ECE 595 HW2

Ruijie Song

Feb.17.2021

Exercise 1: Loading Data via Python

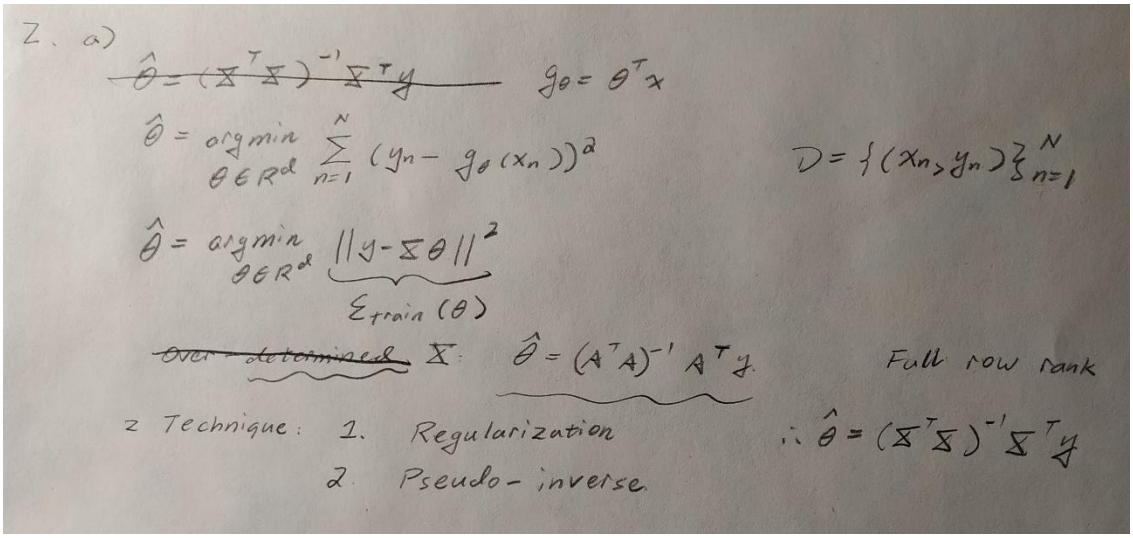
```
['index', 'male_bmi', 'male_stature_mm']
['0', '3.0', '1.679']
['1', '2.56', '1.586']
['2', '2.42', '1.773']
['3', '2.7399999999999998', '1.816']
['4', '2.59', '1.809']
['5', '2.5300000000000002', '1.662']
['6', '2.27', '1.829']
['7', '2.54', '1.686']
['8', '3.41', '1.761']
['9', '3.34', '1.797']

['index', 'female_bmi', 'female_stature_mm']
['0', '2.82', '1.563']
['1', '2.2199999999999998', '1.716']
['2', '2.71', '1.484']
['3', '2.81', '1.651']
['4', '2.55', '1.548']
['5', '2.3', '1.665']
['6', '3.56', '1.564']
['7', '3.1100000000000003', '1.676']
['8', '2.46', '1.69']
['9', '4.3', '1.704']
```

Figure 1. Exercise 1 result

Exercise 2: Build a Linear Classifier via Optimization

a.



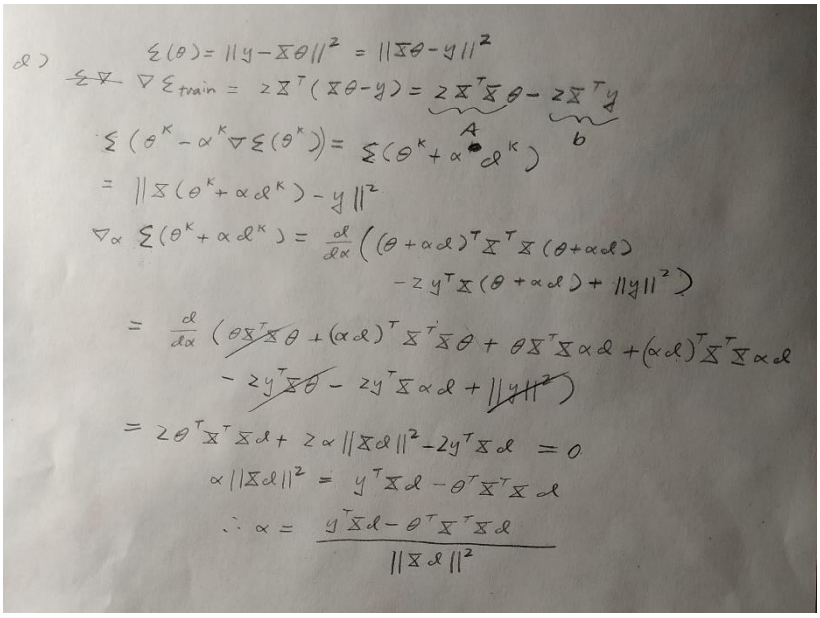
b.

theta = [-1.070e+01; -1.233e-01; 6.674e+00]

c.

theta = [-1.070e+01; -1.233e-01; 6.674e+00]

d.



e.

theta = [-1.070e+01; -1.233e-01; 6.674e+00]

f.

