ML HomeWork 1

Hasini Witharana

February 2023

1 Data Generator

1.1 Generating Parameters (Θ^*)

K parameters/coefficients θ_i (i from 1 to K) from a Gaussian distribution, $i.i.d$ with $\mu = 0$ and $\sigma^2 = 1$ (Figure 1).

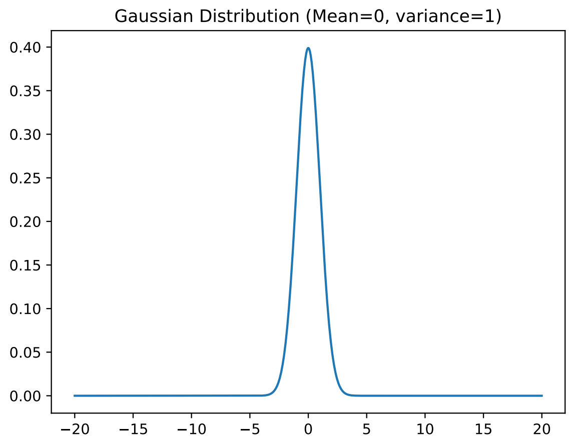

Figure 1: Gaussian distribution for parameters

1.2 Generating Covariates (x_i)

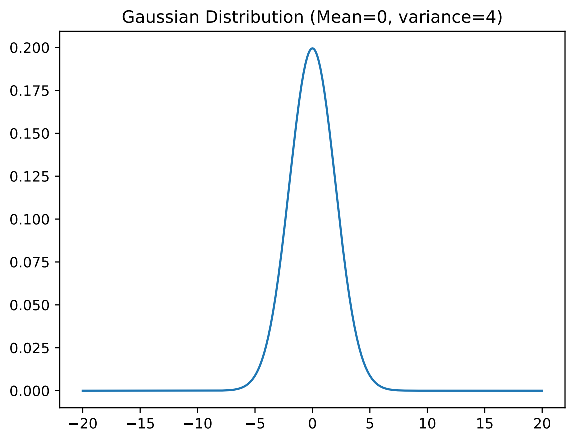

Figure 2: Gaussian distribution for covariates

The model for the covariates generation process is as follows. For each feature (i from 1 to K) x_i is drawn $i.i.d$ from a Gaussian distribution with $\mu = 0$ and $\sigma^2 = 4$ (Figure 2).

1.3 Generating Output (y)

The output y for any feature vector $\langle x_1, \ldots, x_K \rangle$ is $y = \sum_{i=1}^{K} x_i\theta_i + \epsilon$ where ϵ is drawn $i.i.d$ from a Gaussian distribution with $\mu = 0$ and $\sigma^2 = 0.1$ (Figure 3).

