# CS536 HW4: Regression and Error

Student Name: Xuenan Wang

NetID: xw336

---

**A Small Regression Example** Consider regression in one dimension, with a data set $\{(x_i, y_i)\}_{i=1,\ldots,m}$.

- Find a linear model that minimizes the training error, i.e., $\hat{w}$ and $\hat{b}$ to minimize

$$\sum_{i=1}^{m} (\hat{w}x_i + \hat{b} - y_i)^2. \quad (1)$$

We define the objective function as $E(w,b) = \sum_{i=1}^{m}(y_i - (wx_i + b))^2$.

To estimate w, we can set the derivatives to zero: $\frac{dE}{dw} = -2\sum_i \big((y_i - y) - w(x_i - x)\big)(x_i - x) = 0$. Solving the equation, we have:

$$w^* = \frac{\sum_i (y_i - Y)(x_i - X)}{\sum_i (x_i - X)^2}$$

To estimate b, we can set the derivatives to zero: $\frac{dE}{db} = -2\sum_i \big(y_i - (wx_i + b)\big) = 0$. Solving the equation, we have:

$$b^* = \frac{\sum_i y_i}{m} - w\frac{\sum_i x_i}{m} = Y - wX$$

---

- Assume there is some true linear model, such that $y_i = wx_i + b + \epsilon_i$, where noise variables $\epsilon_i$ are i.i.d. with $\epsilon_i \sim N(0, \sigma^2)$. Argue that the estimators are unbiased, i.e., $\mathbb{E}[\hat{w}] = w$ and $\mathbb{E}\left[\hat{b}\right] = b$. What are the variances of these estimators?

With $\epsilon^i \sim N(0, \sigma^2)$, we can have $\epsilon^i \sim N(0, I\sigma^2)$. Based on lecture notes, we can also have:

$$w^* \sim w + N(0, \Sigma^{-1}\sigma^2) = N(w, \Sigma^{-1}\sigma^2)$$

The distribution above indicates that the expected value of w* is w, which means $\mathrm{E}[w^*] = w$, and similar, $\mathrm{E}[b^*] = b$.

$$var(w^*) = \sum (w^* - \mathrm{E}[w^*])^2 = \frac{m\sigma^2}{m\sum_{i=1}^m x_i^2 - (\sum_{i=1}^m x_i)^2}$$

$$var(b^*) = \sum (b^* - \mathrm{E}[b^*])^2 = \frac{\sigma^2 \sum_{i=1}^m x_i^2}{m\sum_{i=1}^m x_i^2 - (\sum_{i=1}^m x_i)^2}$$

- Assume that each $x$ value was sampled from some underlying distribution with expectation $\mathbb{E}[x]$ and variance $\mathrm{Var}(x)$. Argue that in the limit, the error on $\hat{w}$ and $\hat{b}$ are approximately

$$\mathrm{Var}(\hat{w}) \approx \frac{\sigma^2}{m}\frac{1}{\mathrm{Var}(x)}$$
$$\mathrm{Var}(\hat{b}) \approx \frac{\sigma^2}{m}\frac{\mathbb{E}\left[x^2\right]}{\mathrm{Var}(x)}. \quad (2)$$

We can get the result of $var(w^*)$ and $var(b^*)$ from last question.

It is easy to see that $\mathrm{E}[x^2] = \frac{1}{m}\sum_{i=1}^m x_i^2$ and $\mathrm{E}[x]^2 = \frac{1}{m^2}(\sum_{i=1}^m x_i)^2$, so we have:

$$var(x) = \mathrm{E}[x^2] - \mathrm{E}[x]^2 = \frac{1}{m}\sum_{i=1}^m x_i^2 - \frac{1}{m^2}(\sum_{i=1}^m x_i)^2$$

Therefore, we got:

$$var(w^*) = \frac{m\sigma^2}{m\sum_{i=1}^m x_i^2 - (\sum_{i=1}^m x_i)^2} = \frac{\sigma^2}{m}\frac{1}{\frac{1}{m}\sum_{i=1}^m x_i^2 - \frac{1}{m^2}(\sum_{i=1}^m x_i)^2} = \frac{\sigma^2}{m}\frac{1}{var(x)}$$

$$var(b^*) = \frac{\sigma^2\sum_{i=1}^m x_i^2}{m\sum_{i=1}^m x_i^2 - (\sum_{i=1}^m x_i)^2} = \frac{\sigma^2}{m}\frac{\frac{1}{m}\sum_{i=1}^m x_i^2}{\frac{1}{m}\sum_{i=1}^m x_i^2 - \frac{1}{m^2}(\sum_{i=1}^m x_i)^2} = \frac{\sigma^2}{m}\frac{\mathrm{E}[x^2]}{var(x)}$$

- Argue that recentering the data ($x_i' = x_i - \mu$) and doing regression on the re-centered data produces the same error on $\hat{w}$ but *minimizes* the error on $\hat{b}$ when $\mu = \mathbb{E}[x]$ (which we approximate with the sample mean).