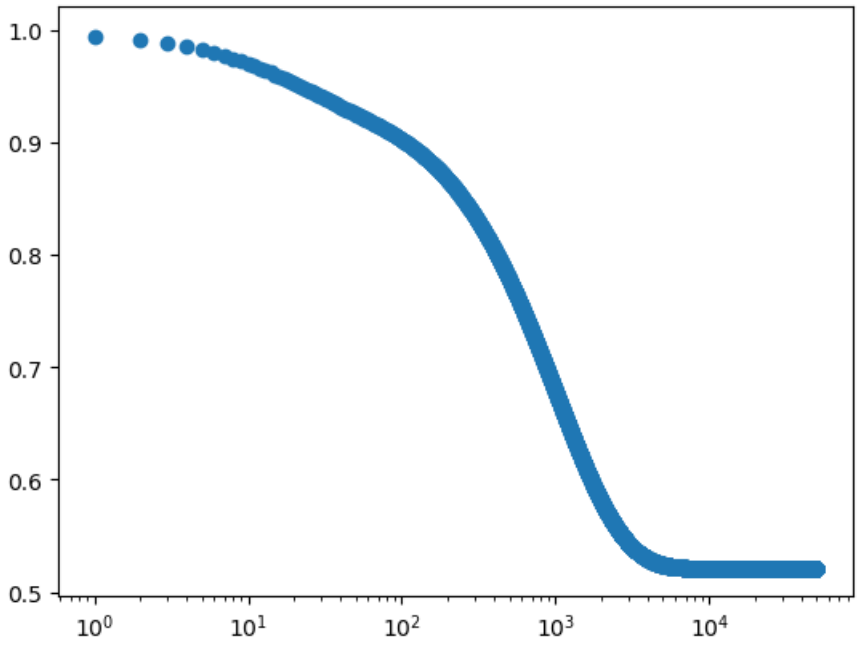


Figure 2. plot of the training loss

g.

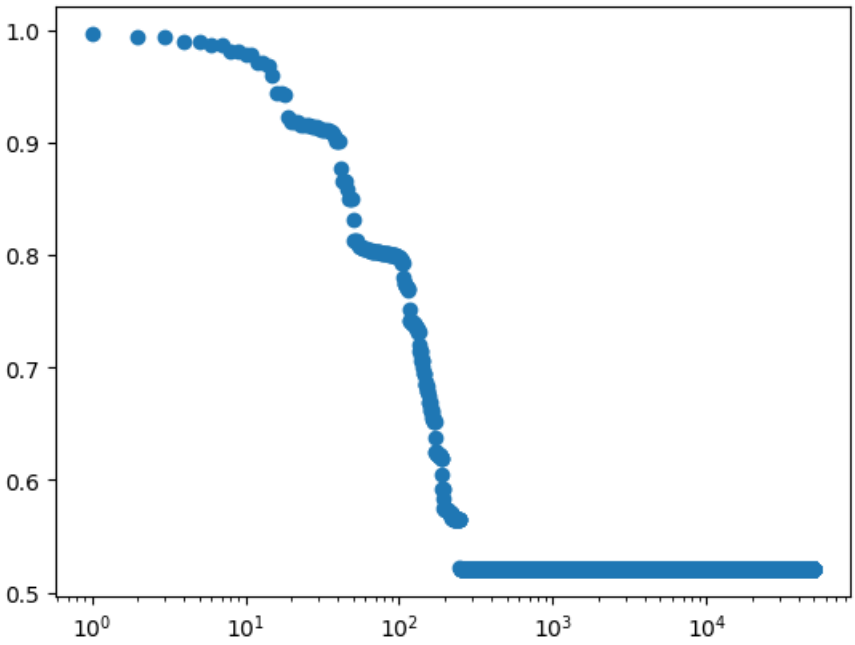
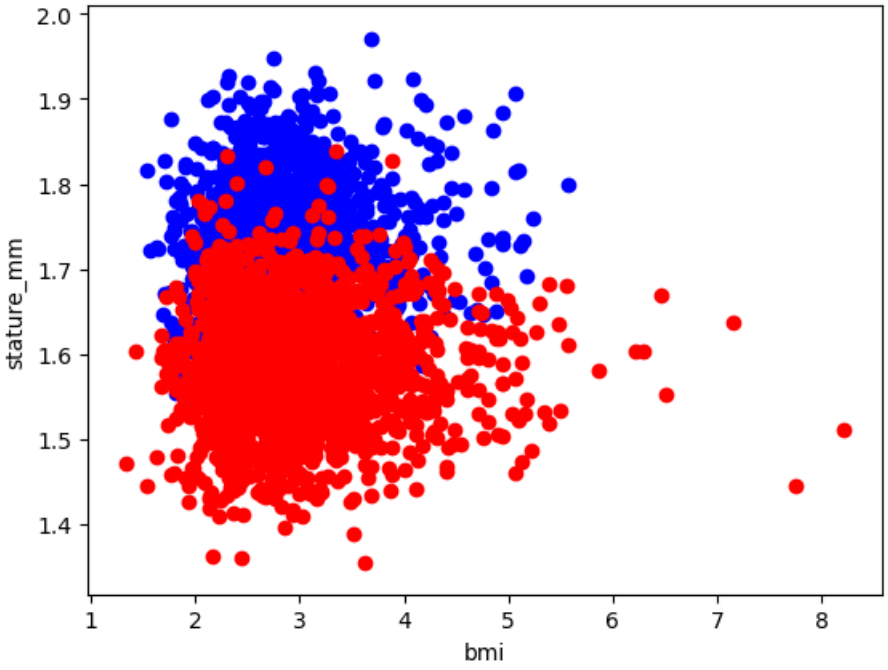


