CS 760 Homework 3 by Cheng-Wei Lu

Problem 3.1

I normalize all non-nominal features and add an intercept to every data.

(a)

I set the iteration number of $\hat{\theta}$ update to 50000. Original step size = 1/20. I change the step size by multiply it with 3/4 every 50 iterations. The stop criteria is that when iteration number reaches 1000 or when the improvement for likelihood is less than 10^{-5}, the parameter update will stop.

(b)

It took about 57 seconds for my computer to converge.

(c)

Please note that I have normalized all non-nominal features in my data.

$$\hat{\theta} = \begin{pmatrix} -4.097(\text{class }) \\ 2.957(\text{sex}) \\ -2.070(\text{age}) \\ -0.259(\text{siblings/spouses aboard}) \\ -0.033(\text{parents/children aboard}) \\ -0.038(\text{fare}) \\ 4.773(\text{intercept}) \end{pmatrix}$$

(d)

likelihood $= -399.89523993$

(e)

The distribution of $\hat{\theta}$ is $N(\theta^*, I_{\theta^*}^{-1})$, where the θ^* is the real θ and the variance is represented by a covariance matrix $I_{\theta^*}^{-1}$, which is of the following value :

```
array([[ 0.14148657, -0.02824962,  0.05274623,  0.00238749, -0.00149039,
         0.01208782, -0.18864334],
       [-0.02824962,  0.04773139, -0.01226217, -0.00279729, -0.00192093,
        -0.00070188,  0.0254081 ],
       [ 0.05274623, -0.01226217,  0.06696317,  0.00471711,  0.00046209,
         0.00097325, -0.11758448],
       [ 0.00238749, -0.00279729,  0.00471711,  0.00406951, -0.0008746 ,
        -0.00042626, -0.00825114],
       [-0.00149039, -0.00192093,  0.00046209, -0.0008746 ,  0.00246007,
        -0.00065958,  0.00067411],
       [ 0.01208782, -0.00070188,  0.00097325, -0.00042626, -0.00065958,
         0.00516836, -0.01619391],
       [-0.18864334,  0.0254081 , -0.11758448, -0.00825114,  0.00067411,
        -0.01619391,  0.31810431]])
```

Problem 3.2

(a)

By the invariance property of MLE, we know that $\hat{w} = x^T\hat{\theta}$.

(b)

Since $\hat{\theta} \xrightarrow{d} N(\theta^*, I_{\theta^*}^{-1})$, and $\hat{w} = x^T\hat{\theta}$. It implies $\hat{w} \xrightarrow{d} N(x^T\theta^*, x^T I_{\theta^*}^{-1} x)$.

Problem 3.3

For this part, I create a data as follow:

$$x = \begin{pmatrix} 2(\text{class}) \\ 0(\text{sex}) \\ 25(\text{age}) \\ 0(\text{siblings/spouses aboard}) \\ 0(\text{parents/children aboard}) \\ 50(\text{fare}) \\ 1(\text{intercept}) \end{pmatrix}$$

Therefore, the normalized data would be :

$$x_{normalized} = \begin{pmatrix} 0.867(\text{class}) \\ 0(\text{sex}) \\ 0.848(\text{age}) \\ 0(\text{siblings/spouses aboard}) \\ 0(\text{parents/children aboard}) \\ 1.547(\text{fare}) \\ 1(\text{intercept}) \end{pmatrix}$$

(a)

According to the logistic prediction and setting odds threshold bigger than 1/2 as being able to survive, I would not have survived the sinking because that I am a male and I am not young and rich enough. Just like in the movie - they have to let women and children escape first.

(b)

According to the prediction,the log odds would be -0.5965 . The variance for odds is calculated according to $x^T I_{\theta^*}^{-1} x$. The standard deviation is therefore 0.1444. The z value for $\alpha = 0.05$ is approximately 1.96, therefore the confidence interval for my log odds of survival is $[-0.8797, -0.3133]$.

(c)

If we are talking about the odds, I think has big variance. But if we are talking about whether I would have died, it is certain because with 95 percent confidence, even the at upper bound of my odds interval, my probability to have survived is 0.422, which is still less than 1/2 (the standard to see if one would have died).

