\theta = P^{-1}(D-\lambda I)P\theta$

$$\Downarrow$$

$$\begin{pmatrix} \lambda_1-\lambda & & 0 \\ & \ddots & \\ 0 & & \lambda_K-\lambda \end{pmatrix}$$

$\Rightarrow \theta_{j+1} = (X^TX - \lambda I)^{-1}(X^Ty - \lambda\theta_j)$

do iteration

$\theta_0, \theta_1, \theta_2, \dots, \theta_j, \dots$

$\rightarrow \theta^*, X^Ty = X^TX\theta^*$

$$\sum_{j=1}^{n} h_{kj}h_{mj}x_j^2 = \frac{h_{kj}}{|h_{kj}|}\cdot\frac{h_{mj}}{|h_{mj}|}\cdot x_j^2\cdot\frac{1}{\sqrt{n}}\cdot\frac{1}{\sqrt{n}}$$

$$= \frac{x_j^2}{n}\cdot\frac{h_{kj}}{|h_{kj}|}\cdot\frac{h_{mj}}{|h_{mj}|}$$

Multi-d LR.

d is the dimension of the unknown θ.
n is number of samples. $d > n$

$$y_i = \theta_0 + x_{i1}\theta_1 + \cdots + x_{id}\theta_d + \varepsilon_i, \quad i = 1, \cdots, n.$$

$$\min_\theta \|y - X\theta\|^2 \Rightarrow X^TX\hat{\theta} = X^Ty \quad (*)$$

If X^TX is invertible, $\hat{\theta} = (X^TX)^{-1}X^Ty$

If X^TX is not invertible, try $X \in \mathbb{R}^{n \times (d+1)}$

$$\theta_{j+1} = (X^TX + \lambda I)^{-1}(X^Ty + \lambda\theta_j)$$

error comparison to ridge regression

If $\lim_{j\to\infty} \theta_j = \hat{\theta}$ exists, then $\hat{\theta}$ solves $(*)$.

$X^TX = PDP^T$, $P^T = P^{-1}$ ($(d+1)\times(d+1)$)

$$D = \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_{d+1} \end{pmatrix}$$

$\theta_0, \theta_1, \cdots, \theta_j, \cdots$
If $\exists\, \delta \in (0,1)$ s.t.
$\|\theta_{j+1} - \theta_j\| \le \delta \cdot \|\theta_j - \theta_{j-1}\|$, for all j,
then $\lim_{j\to\infty} \theta_j$ exists.

$$\theta_{j+1} - \theta_j = (X^TX + \lambda I)^{-1}\lambda(\theta_j - \theta_{j-1})$$
$$= P(D + \lambda I)^{-1}P^{-1}\lambda(\theta_j - \theta_{j-1})$$
$$\Rightarrow P^{-1}(\theta_{j+1} - \theta_j) = (D + \lambda I)^{-1}\lambda\, P^{-1}(\theta_j - \theta_{j-1})$$

$\exists\, \delta \in (0,1)$ $\|P^{-1}(\theta_{j+1} - \theta_j)\| \le \delta \|P^{-1}(\theta_j - \theta_{j-1})\|$?

$$(D + \lambda I)^{-1}\lambda = \begin{pmatrix} \frac{\lambda}{\lambda+\lambda_1} & & 0 \\ & \ddots & \\ 0 & & \frac{\lambda}{\lambda+\lambda_{d+1}} \end{pmatrix}$$

$\left|\frac{\lambda}{\lambda+\lambda_i}\right| \le \delta$, for all i

this fails since there are $\lambda_i = 0$.

Multi-dimensional linear regression

Many problems in Statistics can be recast as a multi-dimensional linear regression as the following: given a sequence of observed data

$$(\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n),$$

where

- $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$ for all i,
- one would like to learn from these given data the changing tendency of y_i according to $\mathbf{x}_i$,
- For instance, $\mathbf{x}_i = [x_{i1}, x_{i2}]$ is a vector of height and weight of i-th person and y_i is his blood pressure.

Linear regression: an optimization problem

- Linear regression uses a linear function to model the relation between $\mathbf{x}_i$ and y_i, i.e., we aim to find some $\theta = (\theta_0, \theta_1, ..., \theta_d)' \in \mathbb{R}^{d+1}$ such that

$$y_i = \theta_0 + x_{i1}\theta_1 + ... + x_{id}\theta_d + \epsilon_i, \quad i = 1, ..., n,$$

where ϵ_i is the noise (measurement error), it can be written as

$$\mathbf{y} = \mathbf{X}\theta + \epsilon \tag{4}$$

where $\epsilon = [\epsilon_1, ..., \epsilon_n]^\top$, $\mathbf{X} = [\mathbf{1}, [\mathbf{x}_1, ..., \mathbf{x_n}]^\top] \in \mathbb{R}^{n\times(d+1)}$ with $\mathbf{1} = [1, ...1]^\top \in \mathbb{R}^n$.

n data inputs

$\vec{x_i} \in \mathbb{R}^d$ for $i = 1, 2, \cdots, n$

- The least square (LE) regression is to find the 'best' θ for the equation (4).

Least square regression

- The linear equation (4) can be written as

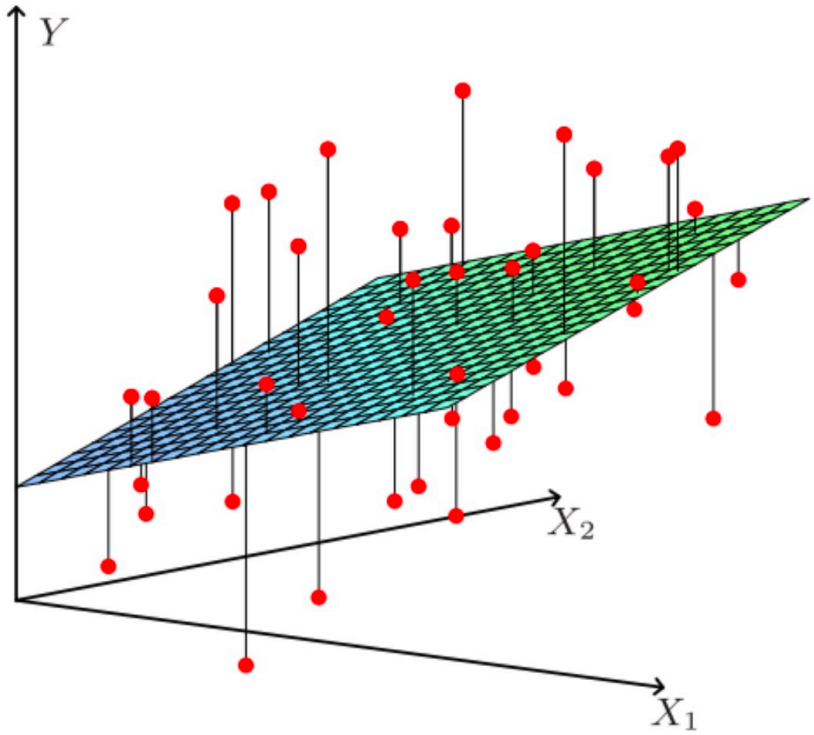
$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \left[\begin{array}{c|ccc} 1 & x_{11} & x_{12} & \dots & x_{1d} \\ 1 & x_{21} & x_{22} & \dots & x_{2d} \\ \dots & \dots & \dots & \dots & \\ 1 & x_{n1} & x_{n2} & \dots & x_{nd} \end{array}\right] \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_d \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

- Inspired by (2), LS regression aims to minimize the following question

$$\min_{\theta} \|\mathbf{y} - \mathbf{X}\theta\|^2 = \min_{\theta} \sum_{i=1}^{n} (y_i - \theta_0 - x_{i1}\theta_1 - \dots - x_{id}\theta_d)^2$$

where $\|.\|$ is the Euclidean distance in $\mathbb{R}^n$, i.e., for all $x \in \mathbb{R}^n$, $\|x\| = \sqrt{x_1^2 + \dots + x_n^2}$.