Figure 3. plot of the training loss

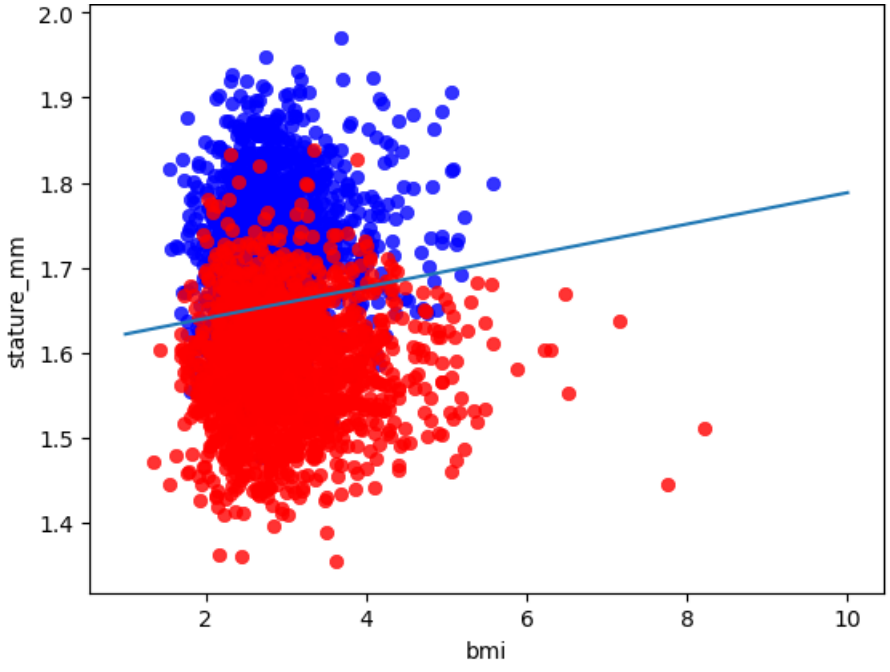
Exercise 3: Visualization and Testing

a.

i.



ii.

