Mohammadsadegh Abolhasani
u1416052
HW1 CS6190
Spring 23

Analytical problems

1.

(a)

$$p(x|\mu,\sigma) = p(x|\mu) = \frac{1}{\sqrt{2\pi}\sigma}\exp\big(-\frac{1}{2\sigma^2}(x-\mu)^2\big)$$

$$\pi_j(\mu) = \sqrt{\mathbf{I}(\mu)}$$

For Fisher information we have:

$$\mathbf{I}(\mu) = -\mathbb{E}_{\mathbf{X}}[\frac{d^2 logp(\mathbf{X}|\mu)}{d\mu^2}]$$

$$\frac{d^2 logp(\mathbf{X}|\mu)}{d\mu^2} = \frac{d^2}{d\mu^2}\frac{1}{2\sigma^2}(X^2 - 2\mu X + \mu^2) = -\frac{1}{\sigma^2}$$

Then we should replace:

$$\pi_j(\mu) = \sqrt{-\mathbb{E}[\frac{d^2 logp(\mathbf{X}|\mu)}{d\mu^2}]}$$

After combining all of the above we have:

$$\pi_j(\mu) = \sqrt{\mathbf{I}(\mu)} = \sqrt{-\mathbb{E}_{\mathbf{X}}[\frac{d^2 logp(\mathbf{X}|\mu)}{d\mu^2}]} = \sqrt{\frac{1}{\sigma^2}} = \frac{1}{\sigma}$$

(b) In contrast to the last part, we will have a sigma constant here.

$$\pi_j(\sigma) = \sqrt{\mathbf{I}(\sigma)}$$

For Fisher's information, we have the:

$$\mathbf{I}(\sigma) = -\mathbb{E}_{\mathbf{X}}[\frac{d^2 logp(\mathbf{X}|\sigma)}{d\sigma^2}]$$

$$\frac{d^2 logp(\mathbf{X}|\sigma)}{d\sigma^2} = \frac{d^2}{d\sigma^2}(-\sigma - \frac{(x-\mu)^2}{2\sigma^2}) = -3\frac{(x-\mu)^2}{\sigma^4}$$

$$\pi_j(\sigma) = \sqrt{\mathbf{I}(\sigma)} = \sqrt{-\mathbb{E}_{\mathbf{X}}[\frac{d^2 logp(\mathbf{X}|\sigma)}{d\sigma^2}]} = \sqrt{-\mathbb{E}[-3\frac{(x-\mu)^2}{\sigma^4}]}$$

$$= \sqrt{\int_{-\infty}^{\infty} p(x|\sigma)3\frac{(x-\mu)^2}{\sigma^4}dx}$$

$$= \sqrt{\int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}\sigma}\exp\big(-\frac{1}{2\sigma^2}(x-\mu)^2\big)3\frac{(x-\mu)^2}{\sigma^4}dx}$$

$$= \sqrt{\frac{3}{\sigma^4\sqrt{2\pi}\sigma}\int_{-\infty}^{\infty}\exp\big(-\frac{1}{2\sigma^2}(x-\mu)^2\big)(x-\mu)^2 dx}$$

2.

$$\pi_j(\lambda) = \sqrt{\mathbf{I}(\lambda)}$$

For Fisher information we have:

$$\mathbf{I}(\lambda) = -\mathbb{E}_{\mathbf{X}}[\frac{d^2 logp(\mathbf{X}|\lambda)}{d\lambda^2}(\frac{d\lambda}{d\lambda})^2 + 0] = -\mathbb{E}_{\mathbf{X}}[\frac{d^2 logp(\mathbf{X}|\lambda)}{d\lambda^2}]$$

Then we should replace:

$$\pi_j(\lambda) = \sqrt{\mathbb{E}[\frac{d^2 logp(\mathbf{X}|\lambda)}{d\lambda^2}]}$$

$$\frac{d}{d\lambda}\log p(n|\lambda) = \frac{d}{d\lambda}\log\left(\frac{\lambda^n e^\lambda}{n!}\right) = \frac{d}{d\lambda}\left(n\log\lambda - \lambda - \log n!\right) = \frac{n}{\lambda} - 1 = \frac{n-\lambda}{\lambda}.$$

$$\pi_j(\lambda) = \sqrt{\mathbb{E}[\frac{d^2 logp(\mathbf{X}|\lambda)}{d\lambda^2}]} = \sqrt{\mathbb{E}[[\frac{dlogp(\mathbf{X}|\lambda)}{d\lambda}]]^2}$$