$$f(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$



Least square regression

- How to solve this problem? By differentiate $\|\mathbf{y} - \mathbf{X}\theta\|^2$ with respect to θ, i.e. for $j = 0, 1, ..., d$,

$$\frac{\partial}{\partial \theta_j} \|\mathbf{y} - \mathbf{X}\theta\|^2 = 0.$$

  which leads to

$$\mathbf{X}^\top \mathbf{y} = \mathbf{X}^\top \mathbf{X}\theta.$$

- As long as $\mathbf{X}^\top \mathbf{X}$ invertible, we get

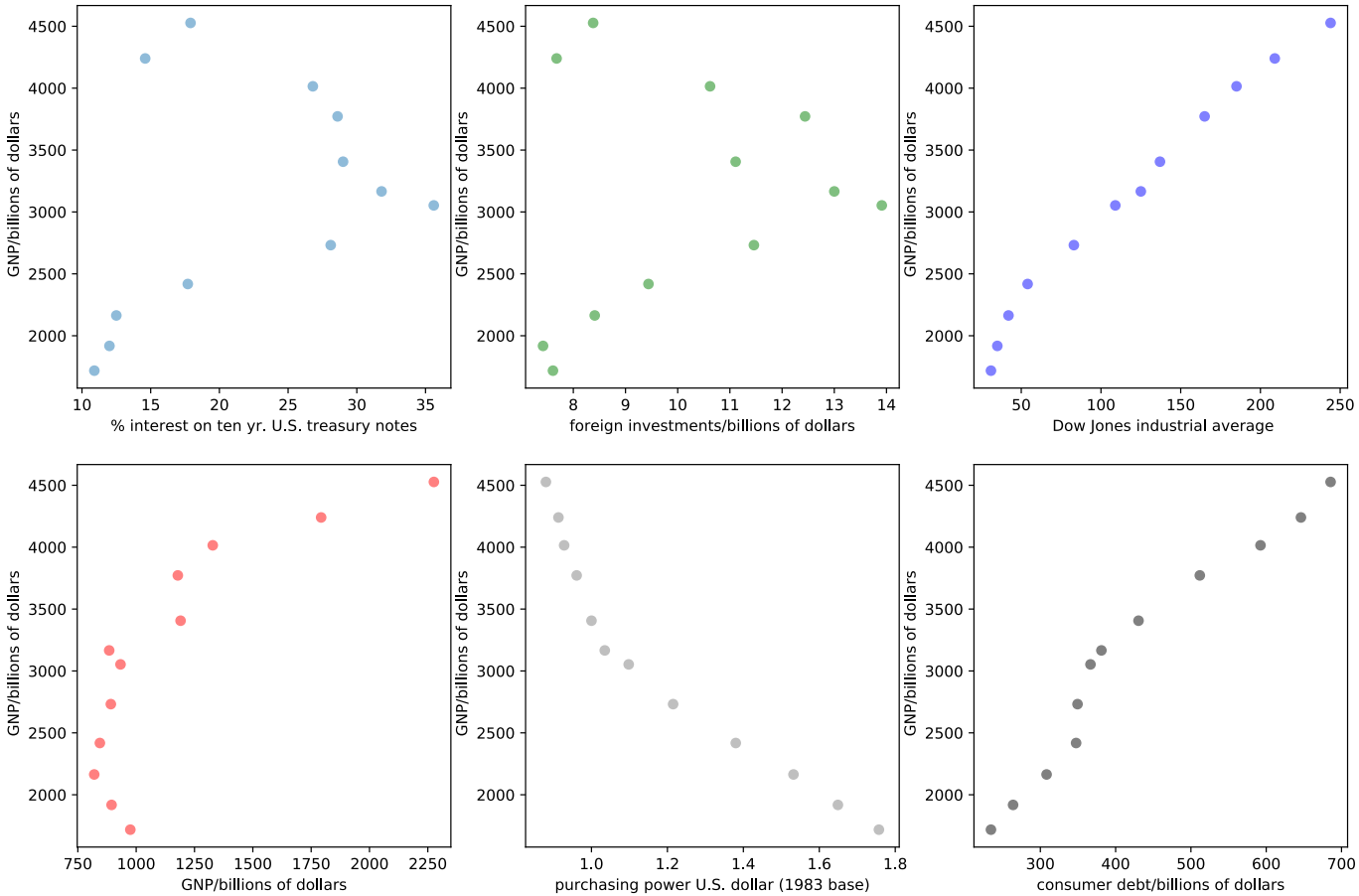
$$\hat{\theta} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}.$$

Example 1: US Gross National Product (GNP)

U.S. Economy Case Study U.S.. economic data 1976 to 1987

- Y= GNP/billions of dollars
- X1 = dollars/barrel crude oil
- X2 = % interest on ten yr. U.S. treasury notes
- X3= foreign investments/billions of dollars
- X4 = Dow Jones industrial average
- X5 = purchasing power U.S. dollar (1983 base)
- X6 = consumer debt/billions of dollars

Example 1: US Gross National Product (GNP)



$$Y=4.9*X1+1.1*X2+5.1*X3-1.1*'X4-8.7*X5+2.1*X6+2531.6.$$

Example 1: US Gross National Product (GNP)

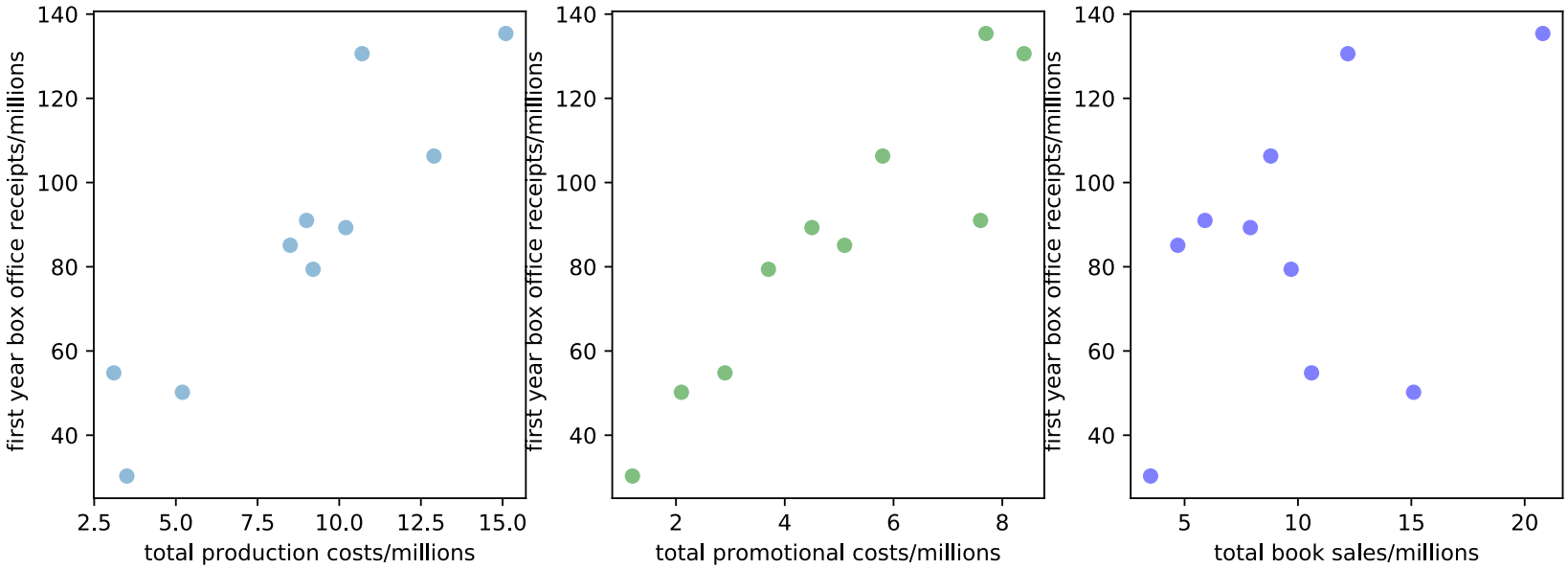
GNP=4.9*'CRUDE'+1.1*'INTEREST'+5.1*'FOREIGN'-1.1*'DJIA'