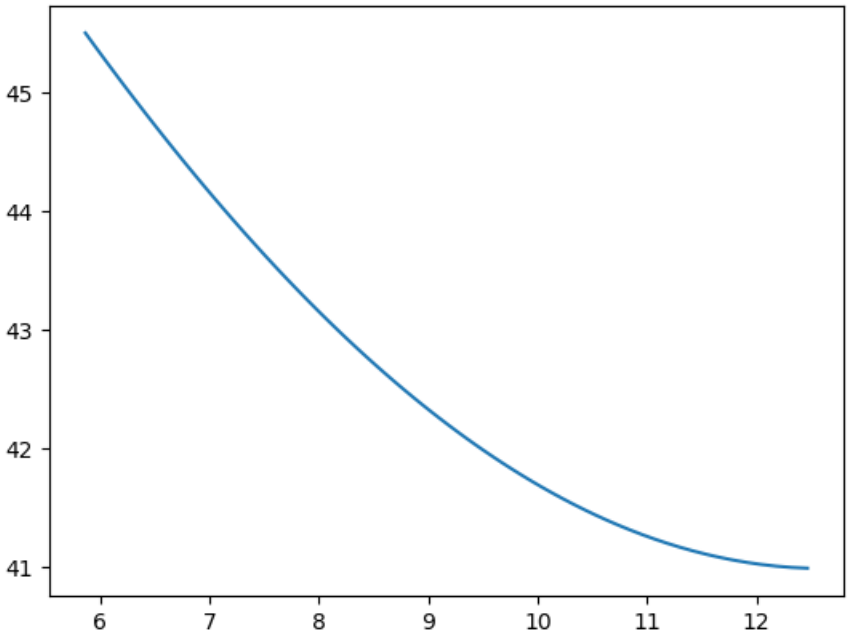


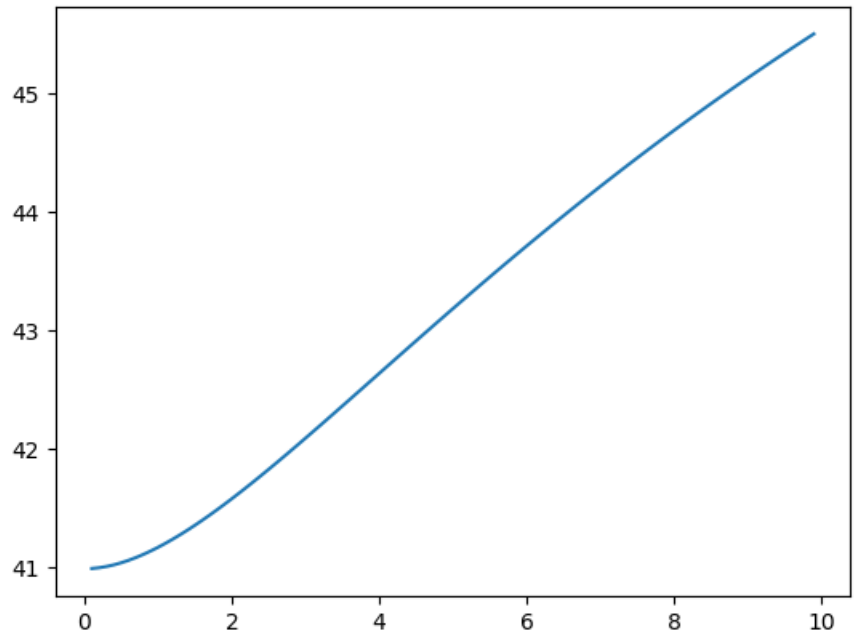
b

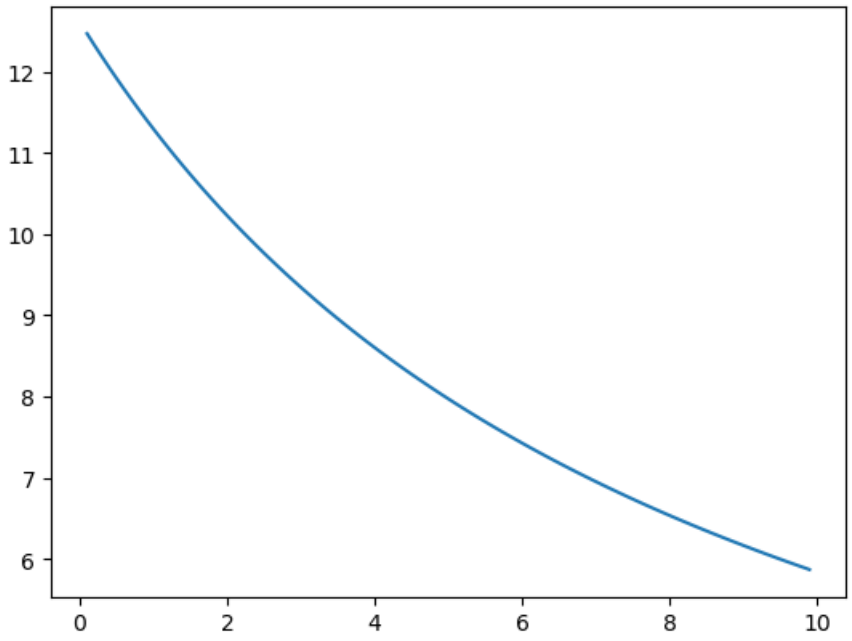
False Alarm = 15.02403846153846%
Miss = 18.653846153846154%
Precision = 0.9998804240631164
Recall = 0.9999036898201988