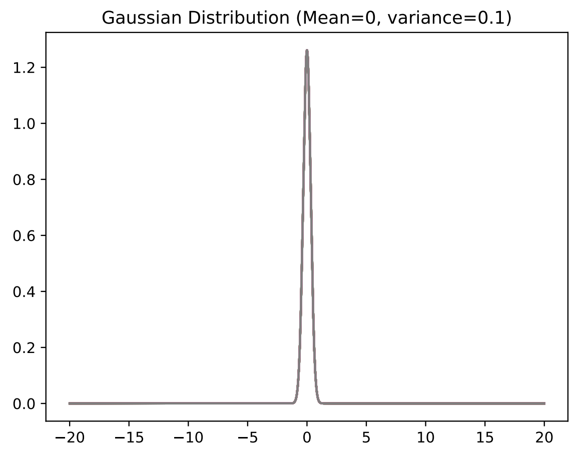

Figure 3: Gaussian distribution for ϵ

2 Ordinary Regression

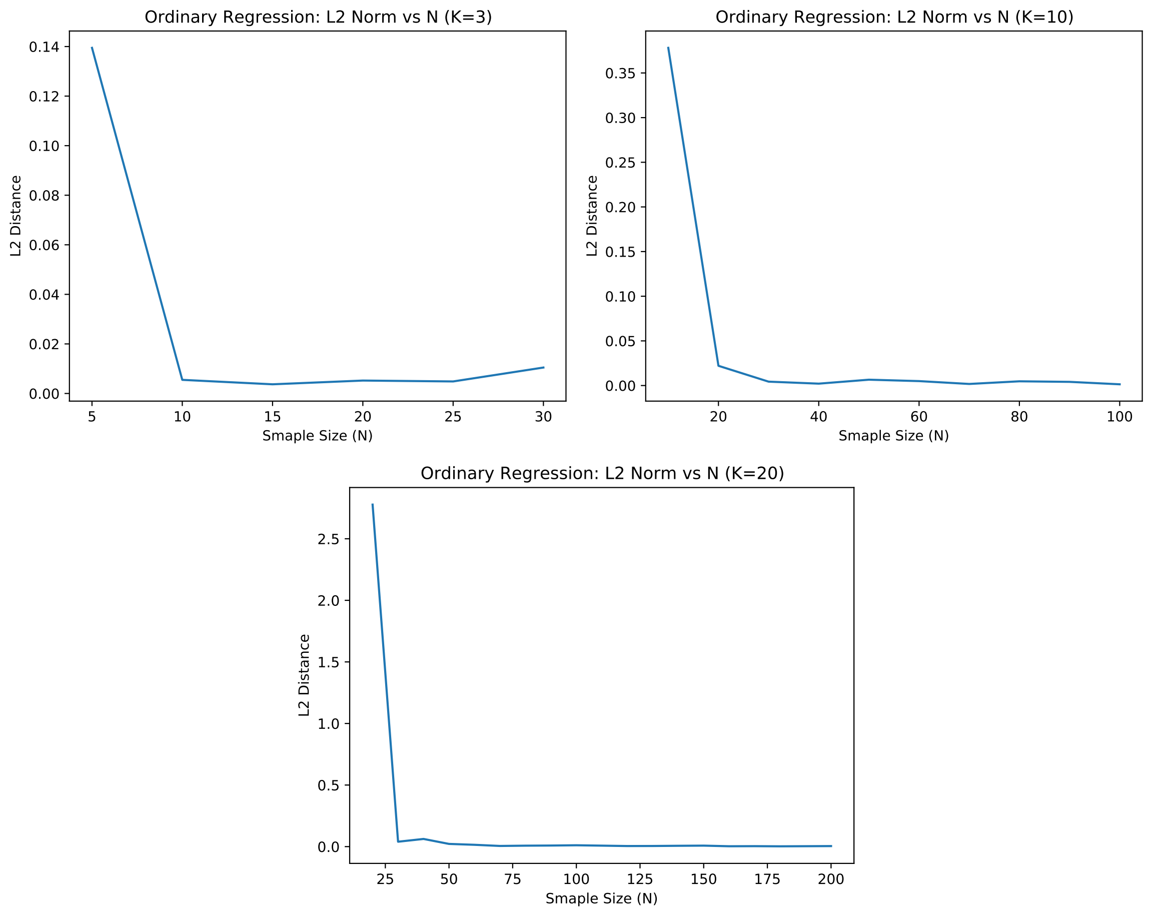

Figure 4: L2 distance for K=3,10,20

This section discusses the parameter prediction results using ordinary regression using $\theta = (X.X^T)^{-1}(X^T.y)$. For sample size (N) less than parameter size (K), sometimes we cannot find the prediction because $(X.X^T)^{-1}$ cannot be solved due to rank deficiency. Figure 4 presents the L2 distance (loss) between predicted and actual generated parameters for different K values (3, 10, 20). For all the K values L2 distance decreases when N is getting increased. For larger N values (cheap data) we can observe that the L2 loss is getting decreased. But we do not need that much of samples to reduce predict accurate results. For an example for K=20 the error is minimized when N=75.