Problem 3.4

(a)

```
For feature 1 : Pclass
    Chisquare score [118.68092055]
    P-value =  [1.] > 95% Therefore, feature 1 is significant.
----------------------------------------
For feature 2 : Sex
    Chisquare score [183.23195285]
    P-value =  [1.] > 95% Therefore, feature 2 is significant.
----------------------------------------
For feature 3 : Age
    Chisquare score [64.03212709]
    P-value =  [1.] > 95% Therefore, feature 3 is significant.
----------------------------------------
For feature 4 : Siblings/Spouses Aboard
    Chisquare score [16.50584764]
    P-value =  [0.9999515] > 95% Therefore, feature 4 is significant.
----------------------------------------
For feature 5 : Parents/Children Aboard
    Chisquare score [0.46983439]
    P-value =  [0.50693663] < 95% Therefore, feature 5 is not significant.
----------------------------------------
For feature 6 : Fare
    Chisquare score [0.28166237]
    P-value =  [0.40438629] < 95% Therefore, feature 6 is not significant.
----------------------------------------
intercept
    Chisquare score [71.63897576]
    P-value =  [1.] > 95% Therefore, feature 7 is significant.
----------------------------------------
```

I used $\frac{\hat{\theta}_J^2}{v_j^2}$ to calculate the chi-score for each feature, and

(b)

Based on the result above, pclass ,sex, age, siblings/spouses aboard, and intercept are significant features.

(c)

The most significant feature is sex according to the chi-square score. If I change my sex, I would be alive with the probability of survival equal 0.913.

hw3_Appendix-Copy1

October 22, 2020

[1]:
```
import csv
import numpy as np
import copy
import time
import math
from scipy import stats
```

[2]:
```
def split_train_label(data):
    train_x = []
    train_y = []
    for i in data:
        train_x.append(i[1:])
        train_y.append([i[0]])

    return train_x,train_y
```

[3]:
```
def normalize(x):
    answer = copy.deepcopy(x)
    for i in range(len(x)):
        if normalized_avg[i] != 0:
            answer[i] = answer[i]/normalized_avg[i]
    return answer
```

[4]:
```
with open('titanic_data.csv','r') as file:
    temp = csv.reader(file)
    data = list(temp)

header = data[0]
data = data[1:]
for i in range(len(data)):
    row_len = len(data[0])
    for j in range(row_len):
        data[i][j] = float(data[i][j])

train_x, train_y = split_train_label(data)
```

[5]:
```
header
```

[5]: ['Survived',
 'Pclass',
 'Sex',
 'Age',
 'Siblings/Spouses Aboard',
 'Parents/Children Aboard',
 'Fare']

1 one hot encoding and normalization

[6]:
```
#normalization
normalized_avg = []
for i in range(len(header[1:])):
    total = 0
    avg = 0
    if header[1:][i] not in ['Sex']:
        for j in train_x:
            total += j[i]
        avg = total/len(train_x)
        for j in range(len(train_x)):
            train_x[j][i] = train_x[j][i]/avg
    normalized_avg.append(avg)
normalized_avg.append(0) # intercept
```

2 logistic regression

[7]:
```
# each x data is a row vector. y is a column vector. Need header to do␣
↪normalization

class logistic_titanic:
    def __init__(self, gradientRate= 3/4, max_iter = 50000, abstol = 1e-5,␣
↪add_intercept = True):
        self.max_iter =  max_iter
        self.abstol = abstol
        #self.reltol = reltol
        self.add_intercept = add_intercept
        self.gradientRate = gradientRate
        self.likelihoodScore = None

    def likelihood_score(self):
        likelihood = 0
        for i in range(len(self.training_y)):
            temp = 0
            x = np.array([self.training_x[i]]).T
```

```
            temp += self.training_y[i]*math.log(1/(1+ (math.exp((-np.dot(self.
↪theta.T,x))))))
            temp += (1-self.training_y[i])*math.log((1/(1+ (math.exp((np.
↪dot(self.theta.T,x)))))))
            likelihood += temp
        self.likelihoodScore = likelihood
        return likelihood
    def gradient(self):
        gradient = np.zeros((len(self.training_x[0]),1))
        for i in range(len(self.training_y)):
            x = np.array([self.training_x[i]]).T
            temp = self.training_y[i] - (1/(1+ (math.exp((-np.dot(self.theta.
↪T,x))))))
            gradient = gradient + temp*x

        return gradient

    def logistic_predict(self):
        answer = []
        for i in range(len(self.training_y)):
            temp = 0
            temp_x = np.array([self.training_x[i]]).T
            temp = 1/(1+math.exp(-np.dot(self.theta.T,temp_x)))
            if temp > 1/2:
                answer.append(1)
            else:
                answer.append(0)
        return answer

    def predict(self,x):##prediction one data
        temp_x = copy.deepcopy(x)
        if self. add_intercept == True:
            temp_x.append(1)
        temp_x = np.array([temp_x]).T
        odds = np.dot(self.theta.T,temp_x)
        probability = 1/(1+math.exp(-np.dot(self.theta.T,temp_x)))
        if probability > 1/2:
            return 1, probability, odds
        else:
            return 0, probability, odds

    def hessian(self):
        hessian = []
        temp = [0] * len(self.training_x[0])
        for i in range(len(self.training_x[0])):
            hessian.append(temp)
```

```
        hessian = np.array(hessian)
        for i in range(len(self.training_y)):
            temp = 0
            x = np.array([self.training_x[i]]).T
            temp = math.exp((-np.dot(self.theta.T,x)))/((1+math.exp((-np.
↪dot(self.theta.T,x))))**2)
            temp = temp * np.dot(x,x.T)
            hessian = hessian + temp

        return hessian
    def fit(self,x,y):
        ## deep copy data
        self.training_x = np.array(copy.deepcopy(x))
        self.training_y = np.array(copy.deepcopy(y))

        ## add intercept
        data_num = len(self.training_x)
        if self. add_intercept == True:
            temp = []
            for i in range(data_num):
                temp.append([1])
            self.training_x = np.append(self.training_x,temp,axis = 1)

        ## initialize theta
        theta = []

        for i in range(len(self.training_x[0])):
            theta.append([1])

        self.theta = np.array(theta)


        ## start training
        last_likelihood = float('-inf')
        parameter_rate = 1/20
        for i in range(self.max_iter):
            if i % 50 == 0:
                parameter_rate = parameter_rate*self.gradientRate
            current_likelihood = self.likelihood_score()
            if abs(current_likelihood - last_likelihood) <= self.abstol:
                break
            last_likelihood = current_likelihood
            gradient_val = self.gradient()
            self.theta = self.theta + parameter_rate*gradient_val

        return(self.theta)
```