Exercise 4: Regularization

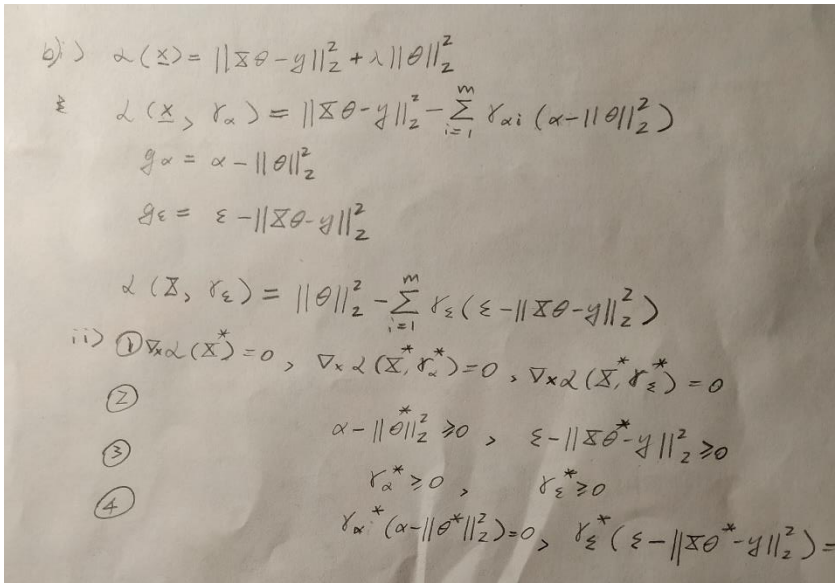
a.






12
11
10
9
8
7
6
0
2
4
6
8
10

b.


b)) $\alpha(\underline{x}) = \|X\theta - y\|_2^2 + \lambda\|\theta\|_2^2$
$\alpha(\underline{x}, \gamma_\alpha) = \|X\theta - y\|_2^2 - \sum_{i=1}^{m} \gamma_{\alpha i}(\alpha - \|\theta\|_2^2)$
$g_\alpha = \alpha - \|\theta\|_2^2$
$g_\varepsilon = \varepsilon - \|X\theta - y\|_2^2$
$\alpha(X, \gamma_\varepsilon) = \|\theta\|_2^2 - \sum_{i=1}^{m} \gamma_\varepsilon(\varepsilon - \|X\theta - y\|_2^2)$
ii) ① $\nabla_x \alpha(X^*) = 0$, $\nabla_x \alpha(X^*, \gamma_\alpha^*) = 0$, $\nabla_x \alpha(X^*, \gamma_\varepsilon^*) = 0$
②
③ $\alpha - \|\theta^*\|_2^2 \geq 0$, $\varepsilon - \|X\theta^* - y\|_2^2 \geq 0$
④ $\gamma_\alpha^* \geq 0$, $\gamma_\varepsilon^* \geq 0$
$\gamma_\alpha^*(\alpha - \|\theta^*\|_2^2) = 0$, $\gamma_\varepsilon^*(\varepsilon - \|X\theta^* - y\|_2^2) =$

iii)

$$\hat{\theta}_\lambda = \underset{\theta \in \mathbb{R}^d}{\arg\min} \|X\theta - y\|_2^2 + \lambda \|\theta\|_2^2$$

$$L(x) = \|X\theta - y\|_2^2 + \lambda \|\theta\|_2^2$$

$$\nabla_x L(x) = 2\|X\theta - y\| \pm = 0$$

$$\|X\theta - y\| = 0 \Rightarrow X\theta = y \qquad \hat{\theta}_\lambda = X^{-1}y$$

$$\gamma_\alpha (\alpha - \|\hat{\theta}_\lambda\|_2^2) = 0 \Rightarrow \gamma_\alpha (\alpha - \|X^{-1}y\|_2^2) = 0$$