3 Ridge Regression

This section discusses the parameter prediction results using ordinary regression using $\theta = (X.X^T + \lambda^2.I)^{-1}(X^T.y)$. For sample size (N) less than parameter size (K), sometimes we can find the prediction by choosing a correct λ.

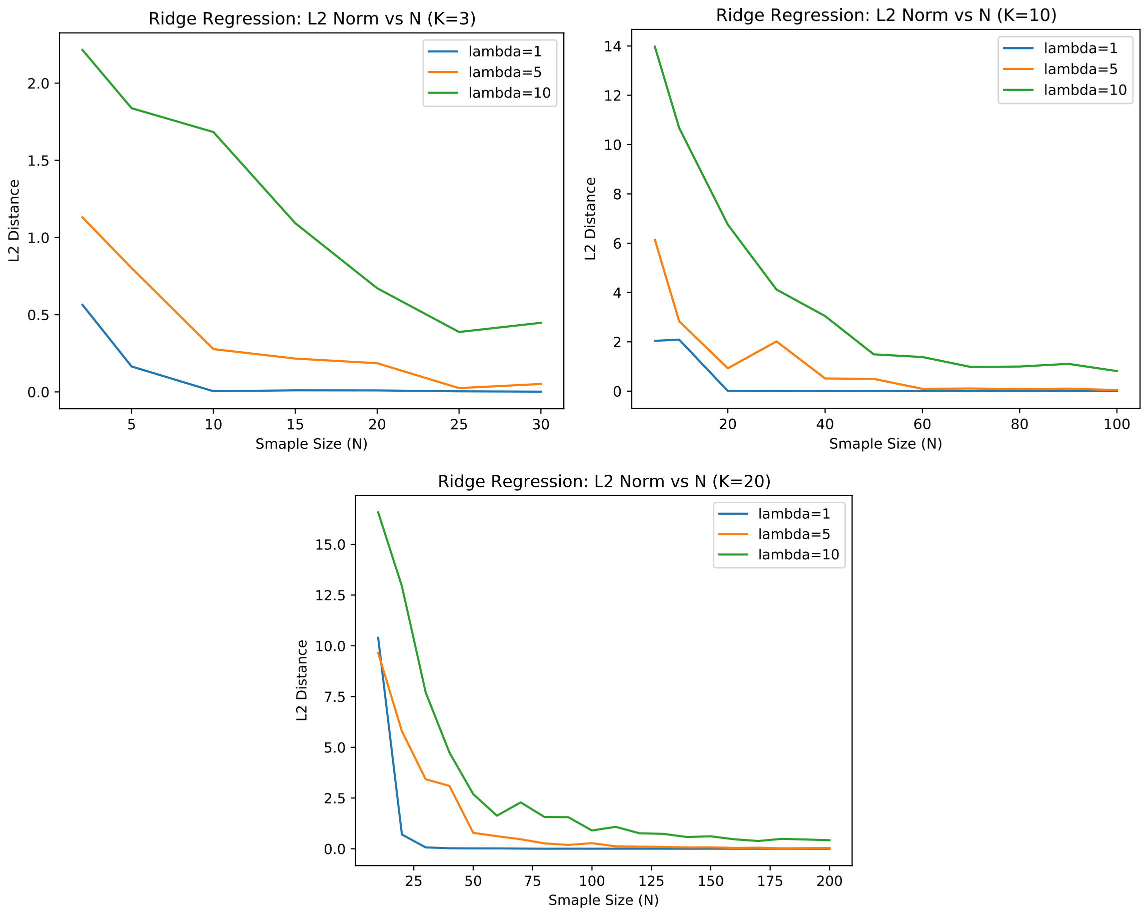

Figure 5: L2 distance for K=3,10,20 with large λ values

Figure 5 presents the L2 distance (loss) between predicted and actual generated parameters for different K values (3, 10, 20) and different large λ values (1, 5, 10). For all the K values L2 distance decreases when N is getting increased. For larger N values (cheap data) we can observe that the L2 loss is getting decreased. For large λ values error is getting increased when the λ is increased.