-8.7*'PURCHASE'+2.1*'CONSUMER'+2531.6

Example 2: Hollywood Movies Box Office Receipts

- Y = first year box office receipts/millions,
- X1 = total production costs/millions,
- X2 = total promotional costs/millions,
- X3 = total book sales/millions

Example 2: Hollywood Movies Box Office Receipts

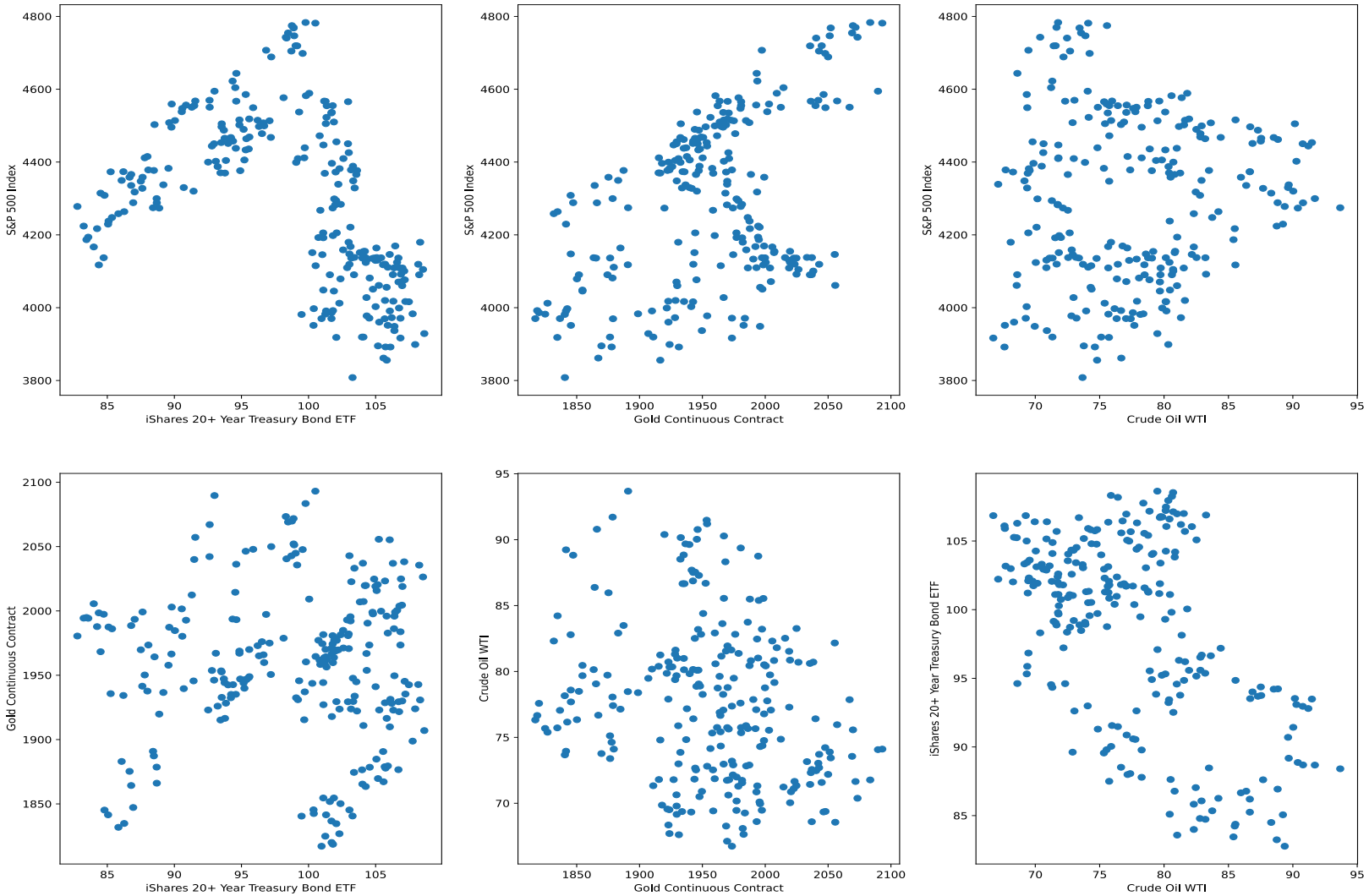


Y=3.7*X1+7.6*X2+0.8*X3+7.7.

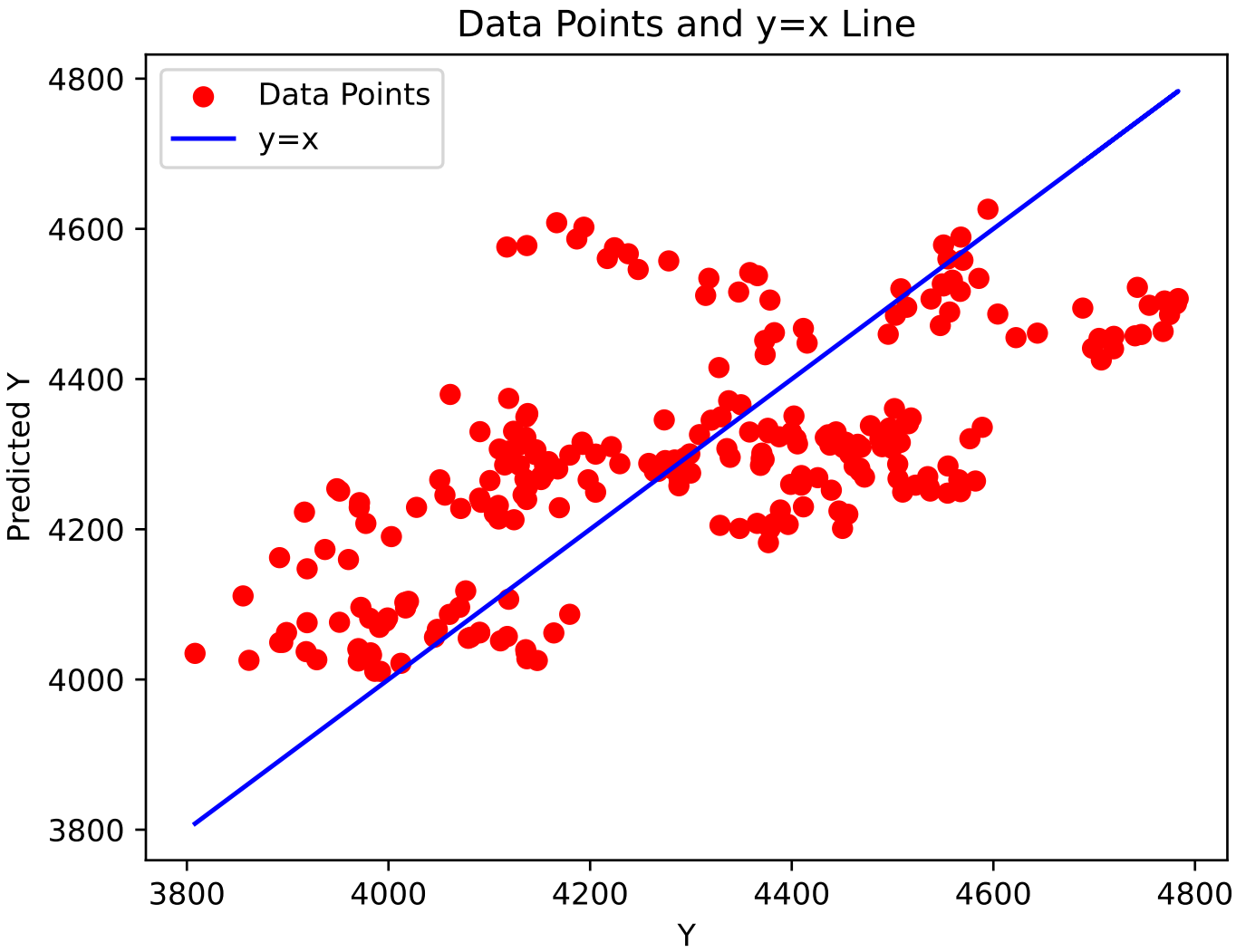
Example 3: S&P 500 Index

- Y = S&P 500 Index,
- X1 =iShares 20+ Year Treasury Bond ETF,
- X2 = Gold Continuous Contract,
- X3 = Crude Oil WTI