Assume $x_i^* = x_i - \mu$, we have:

$$\mathrm{E}[x^*] = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mathrm{E}[x])$$

$$\mathrm{E}[x^{*2}] = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mathrm{E}[x])^2 = \frac{1}{m}\sum_{i=1}^{m}x_i^2 - \frac{1}{m^2}\left(\sum_{i=1}^{m}x_i\right)^2 = \mathrm{E}[x^2] - \mathrm{E}[x]^2 = var(x)$$

Assume the variance after recentering is $var(w^*)^*$ and $var(b^*)^*$, then we have:

$$var(w^*)^* = \frac{\sigma^2}{m}\frac{1}{var(x^*)} = \frac{\sigma^2}{m}\frac{1}{var(x)} = var(w^*)$$

This means that doing regression on the re-centered data produces the same error on $w^*$.

$$var(b^*)^* = \frac{\sigma^2}{m}\frac{\mathrm{E}[x^{*2}]}{var(x^*)} = \frac{\sigma^2}{m}\frac{var(x)}{var(x)} = \frac{\sigma^2}{m}$$

Since $var(x) = \mathrm{E}[x^2] - \mathrm{E}[x]^2$, we have $var(x) \le \mathrm{E}[x^2]$, $\frac{\mathrm{E}[x^2]}{var(x)} \ge 1$, which means:

$$var(b^*)^* \le var(b^*)$$

The error on $b^*$ is minimized when $\mu = \mathrm{E}[x]$.

- Verify this numerically in the following way: Taking $m = 200, w = 1, b = 5, \sigma^2 = 0.1$.
  - Repeatedly perform the following numerical experiment: generate $x_1, \ldots, x_m \sim \mathrm{Unif}(100, 102)$, $y_i = wx_i + b + \epsilon_i$ (with $\epsilon_i$ as a normal, mean 0, variance $\sigma^2$), and $x_i' = x_i - 101$; compute $\hat{w}, \hat{b}$ based on the $\{(x_i, y_i)\}$ data, and $\hat{w}', \hat{b}'$ based on the $\{(x_i', y_i)\}$ data.
  - Do this 1000 times, and estimate the expected value and variance of $\hat{w}, \hat{w}', \hat{b}, \hat{b}'$. Do these results make sense? Do these results agree with the above limiting expressions?

Following are generated data points. Considering m=200 is way too big to present here, I'm doing a m=10 data points generation.

```
xw336@ilab3:~$ /usr/bin/python3 /ilab/users/xw336/MachineLearning/Regression.py
Xi: [101.02270146937673, 100.67981581052167, 101.66820892716487, 100.88191724911738, 100.4
498098490209, 100.84573761475325, 101.31786081036387, 101.29027685547749, 101.743273837178
22, 101.98751918525424]
Yi: [106.19985154337931, 105.54317417530099, 106.63891571859652, 105.81627805193696, 105.4
6891889638013, 105.7472969242733, 106.27348243454857, 106.17019010112656, 106.666118469542
51, 106.89882620816077]
Xiprime: [0.022701469376727346, -0.3201841894783257, 0.6682089271648692, -0.11808275088262
121, -0.5501901509791054, -0.15426238524675284, 0.31786081036386804, 0.2902768554774866, 0
.7432738371782222, 0.9875191852542429]
```

The $w^*$ and $b^*$ are computed as following:

```
xw336@ilab3:~$ /usr/bin/python3 /ilab/users/xw336/MachineLearning/Regression.py
w_star: 0.9933347811164239
b_star: 5.678344336707772
```

The $w^{*\prime}$ and $b^{*\prime}$ are computed as following:

```
xw336@ilab3:~$ /usr/bin/python3 /ilab/users/xw336/MachineLearning/Regression.py
w_star_prime: 1.0138845975080835
b_star_prime: 106.00720055797488
```

Following are the results for mean and variance over 1000 experiments:

```
xw336@ilab3:~$ /usr/bin/python3 /ilab/users/xw336/MachineLearning/Regression.py
mean(w): 0.9996452518199291
mean(b): 5.035803559393814
mean(w_prime): 0.9996452518205954
mean(b_prime): 105.99997399320134
var(w): 0.00015806696978076527
var(b): 1.6121647557915435
var(w_prime): 0.00015806696980201064
var(b_prime): 4.8337367002952105e-05
```

The data we got make sense. I manually checked all previous conclusion, and the results looks good.