Figure 6 presents the L2 distance (loss) between predicted and actual generated parameters for different K values (3, 10, 20) and different small λ values. Most of the cases when λ is small and when we increase it the loss is getting smaller.

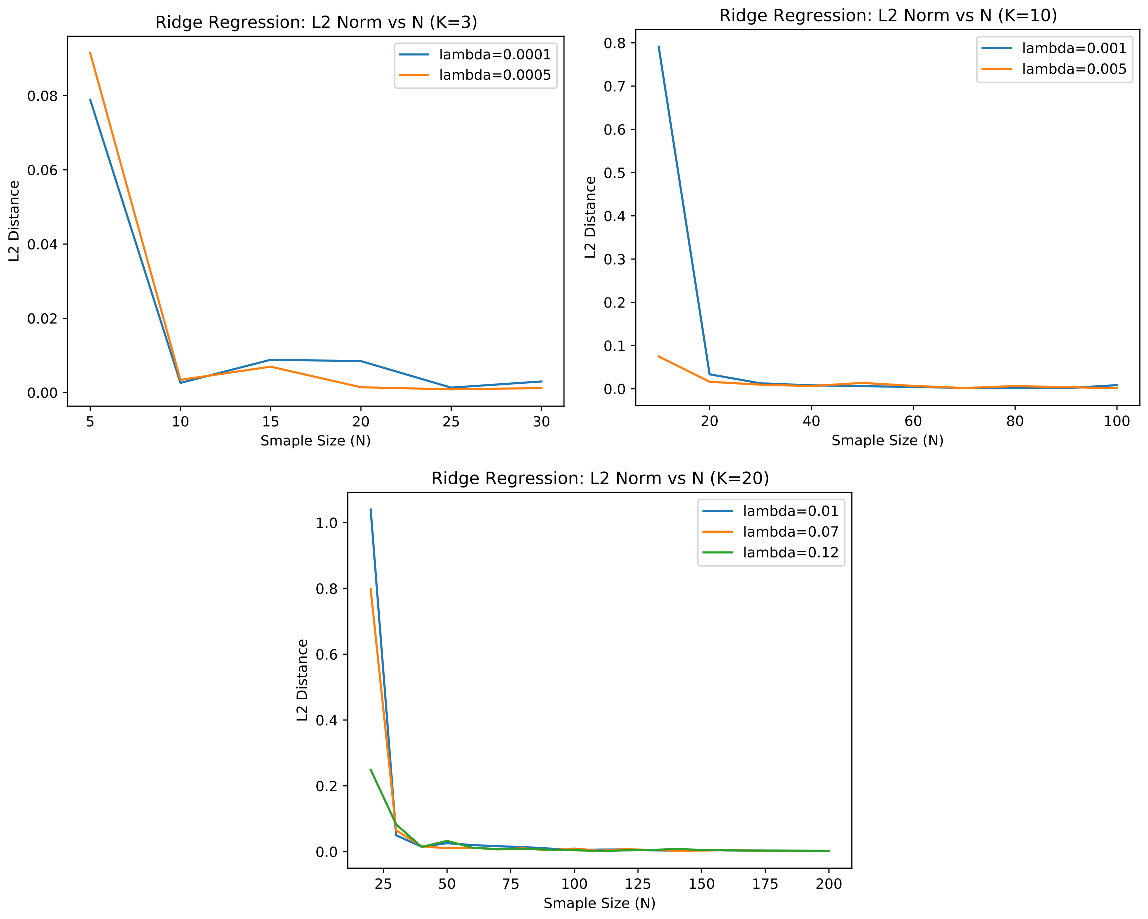

Figure 6: L2 distance for K=3,10,20 with small λ values

4 Code

4.1 Ordinary Regression

Listing 1: Script for ordinary regression

```
import numpy as np
from numpy import random
import matplotlib.pyplot as plt
from scipy.stats import norm
import math


K_list = [3]
N_list = [5,10,15,20,25,30]
normal_loss = [0]*len(N_list)

for K in K_list:
    # Derive parameters from Guassian distribution (mean = 0 and variance = 1)
    mean = 0
    std = 1
    params = random.normal(loc=mean, scale=std, size=(1, K))
    print(params)

    i=0
    for N in N_list:
        print(K,N)

        # Derive data from Guassian distribution (mean = 0 and variance = 4)
        mean = 0
        std = 2
        x = random.normal(loc=mean, scale=std, size=(N, K))

        # Derive epsilon from Guassian distribution (mean = 0 and variance = 0.1)
        mean = 0
        std = math.sqrt(0.1)
        epsilon = random.normal(loc=mean, scale=std, size=(N, 1))


        # Generating Data Set
        y=epsilon
        for n in range(N):
            for k in range(K):
                y[n] = y[n] + x[n][k]*params[0][k]

        # Calculate theta using OLS
        theta_best_values = np.linalg.inv(x.T.dot(x)).dot(x.T).dot(y)
        normal_results = theta_best_values.reshape(-1)
        print(normal_results)

        # Calculate L2 distance (Loss Function)
        l1 = 0
        for k in range(K):
            l1 += (normal_results[k]-params[0][k])**2
        loss1 = l1

        print("Ordinary Regression Loss : ", loss1)
        normal_loss[i] = loss1
        i+=1

print(normal_loss)
plt.plot(N_list,normal_loss)
plt.xlabel("Smaple Size (N)")
plt.ylabel("L2 Distance")
plt.title("Ordinary Regression: L2 Norm vs N (K="+str(K)+")")