Example 3: S&P 500 Index



Example 3: S&P 500 Index

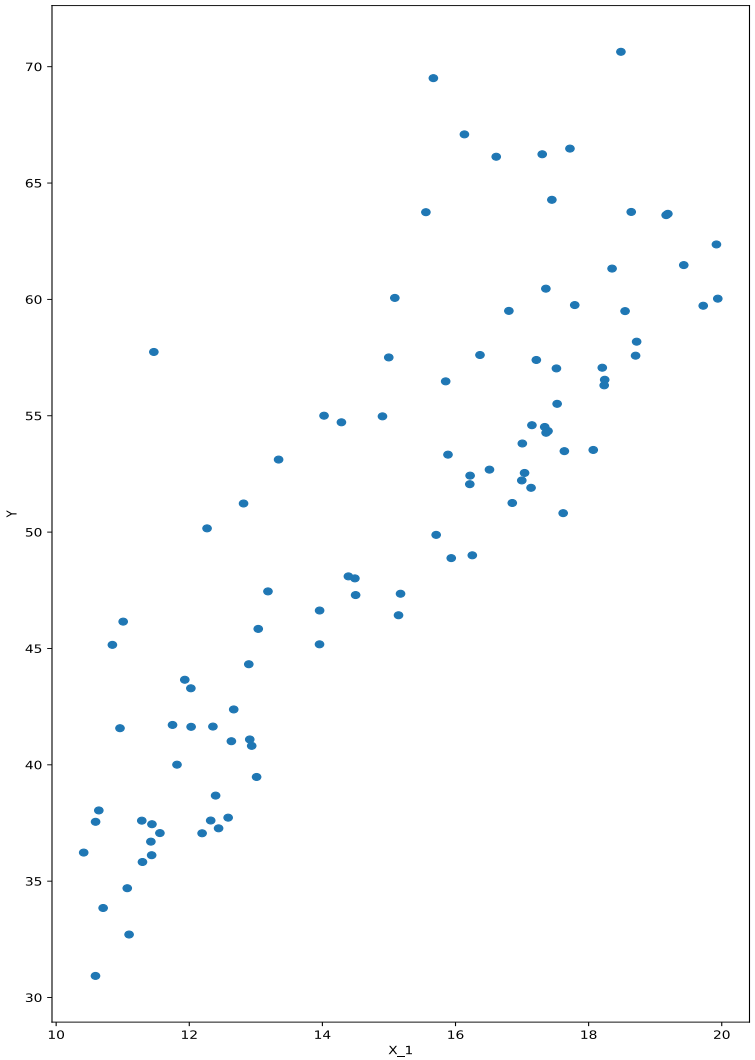


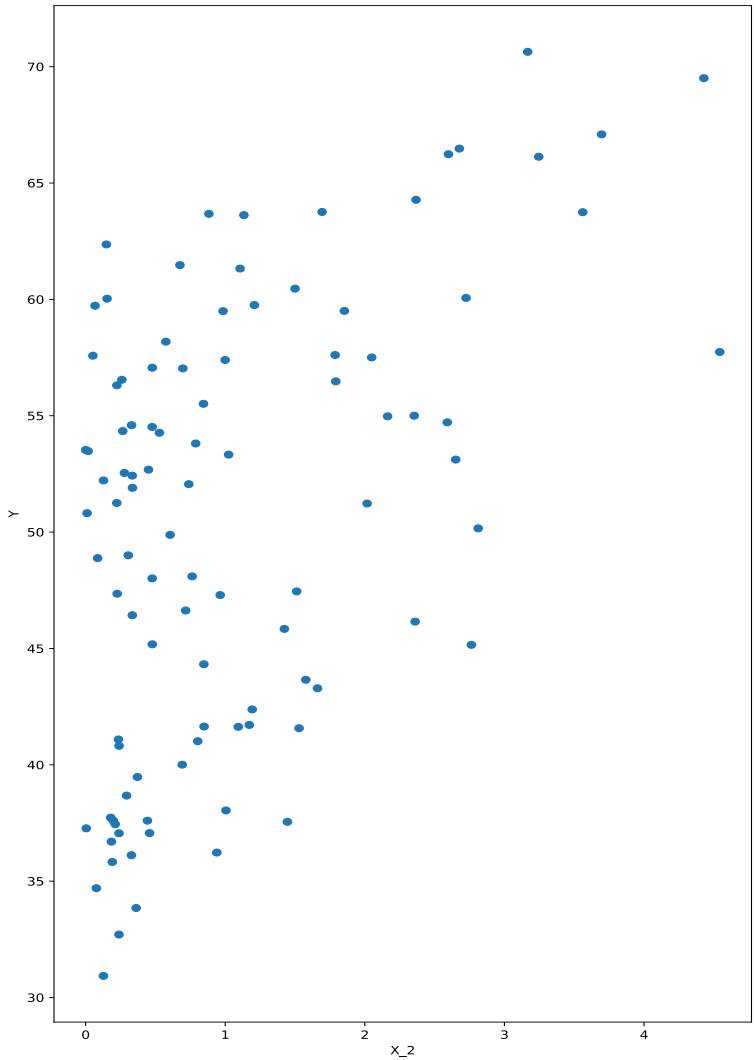
Y=-17.47*X1+1.66*X2-4.33*X3+33104.34.