We replace the derivation of log-likelihood of $p(\mathbf{X}|\lambda)$

$$\pi_j(\lambda) = \sqrt{\mathbb{E}[\frac{d^2 logp(\mathbf{X}|\lambda)}{d\lambda^2}]} = \sqrt{\mathbb{E}[[\frac{n-\lambda}{\lambda}]^2]}$$

$$= \sqrt{\Sigma_{n=0}^{\infty} p(n|\lambda)(\frac{n-\lambda}{\lambda})^2} = \sqrt{\Sigma_{n=0}^{\infty} e^{-\lambda}\frac{\lambda}{n!}(\frac{n-\lambda}{\lambda})^2}$$

3. I will use induction to prove this. First, let's start with level one. Take two I.I.D variables x_1, x_2, then we have:

$$p(X_1 = x_1, X_2 = x_2) = p(X_1 = x_1) * p(X_2 = x_2) = p(X_2 = x_2) * p(X_1 = x_1) = p(X_2 = x_2, X_1 = x_1)$$

Now, we assume the nth level holds true, so we have:

$$p(X_1 = x_1, X_2 = x_2, ..., X_n = x_n) = p(X_1 = x_1) * p(X_2 = x_2) * ... * p(X_n = x_n)$$

Now for the next step which is $(n+1)thstep$ we have:

$$p(X_1 = x_1, X_2 = x_2, ..., X_n = x_n, X_{n+1} = x_{n+1}) = p(X_1 = x_1) * p(X_2 = x_2) * ... * p(X_n = x_n) * p(X_{n+1} = x_{n+1})$$

From the nth level we know that:

$$p(X_1 = x_1, X_2 = x_2, ..., X_n = x_n) = p(X_1 = x_1) * p(X_2 = x_2) * ... * p(X_n = x_n)$$

Therefore:

$$p(X_1 = x_1, X_2 = x_2, ..., X_n = x_n, X_{n+1} = x_{n+1}) = p(X_1 = x_1, X_2 = x_2, ..., X_n = x_n) * p(X_{n+1} = x_{n+1})$$

The above equation equals to:

$$p(X_1 = x_1, X_2 = x_2, ..., X_n = x_n, X_{n+1} = x_{n+1})$$

Which proves that $x_1, x_2, ...x_{n+1}$ are exchangeable.

4. I'll return to the slides for Polya's Urn in noninformative's slide

$$P(0,1,0,1) = \frac{W_0}{B_0+W_0} * \frac{B_0}{B_0+W_0+a-1} * \frac{W_0+a-1}{B_0+W_0+2a-2} * \frac{B_0+a-1}{B_0+W_0+3a-3}$$

$$P(1,1,0,0) = \frac{B_0}{B_0+W_0} * \frac{B_0+a-1}{B_0+W_0+a-1} * \frac{W_0}{B_0+W_0+2a-2} * \frac{W_0+a-1}{B_0+W_0+3a-3}$$

As we can see, both consist of 4 fractions multiplied by each other, the upper side are the same values that just have different orders, and the bottom side is the same. Therefore, the two fractions are the same and $P(0,1,0,1) = P(1,1,0,0)$

5. Newton-Raphson Scheme is:

$$W^{new} = W^{old} - \mathbf{H}^{-1}\nabla\mathbf{E}(\mathbf{w})$$

H is the Hessian Matrix.

$$\mathbf{H} = \nabla\nabla\mathbf{E}(\mathbf{w})$$

$$\mathbf{E}(\mathbf{w}) = -ln(p(\mathbf{t}|\mathbf{w}) - ln(p(\mathbf{w}))) = -\Sigma_{n=1}^{N} t_n ln(y_n) + (1-t_n)ln(1-y_n) + \frac{d}{2}\ln(2\pi\lambda) + \frac{1}{2\lambda}w^T w$$

For this task, we have to first derive $E(\mathbf{w})$ w.r.t. $\mathbf{w}$ one time to be replaced into the equation, and then for a second time to get the hessian matrix.