- *Intuitively, why is there no change in the estimate of the slope when the data is shifted?*

By shifting data, what we are actually doing was move the data points all together horizontally. Since we did not perform any rotation on the data points, the model we have(linear regression) will stay the same slope. Therefore, the estimation of slope remain unchanged.

- Consider augmenting the data in the usual way, going from one dimensions to two dimensions, where the first coordinate of each $\underline{x}$ is just a constant 1. Argue that taking $\Sigma = X^{\mathrm{T}}X$ in the usual way, we get in the limit that

$$\Sigma \to m \begin{bmatrix} 1 & \mathbb{E}[x] \\ \mathbb{E}[x] & \mathbb{E}[x^2] \end{bmatrix} \tag{3}$$

Show that re-centering the data ($\Sigma' = (X')^{\mathrm{T}}(X')$, taking $x'_i = x_i - \mu$), the condition number $\kappa(\Sigma')$ is minimized taking $\mu = \mathbb{E}[x]$.

To recenter the data, we have:

$$\mathrm{E}[x^*] = \mathrm{E}[x - \mu] = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mathrm{E}[x]) = \mathrm{E}[x] - \mathrm{E}[x] = 0$$

$$\mathrm{E}\left[x^{*2}\right] = \mathrm{E}[(x-\mu)^2] = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mathrm{E}[x])^2 = \frac{1}{m}\sum_{i=1}^{m}x_i^2 - \frac{1}{m^2}\left(\sum_{i=1}^{m}x_i\right)^2 = \mathrm{E}[x^2] - \mathrm{E}[x]^2 = var(x)$$

$$\Sigma^* = \begin{bmatrix} 1 & \mathrm{E}[x^*] \\ \mathrm{E}[x^*] & \mathrm{E}\left[x^{*2}\right] \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & var(x) \end{bmatrix}$$

Solve the matrix:

$$\begin{vmatrix} 1-\lambda^* & 0 \\ 0 & var(x) - \lambda^* \end{vmatrix} = 0$$

$$\lambda_1^* = 1, \lambda_2^* = var(x)$$

Similar:

$$\begin{vmatrix} 1-\lambda & \mathrm{E}[x] \\ \mathrm{E}[x] & \mathrm{E}[x^2] - \lambda \end{vmatrix} = 0$$

$$\lambda_1 = \frac{1 + \mathrm{E}[x^2] + \sqrt{(\mathrm{E}[x^2]-1)^2 + 4\mathrm{E}[x]}}{2}, \lambda_2 = \frac{1 + \mathrm{E}[x^2] - \sqrt{(\mathrm{E}[x^2]-1)^2 + 4\mathrm{E}[x]}}{2}$$

We know that condition number $\kappa(\Sigma) = \frac{largest\ eigenvalue\ of\ \Sigma}{smallest\ eigenvalue\ of\ \Sigma}$

$$\kappa(\Sigma) = \frac{1 + \mathrm{E}[x^2] + \sqrt{(\mathrm{E}[x^2]-1)^2 + 4\mathrm{E}[x]}}{1 + \mathrm{E}[x^2] - \sqrt{(\mathrm{E}[x^2]-1)^2 + 4\mathrm{E}[x]}}$$

As for $\kappa(\Sigma)^*$, we need to compare $\lambda_1^*$ and $\lambda_2^*$:

If $\lambda_1^* > \lambda_2^*$:

$$\kappa(\Sigma)^* = \frac{1}{var(x)}$$

$$\kappa(\Sigma)^* - \kappa(\Sigma) = \frac{1}{\mathrm{E}[x^2] - \mathrm{E}[x]^2} - \frac{1 + \mathrm{E}[x^2] + \sqrt{(\mathrm{E}[x^2] - 1)^2 + 4\mathrm{E}[x]}}{1 + \mathrm{E}[x^2] - \sqrt{(\mathrm{E}[x^2] - 1)^2 + 4\mathrm{E}[x]}}$$