Example 4: Simulation

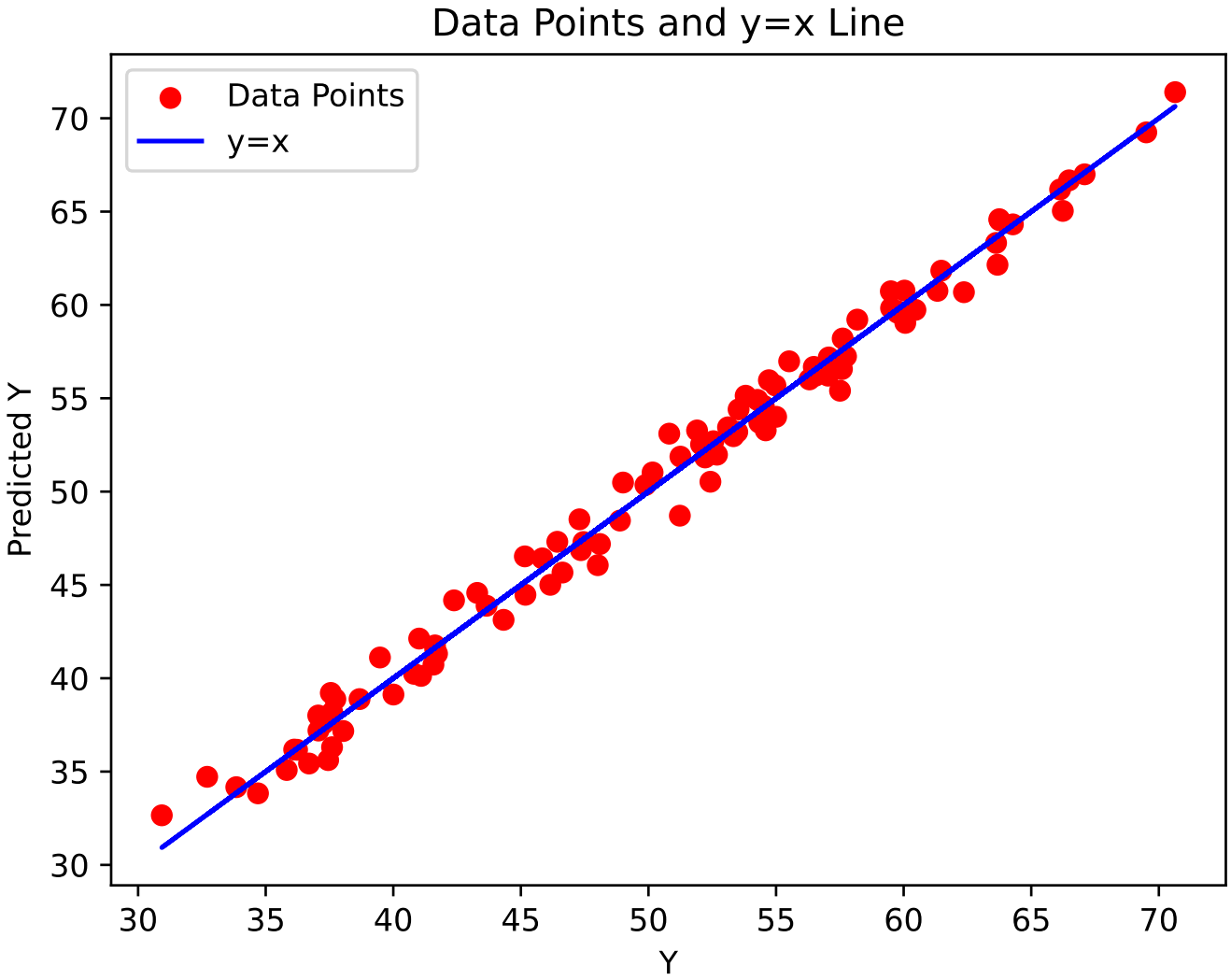
- Y = 3*X1+5*X2+ϵ,
- X1 =Uniform(10,20),
- X2 = Exp(1),
- ϵ = N(0,1)

Example 4: Simulation





Example 4: S&P 500 Index



Y=2.99*X1+4.97*X2+0.32.

Python codes for example 1

```
# U.S.. economic data 1976 to 1987
# X1 = dollars/barrel crude oil
# X2 = % interest on ten yr. U.S. treasury notes
# X3 = foreign investments/billions of dollars
# X4 = Dow Jones industrial average
# X5 = GNP/billions of dollars
# X6 = purchasing power U.S. dollar (1983 base)
# X7 = consumer debt/billions of dollars
data = pd.read_csv("U.S._Economy.csv")
X = data[['CRUDE','INTEREST','FOREIGN','DJIA','PURCHASE','CONSUMER']]
y = data['GNP']
fig = plt.figure(1,figsize=[15.,10.])
ax = plt.subplot(231)
plt.scatter(X['CRUDE'],y,alpha=0.5)
plt.xlabel('% interest on ten yr. U.S. treasury notes')
plt.ylabel(' GNP/billions of dollars')
plt.subplot(232)
plt.scatter(X['INTEREST'],y,c='green',alpha=0.5)
plt.xlabel('foreign investments/billions of dollars')
plt.ylabel(' GNP/billions of dollars')
plt.subplot(233)
plt.scatter(X['FOREIGN'],y,c='blue',alpha=0.5)
plt.xlabel(' Dow Jones industrial average')
plt.ylabel(' GNP/billions of dollars')
plt.subplot(234)
plt.scatter(X['DJIA'],y,c='red',alpha=0.5)
plt.xlabel('GNP/billions of dollars')
plt.ylabel(' GNP/billions of dollars')
plt.subplot(235)
plt.scatter(X['PURCHASE'],y,c='grey',alpha=0.5)
plt.xlabel('purchasing power U.S. dollar (1983 base)')
plt.ylabel(' GNP/billions of dollars')
plt.subplot(236)
plt.scatter(X['CONSUMER'],y,c='black',alpha=0.5)
plt.xlabel('consumer debt/billions of dollars')
plt.ylabel(' GNP/billions of dollars')
plt.savefig("U.S._Economy scatter plots.pdf")
plt.show()
regr10 = LinearRegression()
regr10.fit(X,y)
predicts = regr10.predict(X) #predicted values
coef10= regr10.coef_
intercept10 = regr10.intercept_
print(coef10, intercept10)
```

Python codes for example 2

```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
# Hollywood Movies data,
# X1 = first year box office receipts/millions,
# X2 = total production costs/millions,
# X3 = total promotional costs/millions,
# X4 = total book sales/millions
data = pd.read_csv("Hollywood_Movies.csv")
X = data[['X2','X3','X4']]
y = data['X1']
#plot the scatters
fig = plt.figure(1,figsize=[12.,4.])
ax = plt.subplot(131)
plt.scatter(X['X2'],y,alpha=0.5)
plt.xlabel('total production costs/millions')
plt.ylabel(' first year box office receipts/millions')
plt.subplot(132)
plt.scatter(X['X3'],y,c='green',alpha=0.5)
plt.xlabel('total promotional costs/millions')
plt.ylabel(' first year box office receipts/millions')
plt.subplot(133)
plt.scatter(X['X4'],y,c='blue',alpha=0.5)
plt.xlabel('total book sales/millions')
plt.ylabel(' first year box office receipts/millions')
plt.savefig("Hollywood_Movies scatter plots.pdf")
plt.show()
# build regression model
regr4 = LinearRegression()
regr4.fit(X,y)
predicts = regr4.predict(X) #predicted values
coef4 = regr4.coef_
intercept4 = regr4.intercept_
print(coef4, intercept4)
```

High Dimensional Linear regression

- Recall the linear regression:

$$y_i = \theta_0 + x_{i1}\theta_1 + ... + x_{id}\theta_d + \epsilon_i, \quad i = 1, ..., n,$$

where ϵ_i is the noise (measurement error), it can be written as

$$\mathbf{y} = \mathbf{X}\theta + \epsilon. \tag{5}$$

- As long as $\mathbf{X}^\top\mathbf{X}$ invertible, we get $\hat{\theta} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}$.
- In high dimensional regression problem, we often have the following case:

$$d > n,$$

i.e. the dimension of parameter is larger than the sample size. Then $\mathbf{X}^\top\mathbf{X}$ is NOT invertible.

Ridge Linear Regression

- We consider the linear regression:

$$\mathbf{y} = \mathbf{X}\theta + \epsilon, \tag{6}$$

with the restriction: for some $K > 0$, $\|\theta\| \leq K$.

- By Lagrangian duality, to estimate the parameter θ, we solve the following optimization problem:

$$\min_{\theta} \left[\|\mathbf{y} - \mathbf{X}\theta\|^2 + \lambda \|\theta\|^2 \right], \tag{7}$$