$$\nabla E(\mathbf{w}) = -\Sigma_{n=1}^{n} \frac{t_n}{y_n} y_n(1-y_n)\phi_n - \frac{(1-t_n)}{(1-y_n)} y_n(1-y_n)\phi_n + \frac{\mathbf{w}}{\lambda}$$

$$= -\Sigma_{n=1}^{n} t_n(1-y_n)\phi_n - (1-t_n)y_n\phi_n + \frac{\mathbf{w}}{\lambda}$$

$$= \Sigma_{n=1}^{n}(y_n - t_n)\phi_n + \frac{\mathbf{w}}{\lambda}$$

Now that we have the first derivative, we should repeat the process again:

$$H = \nabla\nabla E(\mathbf{w}) == \Sigma_{n=1}^{n}(1-y_n)y_n\phi_n^T\phi_n + \frac{\mathbf{I}}{\lambda}$$

Now we replace them into the formula:

$$w^{(new)} = w - (\Sigma_{n=1}^{n}(1-y_n)y_n\phi_n^T\phi_n + \frac{\mathbf{I}}{\lambda})^{-1}(\Sigma_{n=1}^{n}(y_n - t_n)\phi_n + \frac{\mathbf{w}}{\lambda})$$

6. The first step is like the previous problem. We have:
Newton-Raphson Scheme is:

$$W^{new} = W^{old} - \mathbf{H}^{-1}\nabla\mathbf{E}(\mathbf{w})$$

H is the Hessian Matrix.

$$\mathbf{H} = \nabla\nabla\mathbf{E}(\mathbf{w})$$

Now, the log-likelihood is:

$$\Sigma_{n=1}^{N}[y_n ln\Phi(\mathbf{w}^T x_n) + (1-y_n)ln(1-\Phi(\mathbf{w}^T x_n))] - \frac{1}{2\lambda}\mathbf{w}^T\mathbf{w}$$

Here, Φ function is the CDF of the standard normal distribution. Now we should derive it once first w.r.t. to $\mathbf{w}$ and then twice for the Hessian matrix. and then replace them inside the Newton-Raphson update formula:

$$\Sigma_{n=1}^{N}[\frac{y_n x_n \Phi'}{\Phi(\mathbf{w}^T x_n)} - \frac{(1-y_n)x_n\Phi'}{(1-\Phi(\mathbf{w}^T x_n))}] - \frac{1}{\lambda}\mathbf{w}^T$$

The derivation of $\Phi(\mathbf{w}^T x_n)$ w.r.t. $\mathbf{w}$ is $x_n\Phi(\mathbf{w}^T x_n)$. Therefore we have:

$$-\Sigma_{n=1}^{N}[\mathbf{y}_n + \frac{(1-y_n)x_n\Phi(\mathbf{w}^T x_n)}{(1-\Phi(\mathbf{w}^T x_n))}] - \frac{1}{\lambda}\mathbf{w}^T \ (1)$$

Now with the second derivation we have:

$$-\Sigma_{n=1}^{N}[\frac{\partial}{\partial\mathbf{w}}(\frac{(1-y_n)x_n\Phi(\mathbf{w}^T x_n)}{(1-\Phi(\mathbf{w}^T x_n))})] - \frac{1}{\lambda}\mathbf{I}$$

Now we can implement them into the Newton-Raphson Update in the format that has been provided above.

7.

(a) Logistic regression: A link function is a mathematical function that connects a response variable's mean or coefficient to a linear predictor. For logistic, we use Bernoulli distribution that belongs to the exponential family $y|\mu \sim Bernoulli(\mu)$:

$$f(y|\mu) = \mu^{\mathbf{y}}(1-\mu)^{1-y} = exp(ylog(\mu)+(1-y)log(1-\mu)) = exp(ylog(\mu)+log(1-\mu)-ylog(1-\mu))$$

$$= exp(ylog(\frac{\mu}{1-\mu}) + log(1-\mu))$$

By looking at the coefficient of y we see that $log(\frac{\mu}{1-\mu})$ is the link function.

(b) Like the previous part we have $y|\mu \sim Poisson(\mu)$, which belongs to the exponential family:

$$f(y|\mu) = \frac{\mu^y e^{-\mu}}{y!}$$

$$\log f(y|\mu) = \log \frac{\mu^y e^{-\mu}}{y!} = \log \mu^y + \log e^{-\mu} - \log y! = y \log \mu - \mu - \log y!$$

As it is visible, $y \log \mu$ is the link function.