[8]:
```
a =   logistic_titanic()
```

[9]:
```
start = time.time()
theta = a.fit(train_x, train_y)
end = time.time()
print('time = ', end - start)
```

```
time =  59.22251105308533
```

[10]:
```
a.likelihoodScore
```

[10]:
```
array([-399.89523993])
```

[11]:
```
current_hessian = a.hessian()

fisher_inv = np.linalg.inv(current_hessian)

x_my = normalize([2,0,25,0,0,50,1])
x_my = np.array([x_my]).T
w_variance = np.dot(x_my.T,np.dot(fisher_inv,x_my))
```

[12]:
```
theta
```

[12]:
```
array([[-4.09777457],
       [ 2.95734936],
       [-2.070699  ],
       [-0.25917329],
       [-0.03399745],
       [-0.03815405],
       [ 4.77374765]])
```

[13]:
```
w_variance**(1/2)
```

[13]:
```
array([[0.14449655]])
```

[14]:
```
x_my = normalize([2,0,25,0,0,50,1])
```

[15]:
```
x_my
```

[15]:
```
[0.867481662591687, 0, 0.8482787878277828, 0.0, 0.0, 1.5477278958396383, 1]
```

[16]:
```
prediction, probability ,odds = a.predict([0.867481662591687, 0, 0.
↪8482787878277828, 0.0, 0.0, 1.5477278958396383])
print(prediction, probability, odds)
```

```
0 0.355126805595913 [[-0.59657878]]
```

[17]:
```python
[odds -(1.96*w_variance**(1/2)) , odds +(1.96*w_variance**(1/2))]
```

[17]: [array([[-0.879792]]), array([[-0.31336555]])]

[18]:
```python
for i in range(len(theta)):
    if i <=5 :
        print("For feature", i+1, ":", header[i+1])
    else:
        print("intercept")
    score = (theta[i]**2)/fisher_inv[i][i]
    print("    Chisquare score", (theta[i]**2)/fisher_inv[i][i]  )
    test = stats.chi2.cdf(score, 1, loc=0, scale=1)
    if stats.chi2.cdf(score, 1, loc=0, scale=1) > 0.95:
        print("    P-value = ",test, "> 95%", "Therefore, feature" ,i+1 ,"is␣
 ↪significant.")
    else:
        print("    P-value = ",test, "< 95%", "Therefore, feature" ,i+1 ,"is␣
 ↪not significant.")
    print('----------------------------------------')
```

```
For feature 1 : Pclass
    Chisquare score [118.68092055]
    P-value =  [1.] > 95% Therefore, feature 1 is significant.
----------------------------------------
For feature 2 : Sex
    Chisquare score [183.23195285]
    P-value =  [1.] > 95% Therefore, feature 2 is significant.
----------------------------------------
For feature 3 : Age
    Chisquare score [64.03212709]
    P-value =  [1.] > 95% Therefore, feature 3 is significant.
----------------------------------------
For feature 4 : Siblings/Spouses Aboard
    Chisquare score [16.50584764]
    P-value =  [0.9999515] > 95% Therefore, feature 4 is significant.
----------------------------------------
For feature 5 : Parents/Children Aboard
    Chisquare score [0.46983439]
    P-value =  [0.50693663] < 95% Therefore, feature 5 is not significant.
----------------------------------------
For feature 6 : Fare
    Chisquare score [0.28166237]
    P-value =  [0.40438629] < 95% Therefore, feature 6 is not significant.
----------------------------------------
intercept
    Chisquare score [71.63897576]
    P-value =  [1.] > 95% Therefore, feature 7 is significant.
```