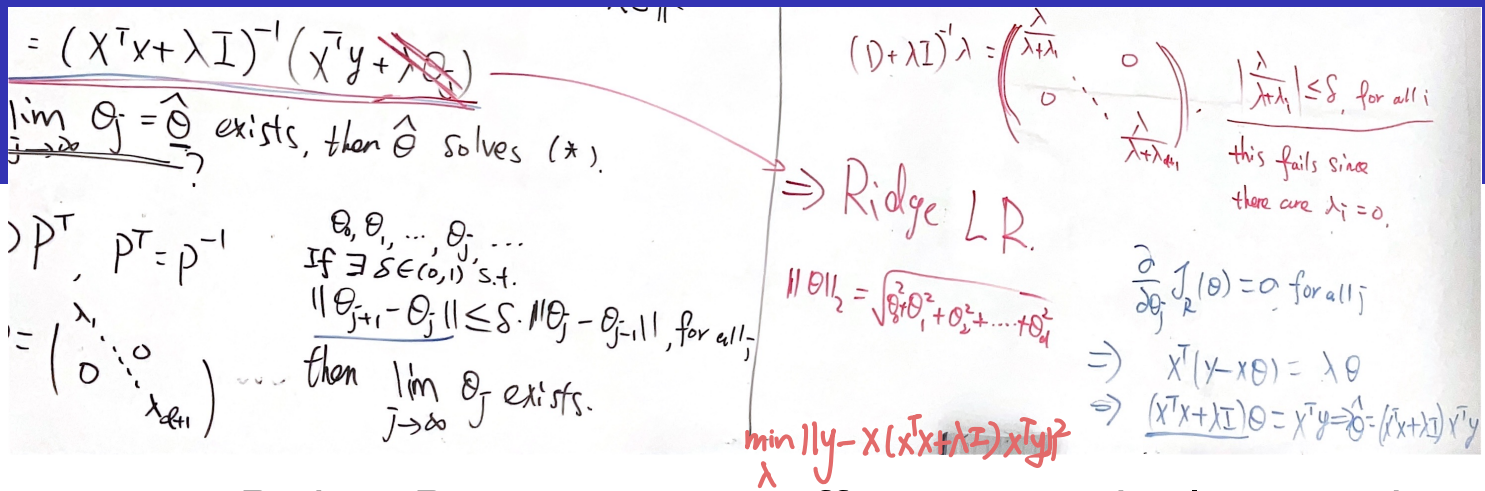
where $\lambda > 0$ is some tuning parameter to be chosen.

- Assignment 4: For (7), we have

$$\hat{\theta} = (\lambda I_d + \mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}, \tag{8}$$

where I_d is the $d \times d$ identity matrix.

Ridge Regression



Advantages:

1. Handles Multicollinearity: Ridge Regression is effective in dealing with multicollinearity, which occurs when predictors are highly correlated. It reduces the impact of correlated predictors by shrinking their coefficients without eliminating them entirely.

2. Stable Solution: Ridge Regression provides a more stable solution compared to Lasso Regression, as it does not force coefficients to be exactly zero. This makes it more suitable when all predictors are potentially relevant.

$\min_{\lambda} \| y - X\cdot(X^TX+\lambda I)X^Ty \|^2$ for $\alpha \in (0,1)$, suitable λ.

$\theta_{j+1} = (X^TX+\lambda I)^{-1}(X^Ty + \alpha\lambda\theta_j) \Rightarrow \lim_{j\to\infty} \theta_j$ exists.

$\|P^{-1}(\theta_{j+1}-\theta_j)\| = \left\| \begin{pmatrix} \frac{\alpha\lambda}{\lambda+\lambda_1} & & 0 \\ & \ddots & \\ 0 & & \frac{\alpha\lambda}{\lambda+\lambda_{d+1}} \end{pmatrix} P^{-1}(\theta_j-\theta_{j-1}) \right\|$

$\le \delta \|P^{-1}(\theta_j-\theta_{j-1})\|$

$\alpha=0 \Rightarrow$ Ridge L.R.

$\alpha=1 \Rightarrow$ original L.R.

$\alpha\in(0,1)$?, $\theta=(X^TX+\lambda I)^{-1}(X^Ty+\alpha\lambda\theta)$

$\Rightarrow$ Ridge LR.

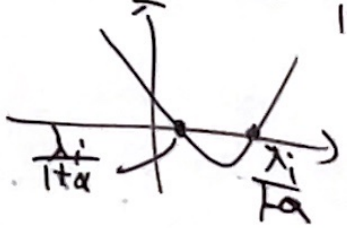
$\|\theta\|_2 = \sqrt{\theta_0^2+\theta_1^2+\theta_2^2+\cdots+\theta_d^2}$

$\left|\frac{\alpha\lambda}{\lambda+\lambda_i}\right| < 1$

$\Leftrightarrow \alpha^2\lambda^2 < \lambda^2+2\lambda_i\lambda+\lambda_i^2$

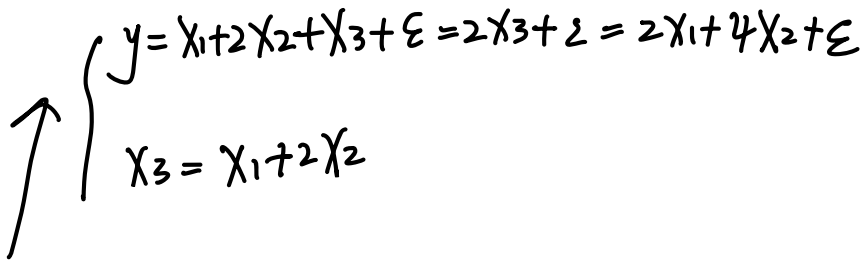
$\Leftrightarrow (1-\alpha^2)\lambda^2+2\lambda_i\lambda+\lambda_i^2 > 0$

$= (1-\alpha^2)\cdot\left(\lambda+\frac{\lambda_i}{1+\alpha}\right)\cdot\left(\lambda+\frac{\lambda_i}{1-\alpha}\right) > 0$



$\lambda > \frac{\lambda_i}{1-\alpha}$ or $\lambda < \frac{\lambda_i}{1+\alpha}$

Ridge Regression



Disadvantages:

1. Does Not Perform Feature Selection: Unlike Lasso Regression, Ridge Regression does not perform feature selection. It shrinks the coefficients of all predictors but does not force any of them to become exactly zero. Therefore, it does not provide a sparse model with selected predictors.

2. Less Interpretable: Due to the shrinkage effect, Ridge Regression may not provide a readily interpretable model, as the coefficients are not easily distinguishable between important and unimportant predictors.

Lasso Linear Regression

The least absolute shrinkage and selection operator (lasso) was put forward by Tibshirani (1996).

- We consider the linear regression:

$$\mathbf{y} = \mathbf{X}\theta + \epsilon, \tag{9}$$

with the restriction: for some $K > 0$, $\sum_{i=0}^{d} |\theta_i| \leq K$.

- By Lagrangian duality, to estimate the parameter θ, we solve the following optimization problem:

$$\min_{\theta} \left[\|\mathbf{y} - \mathbf{X}\theta\|^2 + \lambda \sum_{i=0}^{d} |\theta_i| \right],$$

where $\lambda > 0$ is some tuning parameter to be chosen.

Lasso Regression

Advantages:

1. Feature Selection: Lasso Regression performs feature selection by driving some coefficients to exactly zero. This makes it useful when dealing with high-dimensional datasets with many irrelevant features, as it helps in identifying the most important predictors.

2. Simplicity: Lasso Regression provides a simple and interpretable model by shrinking less important features to zero, effectively eliminating them from the model.

Lasso Regression

Disadvantages:

1. Unstable Solution: Lasso Regression can be sensitive to the presence of highly correlated predictors. In the presence of multicollinearity, it tends to arbitrarily select one variable over another, leading to an unstable solution.

2. Biased Estimates: Lasso Regression tends to introduce bias in coefficient estimates, particularly when dealing with small sample sizes or when there is a high degree of multicollinearity among predictors.

Lasso Regression v.s. Ridge Regression

In summary, Lasso Regression offers feature selection and simplicity at the cost of potential instability and bias. On the other hand, Ridge Regression handles multicollinearity and provides a stable solution but lacks feature selection capability and may be less interpretable. The choice between the two techniques depends on the specific requirements of the problem and the nature of the dataset.