8. I asked the TAs for help for this question. Here, we want to do a task of marginalization of z, and we head to do the integrals:

$$p(t=1) = \int_{-\infty}^{\infty} p(t=1|z)p(z)dz$$

And we know for the probit regression:

$$p(\mathbf{t}|z) = \mathbf{I}(t=0)\mathbf{I}(z \leq 0) + \mathbf{I}(t=1)\mathbf{I}(z \geq 0)$$

We now have:

$$p(t=1) = \int_{-\infty}^{\infty} \mathbf{I}(z \geq 0)p(z)dz$$

Where $\mathbf{I}(z \geq 0)$ equals to 1, so we separate the integral and we will have:

$$p(t=1) = \int_{0}^{\infty} p(z)dz$$

We will now rewrite it using the CDF of normal:

$$p(t=1) = 1 - \Phi(-w^T \phi_i)$$

Since the $\Phi(x)$ function is the PDF of normal distribution, we can rewrite it as:

$$\Phi(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{(2\pi)}} e^{-\frac{z^2}{2}} dz$$

We will now implement it to the $p(t=1)$:

$$p(t=1) = 1 - \int_{-\infty}^{-w^T \phi_i} \frac{1}{\sqrt{(2\pi)}} e^{-\frac{z^2}{2}} dz$$

Which indeed can be rewritten to:

$$\int_{-w^T \phi_i}^{\infty} \frac{1}{\sqrt{(2\pi)}} e^{-\frac{z^2}{2}} dz$$

Which is equal to the original likelihood of the probit regression.

9.

Practice

The codes for this section are written in python and are included together with the read-me file in the zip file that is attached to this assignment.

1.

(a) Here, we can see that the heat map for our prior is completely distributed along the ranges and has no concentration. Also, when we draw the lines, we see that they are spread over the area and contrary to the samples that we have selected, do not seem to follow any way of being close to the regression line that would go between the samples.

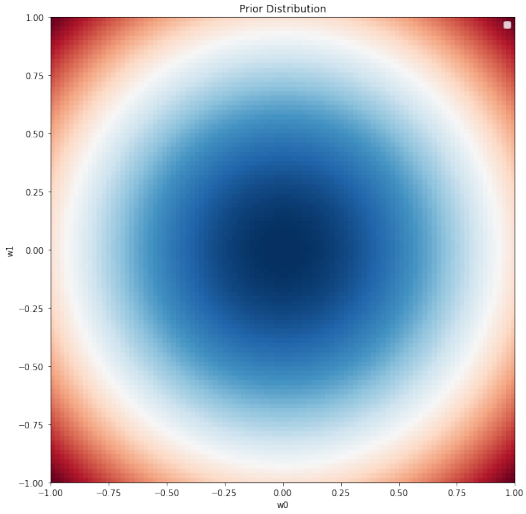

Figure 1: Practice 1, Part a: Heat map of the prior $p(\mathbf{w})$

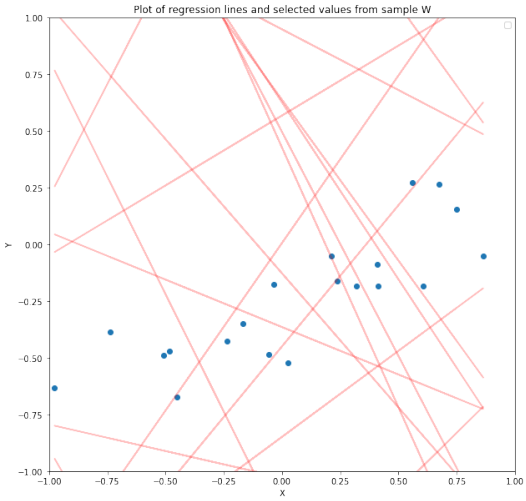

Figure 2: Practice 1, Part a: regression lines $y = w_0 + w_1 x$

(b) Here, we see that the posterior seems shrunken to an area and the regression lines are gathering together at an area around the single point sampled to the posterior