~~$\gamma_{\alpha i}(\alpha - (x_i^{-1}y_i)^2) = 0$~~

~~if $\gamma_{\alpha i} = 0 \quad \alpha - (x_i^{-1}y_i)^2 > 0 \quad \alpha > (x_i^{-1}y_i)^2$~~

if $\gamma_\alpha = 0$, $\alpha \geq \|X^{-1}y\|_2^2$

if $\alpha - \|X^{-1}y\|_2^2 = 0$, $\gamma_\alpha > 0$

$$\alpha = \|X^{-1}y\|_2^2$$

iv)

$$\gamma_\varepsilon (\varepsilon - \|XX^{-1}y - y\|_2^2) = 0 \Rightarrow \gamma_\varepsilon \cdot \varepsilon = 0$$

$\gamma_\varepsilon = 0$ or $\varepsilon = 0$

v) $\nabla_\theta \left(\|X\theta - y\|_2^2 - \sum_{i=1}^{m} \gamma_\alpha (\alpha - \|\theta\|_2^2) \right) = 0$

$$2(X\theta - y) + \gamma_\alpha \cdot 2\theta = 0$$

$$(X + \gamma_\alpha)\theta = y \qquad \theta = (X + \gamma_\alpha)^{-1} y$$

If $\gamma_\alpha = 0$, . Yes

If $\gamma_\alpha > 0$, ~~Yes~~ No

$$\gamma_\alpha (\alpha - \|(X + \gamma_\alpha)^{-1}y\|_2^2) = 0$$

Since $\gamma_\alpha > 0$

$$(X + \gamma_\alpha)^{-1} y \neq X^{-1}y \neq \alpha$$

```python
# -*- coding: utf-8 -*-
"""
Created on Tue Feb 16 21:22:39 2021

@author: 11327
"""

import numpy as np
import matplotlib.pyplot as plt
import cvxpy as cp
import csv

# Exercise 1

# Reading csv file for male data
with open("./data/male_train_data.csv", "r") as csv_file:
    reader = csv.reader(csv_file, delimiter=',')
    # Add your code here to process the data into usable form
    i = 0
    data_m = np.array([])
    for line in reader:
        try:
            line[1] = str(float(line[1])/10) # normalizing BMI
        except:
            None
        try:
            line[2] = str(float(line[2])/1000) # normalizing stature_mm
        except:
            None
        if i == 0:
            data_m = np.array(line)
        else:
            data_m = np.row_stack((data_m,np.array(line)))
        i = i+1
    print(data_m[0:11,:])

csv_file.close()

print()

# Reading csv file for female data
with open("./data/female_train_data.csv", "r") as csv_file:
    reader = csv.reader(csv_file, delimiter=',')
    # Add your code here to process the data into usable form
    i = 0
    data_f = np.array([])
    for line in reader:
        try:
            line[1] = str(float(line[1])/10) # normalizing BMI
        except:
            None
        try:
            line[2] = str(float(line[2])/1000) # normalizing stature_mm
        except:
            None
        if i == 0:
            data_f = np.array(line)
        else:
            data_f = np.row_stack((data_f,np.array(line)))
        i = i+1
    print(data_f[0:11,:])
```

```
csv_file.close()



# Exercise 2
# b

data_m = np.around(data_m[1:,1:].astype(np.float), decimals = 3)
data_f = np.around(data_f[1:,1:].astype(np.float), decimals = 3)

X = np.row_stack((data_m,data_f))
# X = X.astype(np.float)

# add a column of 1 to the X
X = np.c_[np.ones(len(X)), X]

# Producing yn
len_m = len(data_m)
len_f = len(data_f)
y = np.concatenate((np.array(np.ones(len_m)), np.array(-1.0*np.ones(len_f))))

# Calculating theta
theta = np.dot(np.dot(np.linalg.pinv(np.dot(X.T,X)),X.T),y)

# c
THETA = cp.Variable(X.shape[1])
objective = cp.Minimize(cp.sum_squares(y - X@THETA))
prob = cp.Problem(objective)

result = prob.solve()
THETA_c = THETA.value

# e: Gradient descent
# Initialize gradient descent
d = X.shape[1]