Lasso Regression v.s. Ridge Regression

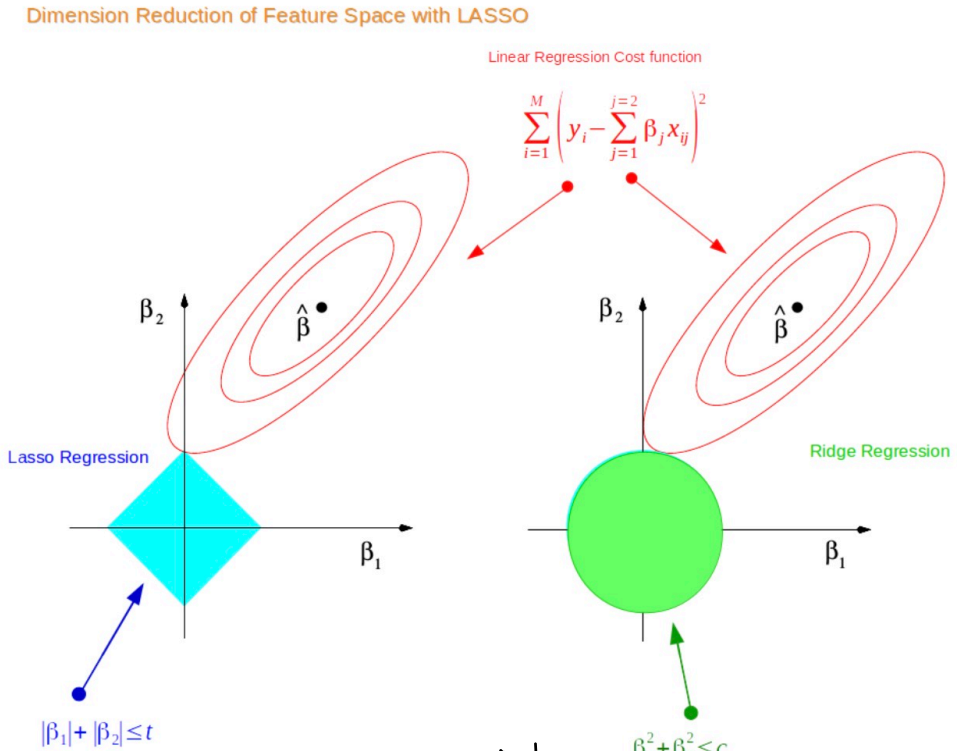
$\|\theta\|_1 = \sum_{i=1}^{d} |\theta_i|$, $\|\theta\|_2 = \sqrt{\sum_{i=1}^{d} |\theta_i|^2}$

↑ Lasso L.R. ↑ Ridge L.R.

Ridge: $\min_\theta \left(\underbrace{\|y - X\theta\|_2^2}_{=MSE} + \lambda \|\theta\|_2^2 \right) = J(\theta)$

Lasso: $\min_\theta \|y - X\theta\|_2^2 + \lambda \|\theta\|_1$

Geometric interpretation of Lasso and ridge regressions:



Ridge L.R.

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

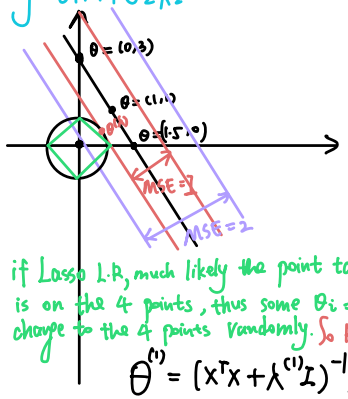
step 2: $\min_\lambda \|y - X\hat{\theta}(\lambda)\|_2$

X_1 and X_2

one case: linear dependence

$\begin{cases} y = X_1 + X_2 = 3X_2 = 1 \cdot X_1 \\ X_1 = 2X_2 \end{cases}$

$y = \theta_1 x_1 + \theta_2 x_2$



if Lasso L.R, much likely the point tangent is on the 4 points, thus some $\theta_i = 0$, and may change to the 4 points randomly. So Ridge L.R. is better for continuous changing tangent point.

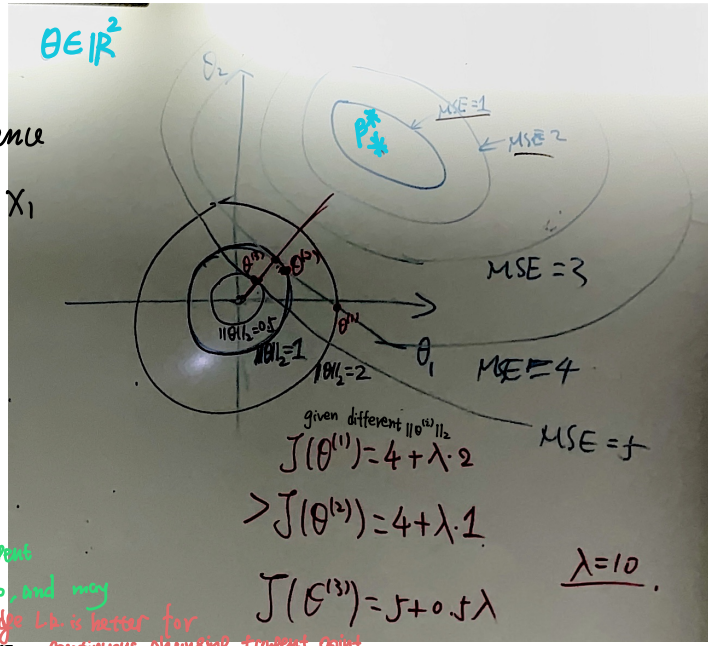
$\theta^{(1)} = (X^T X + \lambda^{(1)} I)^{-1} X^T y$

$\theta^{(2)} = (X^T X + \lambda^{(2)} I)^{-1} X^T y$

if $\lambda^{(2)} > \lambda^{(1)}$

then $\|\theta^{(2)}\|_2 < \|\theta^{(1)}\|_2$

$\theta \in \mathbb{R}^2$



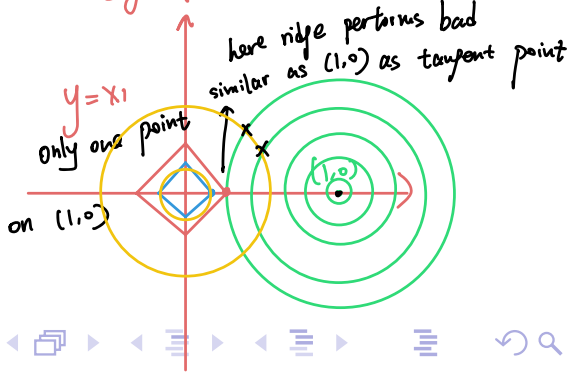
given different $\|\theta^{(i)}\|_2$

$J(\theta^{(1)}) = 4 + \lambda \cdot 2$

$> J(\theta^{(2)}) = 4 + \lambda \cdot 1$

$J(\theta^{(3)}) = 5 + 0 \cdot \lambda$

$\lambda = 10$.