plt.savefig("K"+str(K)+"OR.pdf", bbox_inches='tight')
plt.show()
```

4.2 Ridge Regression

Listing 2: Script for ridge regression

```
import numpy as np
from numpy import random
import matplotlib.pyplot as plt
from scipy.stats import norm
import math


K = 3
N_list = [2,5,10,15,20,25,30]
lamda_list = [1, 5, 10]

ridge_loss = [[0 for i in range(len(N_list))] for j in range(len(lamda_list))]



i=0
for lamda in lamda_list:
    # Derive parameters from Guassian distribution (mean = 0 and variance = 1)
    mean = 0
    std = 1
    params = random.normal(loc=mean, scale=std, size=(1, K))
    print(params)

    j=0
    for N in N_list:
        print(K,N)

        # Derive data from Guassian distribution (mean = 0 and variance = 4)
        mean = 0
        std = 2
        x = random.normal(loc=mean, scale=std, size=(N, K))


        # Derive epsilon from Guassian distribution (mean = 0 and variance = 0.1)
        mean = 0
        std = math.sqrt(0.1)
        epsilon = random.normal(loc=mean, scale=std, size=(N, 1))

        # Generating Data Set
        y=epsilon
        for n in range(N):
            # print(y[n])
            for k in range(K):
                y[n] = y[n] + x[n][k]*params[0][k]

        # Calculate Ridge Regression
        I = np.identity(K)
        theta_best_values = np.linalg.inv(x.T.dot(x)+(lamda**2)*(I)).dot(x.T).dot(y)
        ridge_results = theta_best_values.reshape(-1)
        print(ridge_results)

        # Calculate L2 distance (Loss Function)
        l2 = 0
        for k in range(K):
            l2 += (ridge_results[k]-params[0][k])**2
        loss2 = l2

        print("Ridge Regression Loss : ", loss2)
        ridge_loss[i][j] = loss2
        j+=1
    i+=1

print(ridge_loss)
i=0
for loss in ridge_loss:
    print(loss)
    plt.plot(N_list, loss, label='lambda='+str(lamda_list[i]))
    i+=1
plt.xlabel("Smaple Size (N)")
plt.ylabel("L2 Distance")
plt.title("Ridge Regression: L2 Norm vs N (K="+str(K)+")")
plt.legend()
plt.savefig("K"+str(K)+"RR.pdf", bbox_inches='tight')
plt.show()
```