max_itr = 50000
cost = np.zeros(max_itr)
theta_e = [0.,0.,0.]
XtX = np.dot( np.transpose(X), X)
theta_store = np.zeros((d,max_itr+1))
for i in range(d):
    theta_store[i,0] = 0

# Gradient descent
for itr in range(max_itr):
  dJ     = 2*(np.dot(XtX,theta_e) - np.dot(X.T,y))
  dd     = -1 * dJ
  alpha  = (np.dot(np.dot(y.T,X),dd) - np.dot(np.dot(theta_e,XtX),dd)) / np.sum((np.dot(X,dd))*
  theta_e  = theta_e + alpha*dd
  theta_store[:,itr+1] = theta_e
  cost[itr] = np.linalg.norm(y-np.dot(X, theta_e))**2/X.shape[0]

# f
plt.figure(1)
plt.semilogx(cost,'o',linewidth=8)

# g
beta = 0.9
# Initialize gradient descent
d = X.shape[1]
```

```
max_itr = 50000
cost = np.zeros(max_itr)
theta_g = [0.,0.,0.]
XtX = np.dot( np.transpose(X), X)
theta_store = np.zeros((d,max_itr+1))
for i in range(d):
    theta_store[i,0] = 0

# Gradient descent
itr = 0
dJ     = 2*(np.dot(XtX,theta_g) - np.dot(X.T,y))
dJ_lasttime = 2*(np.dot(XtX,theta_store[:,itr]) - np.dot(X.T,y))
dd     = -1 * (beta*dJ_lasttime + (1-beta)*dJ)
alpha  = (np.dot(np.dot(y.T,X),dd) - np.dot(np.dot(theta_g,XtX),dd)) / np.sum((np.dot(X,dd))**2)
theta_g  = theta_g + alpha*dd
theta_store[:,itr+1] = theta_g
cost[itr] = np.linalg.norm(y-np.dot(X, theta_g))**2/X.shape[0]

for itr in range(1,max_itr):
    dJ     = 2*(np.dot(XtX,theta_g) - np.dot(X.T,y))
    dJ_lasttime = 2*(np.dot(XtX,theta_store[:,itr-1]) - np.dot(X.T,y))
    dd     = -1 * (beta*dJ_lasttime + (1-beta)*dJ)
    alpha  = (np.dot(np.dot(y.T,X),dd) - np.dot(np.dot(theta_g,XtX),dd)) / np.sum((np.dot(X,dd))
    theta_g  = theta_g + alpha*dd
    theta_store[:,itr+1] = theta_g
    cost[itr] = np.linalg.norm(y-np.dot(X, theta_g))**2/X.shape[0]

# h
plt.figure(2)
plt.semilogx(cost,'o',linewidth=8)



# Exercise 3

# a
# i
plt.figure(3)
plt.scatter(data_m[:,0],data_m[:,1],c = 'b',alpha = 0.8,linewidth = 0.5)
plt.scatter(data_f[:,0],data_f[:,1],c = 'r',alpha = 0.8,linewidth = 0.5)
plt.xlabel('bmi')
plt.ylabel('stature_mm')

# ii
xaxis = np.linspace(1,10,100)
yaxis = (-1*theta[0]-theta[1]*xaxis) / theta[2]
plt.plot(xaxis,yaxis)

# b
f_predicted_mm =  (-1*theta[0]-theta[1]*data_f[:,0]) / theta[2]
m_predicted_mm =  (-1*theta[0]-theta[1]*data_m[:,0]) / theta[2]

false_alarm = 0
for i in range(len_f):
    if (data_f[i,1] > f_predicted_mm[i]):
        false_alarm = false_alarm + 1
false_alarm = false_alarm / len_f

Miss = 0
for i in range(len_m):
    if (data_m[i,1] < m_predicted_mm[i]):
```

```
        Miss = Miss + 1
Miss = Miss / len_m

TP = len_m - Miss
FP = Miss
FN = false_alarm
precision = (TP)/(TP+FP)
recall = TP/(TP+FN)



# Exercise 4: Regularization
# a

THETA_4 = []
lambd = np.arange(0.1,10,0.1)

for i in range(len(lambd)):
    THETA = cp.Variable(X.shape[1])
    objective = cp.Minimize(cp.sum_squares(X@THETA - y)+lambd[i]*cp.sum_squares(THETA))
    prob = cp.Problem(objective)

    result = prob.solve()
    THETA_4.append(THETA.value)

x_4a = []
y_4a = []
for i in range(len(lambd)):
    x_4a.append(np.linalg.norm(THETA_4[i], ord=2))
    y_4a.append(np.linalg.norm(X@THETA_4[i] - y, ord=2))
plt.figure(1)
plt.plot(x_4a, y_4a)

plt.figure(2)
plt.plot(lambd, y_4a)

plt.figure(3)
plt.plot(lambd, x_4a)
```