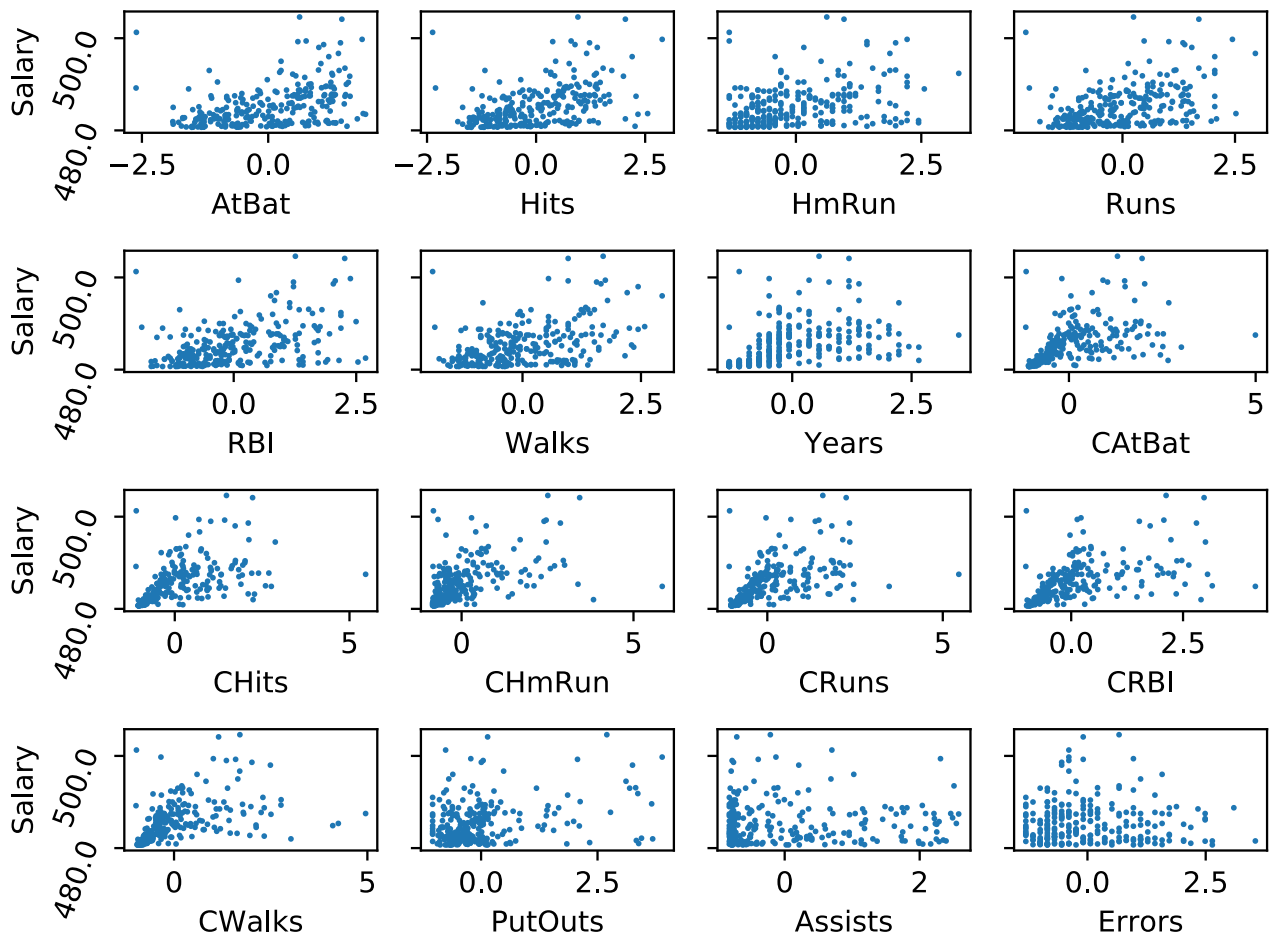
Example 3: Major League Baseball Data from the 1986 and 1987 seasons

- Y = Salary 1987 annual salary on opening day in thousands of dollars,
- X1 = Number of times at bat in 1986 (AtBat),
- X2 = Number of hits in 1986 (Hits),
- X3 = Number of home runs in 1986 (HmRun),
- X4 = Number of runs in 1986 (Runs),
- X5 = Number of runs batted in 1986 (RBI),
- X6 = Number of walks in 1986 (Walks),
- X7 = Number of years in the major leagues (Years),

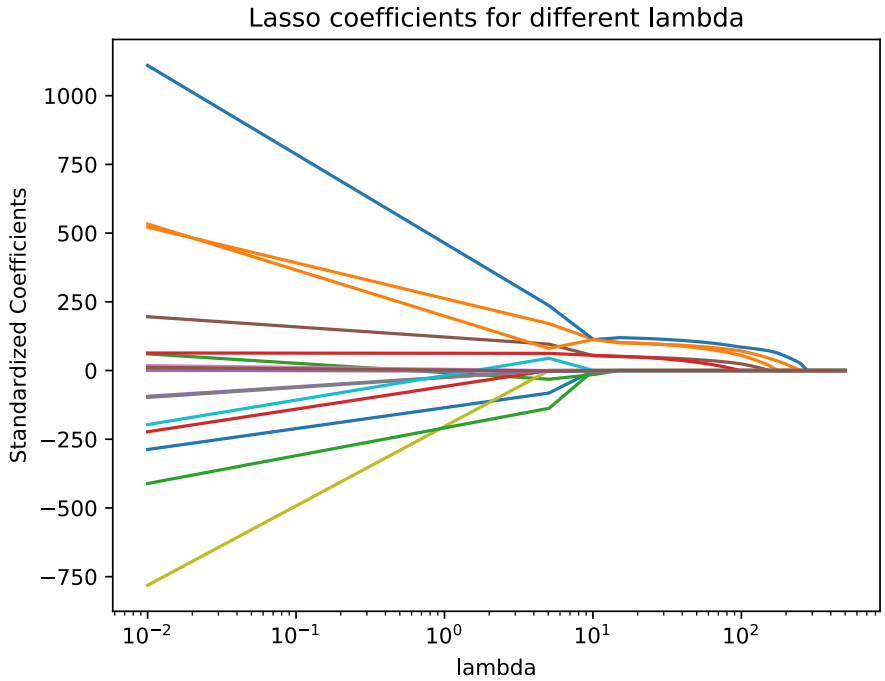
Example 3: Major League Baseball Data from the 1986 and 1987 seasons

- X8 = Number of times at bat during his career (CAtBat),
- X9 = Number of hits during his career (CHits),
- X10 = Number of home runs during his career (CHmRun),
- X11 = Number of runs during his career (CRuns),
- X12 = Number of runs batted in during his career (CRBI),
- X13 = Number of walks during his career (CWalks),
- X14 = Number of put outs in 1986 (PutOuts),
- X15 = Number of assists in 1986 (Assists),
- X16 = Number of errors in 1986 (Errors).

Example 3: Major League Baseball Data from the 1986 and 1987 seasons



Example 3: Major League Baseball Data from the 1986 and 1987 seasons



- Tuning parameters λ need to be carefully chosen.
- By 5-fold cross-validation, the optimal tuning parameter λ is 24.15.
- Y=97.48*X2+47.78*X6+115.97*X11+97.28*X12+46.16*X14.

Python codes for example 3

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import Lasso
from sklearn.metrics import mean_squared_error
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LassoCV
df = pd.read_csv('Hitters.csv')
# data processing and delete the missing values
df = df.dropna()
# Convert the categorical variables to dummy variables
y = df['Salary']
X = df.drop(['Salary', 'League', 'Division', 'NewLeague'], axis=1).astype('float64')
# Standardizing X
list_numerical = X.columns
scaler1 = StandardScaler().fit(X[list_numerical])
X = scaler1.transform(X)
# Split the data into training data and test data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=10)
# Training the Lasso regression
reg = Lasso(alpha=1)
reg.fit(X_train, y_train)
beta = reg.coef_
print('Coefficients', beta)
# Test data
pred = reg.predict(X_test)
mse_test =mean_squared_error(y_test, pred)
```

Python codes for example 3

```
# Plot of Lasso coefficients for different alpha
alphas = np.linspace(0.01,500,100)
lasso = Lasso(max_iter=10000)
coefs = []
for a in alphas:
    lasso.set_params(alpha=a)
    lasso.fit(X_train, y_train)
    coefs.append(lasso.coef_)
ax = plt.gca()
ax.plot(alphas, coefs)
ax.set_xscale('log')
plt.axis('tight')
plt.xlabel('alpha')
plt.ylabel('Standardized Coefficients')
plt.title('Lasso coefficients for different alpha')
plt.savefig('Lasso_coefficients.pdf')
plt.show()
# Lasso with 5-fold cross-validation
model = LassoCV(cv=5, random_state=0, max_iter=10000)
model.fit(X_train, y_train)
print('Optimal alpha', model.alpha_)
#Best model
lasso_best = Lasso(alpha=model.alpha_)
lasso_best.fit(X_train, y_train)
coefs = lasso_best.coef_
print(coefs)
best_mse_test = mean_squared_error(y_test, lasso_best.predict(X_test))
print('Best MSE test set', round(best_mse_test, 2))
```