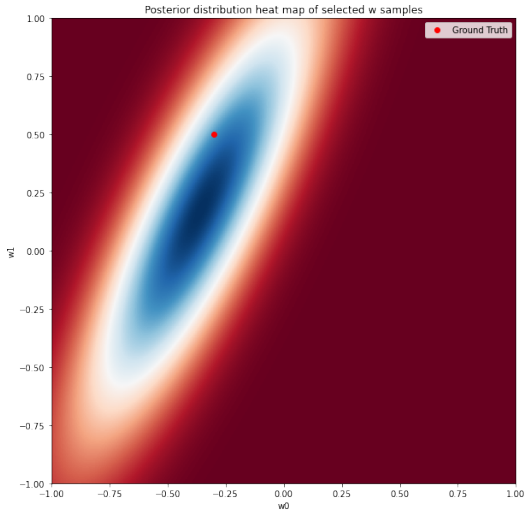

Figure 3: Practice 1, Part b: Heat map of the posterior $p(\mathbf{w})$

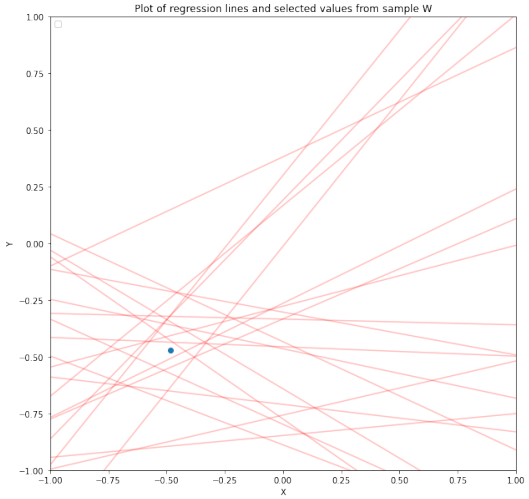

Figure 4: Practice 1, Part b: regression lines $y = w_0 + w_1 x$

(c) Here we can see the trend that with more samples being drawn to the posterior, the heat map area shrinks to an area around the ground truth and also regression lines seem to converge like a line that resembles the true regression line that fits all the samples.

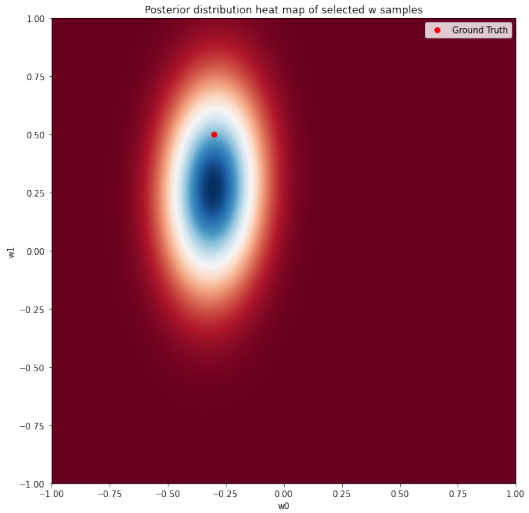

Figure 5: Practice 1, Part c: Heat map of the posterior $p(\mathbf{w})$

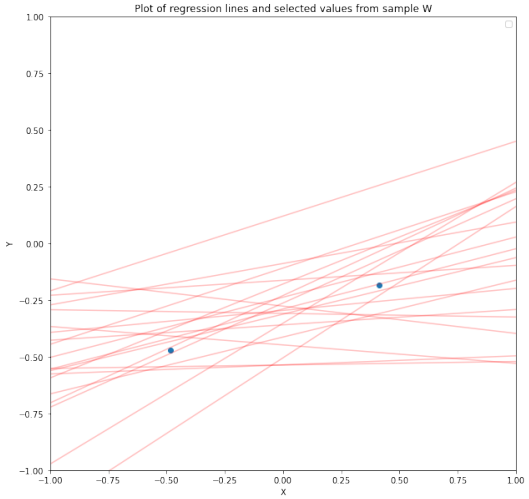

Figure 6: Practice 1, Part c: regression lines $y = w_0 + w_1 x$

(d) The same as it follows for part C, we can see that the heat map shrinks again and the lines become closer