Implementing Noise2Noise for Dynamic Scenes

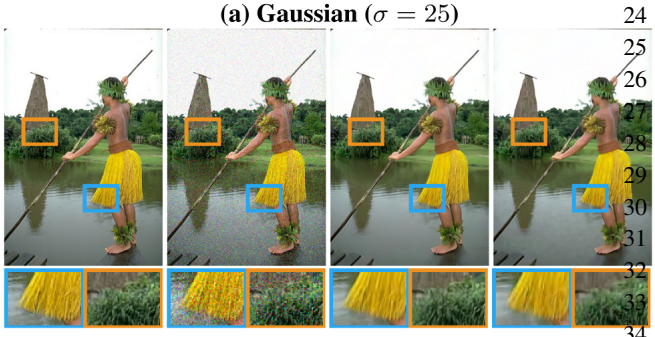
Ruijie Song M.S. [*1]

Abstract

Implementing Noise2Noise for Dynamic Scenes

1. Dataset

(a) Gaussian ($\sigma = 25$)



There's the dataset.

1.1. Codes

```
# Copyright (c) 2018, NVIDIA CORPORATION.
#
# This work is licensed under the Creative
# 4.0 International License. To view a cop
# http://creativecommons.org/licenses/by-n
# Creative Commons, PO Box 1866, Mountain

import tensorflow as tf

def parse_tfrecord_tf(record):
    features = tf.parse_single_example(rec
        'shape': tf.FixedLenFeature([3], t
        'data': tf.FixedLenFeature([], tf.
    data = tf.decode_raw(features['data'],
    return tf.reshape(data, features['shape

# [c,h,w] -> [h,w,c]
def chw_to_hwc(x):
    return tf.transpose(x, perm=[1, 2, 0])

# [h,w,c] -> [c,h,w]
def hwc_to_chw(x):
    return tf.transpose(x, perm=[2, 0, 1])

def resize_small_image(x):
    shape = tf.shape(x)
    return tf.cond(
        tf.logical_or(
            tf.less(shape[2], 256),
            tf.less(shape[1], 256)
        ),
        true_fn=lambda: hwc_to_chw(tf.image.resize_i
        false_fn=lambda: tf.cast(x, tf.float32)
    )

def random_crop_noised_clean(x, add_noise):
    cropped = tf.random_crop(resize_small_image(x),
    return (add_noise(cropped), add_noise(cropped),

def create_dataset(train_tfrecords, minibatch_size,
    print('Setting up dataset source from', train_
    buffer_mb  = 256
    num_threads = 2
    dset = tf.data.TFRecordDataset(train_tfrecords,
    dset = dset.repeat()
    buf_size = 1000
    dset = dset.prefetch(buf_size)
    dset = dset.map(parse_tfrecord_tf, num_parallel
    dset = dset.shuffle(buffer_size=buf_size)
    dset = dset.map(lambda x: random_crop_noised_cl
    dset = dset.batch(minibatch_size)
    it = dset.make_one_shot_iterator()
    return it
```

2. check point 2 basline

We recommend that you build supplementary material in a separate document. If you must create one PDF and cut it up, please be careful to use a tool that doesn't alter the margins, and that doesn't aggressively rewrite the PDF file. pdftk usually works fine.

[*]Equal contribution [1]Department of Electrical and Computer Engineering, Purdue University, West Lafayette, USA. Correspondence to: Cieua Vvvvv <c.vvvvv@googol.com>, Eee Pppp <ep@eden.co.uk>.

Proceedings of the 38th International Conference on Machine Learning, PMLR 139, 2021.

We recommend that you build supplementary material in a separate document. If you must create one PDF and cut it up, please be careful to use a tool that

We recommend that you build supplementary material in a separate document. If you must create one PDF and cut it up, please be careful to use a tool that doesn't alter the margins, and that doesn't aggressively rewrite the PDF file. pdftk usually works fine. We recommend that you build supplementary material in a separate document.We recommend that you build supplementary material in a separate document. If you must create one PDF and cut it up, please be careful to use a tool that d0oesn't alter the margins, and that doesn't aggressively rewrite the PDF file. pdftk usually works fine.

A. Do *not* have an appendix here

Do not put content after the references. Put anything that you might normally include after the references in a separate supplementary file.

We recommend that you build supplementary material in a separate document. If you must create one PDF and cut it up, please be careful to use a tool that doesn't alter the margins, and that doesn't aggressively rewrite the PDF file. pdftk usually works fine.

Please do not use Apple's preview to cut off supplementary material. In previous years it has altered margins, and created headaches at the camera-ready stage.