Since we have:

$$\mathrm{E}[x^2] - \mathrm{E}[x]^2 \leq 0$$

$$\frac{1 + \mathrm{E}[x^2] + \sqrt{(\mathrm{E}[x^2] - 1)^2 + 4\mathrm{E}[x]}}{1 + \mathrm{E}[x^2] - \sqrt{(\mathrm{E}[x^2] - 1)^2 + 4\mathrm{E}[x]}} \geq 0$$

$$\kappa(\Sigma)^* - \kappa(\Sigma) \leq 0$$

Which means that $\kappa(\Sigma)^*$ is minimized.

If $\lambda_1^* < \lambda_2^*$:

$$\kappa(\Sigma)^* = \frac{var(x)}{1} = var(x)$$

$$\kappa(\Sigma)^* - \kappa(\Sigma) = (\mathrm{E}[x^2] - \mathrm{E}[x]^2) - \frac{1 + \mathrm{E}[x^2] + \sqrt{(\mathrm{E}[x^2] - 1)^2 + 4\mathrm{E}[x]}}{1 + \mathrm{E}[x^2] - \sqrt{(\mathrm{E}[x^2] - 1)^2 + 4\mathrm{E}[x]}}$$

Since we have:

$$\mathrm{E}[x^2] - \mathrm{E}[x]^2 \leq 0$$

$$\frac{1 + \mathrm{E}[x^2] + \sqrt{(\mathrm{E}[x^2] - 1)^2 + 4\mathrm{E}[x]}}{1 + \mathrm{E}[x^2] - \sqrt{(\mathrm{E}[x^2] - 1)^2 + 4\mathrm{E}[x]}} \geq 0$$

$$\kappa(\Sigma)^* - \kappa(\Sigma) \leq 0$$

Which means that $\kappa(\Sigma)^*$ is minimized.

# Source Code (Python 3)

```
from matplotlib import pyplot as plt
from random import uniform
from random import normalvariate
from numpy import var
from numpy import mean

class Data:

    def __init__(self, m=200, w=1, b=5, sigma=0.1):
        self.m = m
        self.w = w
        self.b = b
        self.sigma = sigma
        self.x = [uniform(100, 102) for _ in range(self.m)]
        self.y = [self.w*xi+self.b+normalvariate(0, self.sigma) for xi in self.x]
        self.xPrime = [xi-101 for xi in self.x]

    def printData(self):
        print('Xi:', self.x)
        print('Yi:', self.y)
        print('Xiprime:', self.xPrime)

class LinearRegression:

    def __init__(self, x, y):
        self.x = x
        self.y = y
        self.model = self.train(self.x, self.y)

    def train(self, x, y):
        if len(x) != len(y): return None
        l = len(x)
```

```
        b = (sum(y)*sum([xi**2 for xi in x])-sum(x)*sum([x[i]*y[i] for i in range(l)]))/(l*sum([xi**2 for xi in x])-sum(x)**2)
        w = (l*sum(x[i]*y[i] for i in range(l))-sum(x)*sum(y))/(l*sum([xi**2 for xi in x])-sum(x)**2)
        return [w, b]

if __name__ == '__main__':
    # # Question 5-1
    # data = Data()
    # # data.printData()
    # model = LinearRegression(data.x, data.y)
    # print('w_star:', model.model[0])
    # print('b_star:', model.model[1])
    # modelPrime = LinearRegression(data.xPrime, data.y)
    # print('w_star_prime:', modelPrime.model[0])
    # print('b_star_prime:', modelPrime.model[1])

    # Question 5-2
    repeatNum = 1000
    w = []
    b = []
    w_prime = []
    b_prime = []
    for _ in range(repeatNum):
        data = Data()
        model = LinearRegression(data.x, data.y)
        modelPrime = LinearRegression(data.xPrime, data.y)
        w.append(model.model[0])
        b.append(model.model[1])
        w_prime.append(modelPrime.model[0])
        b_prime.append(modelPrime.model[1])
    print('mean(w):', mean(w))
    print('mean(b):', mean(b))
    print('mean(w_prime):', mean(w_prime))
    print('mean(b_prime):', mean(b_prime))
    print('var(w):', var(w))
    print('var(b):', var(b))
    print('var(w_prime):', var(w_prime))
```

```
print('var(b_prime):', var(b_prime))
```