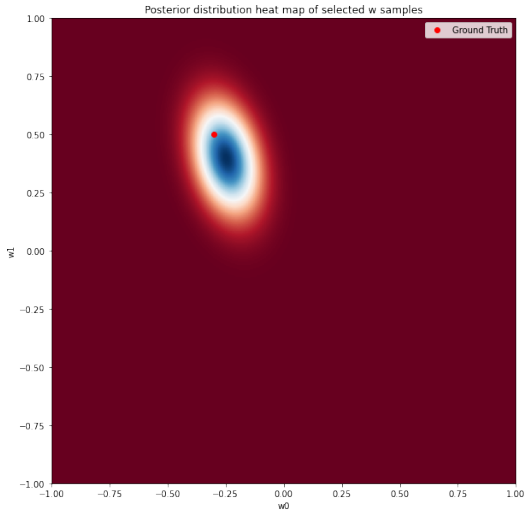

Figure 7: Practice 1, Part d: Heat map of the posterior $p(\mathbf{w})$

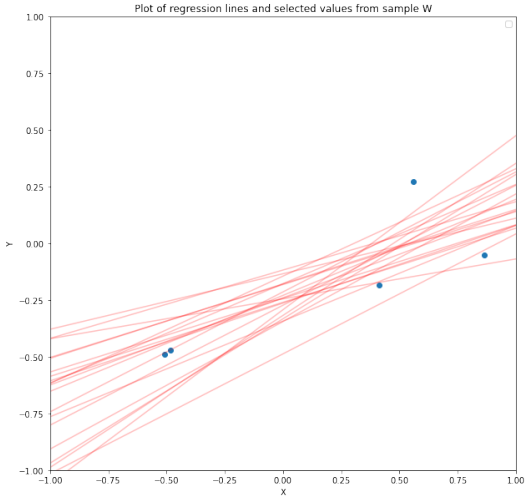

Figure 8: Practice 1, Part d: regression lines $y = w_0 + w_1 x$

(e) For the final part we see that these heat maps and lines closely resemble slide 29 of chapter 8 (generalized linear models). With more samples drawn to the posterior, we can see that the heat map of probabilities has been shrunk significantly to the area near the ground truth values and also the lines are close together mimicking the regression line that would go for the sample points shown in the plot.

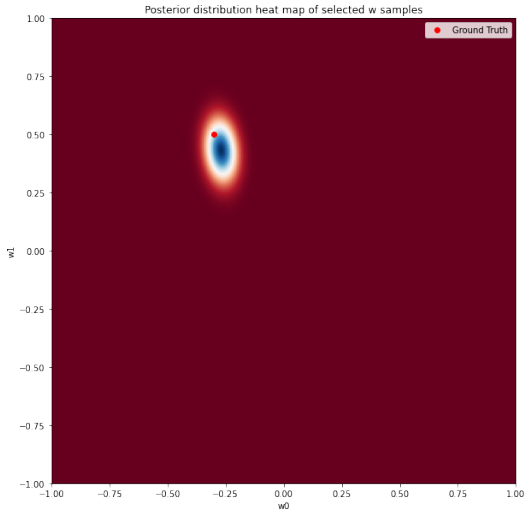

Figure 9: Practice 1, Part e: Heat map of the posterior $p(\mathbf{w})$

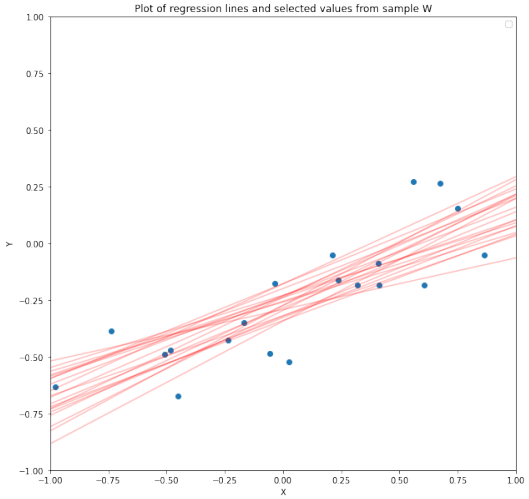

Figure 10: Practice 1, Part e: regression lines $y = w_0 + w_1 x$

2.

(a) I have ran the model many times and in the end, for reporting the results, I got an accuracy of 0.948 for when I am setting the initial weights to be zero and 0.938 for when the weights are set to be random and Gaussian (as asked in the problem). I talked to one of the TAs about it to get confirmation that we usually expect lower accuracies when we have random weights, but occasionally they may match or outperform the accuracy when weights are set to be zero for some random seeds.

```
Accuracy of prediction for Part A, weights set to: 0
0.948
Accuracy of prediction for Part A, weights set to: r
0.938
Accuracy of prediction for Part B, weights set to: 0
0.558
Accuracy of prediction for Part B, weights set to: r
0.812
Accuracy of prediction for Part C, weights set to: 0
0.952
Accuracy of prediction for Part C, weights set to: r
0.952
```

Figure 11: Problem 2: The accuracy results from the code run for all parts

(b) Here, I am having an accuracy of 0.558. Clearly, logistic regression shows a better accuracy, although initially, I was getting much lower accuracies, but after talking to TA I was advised to revise code and normalize the dataset. After doing so, I'm having better accuracies but still, they are much lower when I am using L-BFGS-B for maximization rather than using logistics. Also, for the randomization of weights, I saw results ranging from 0.02 to more than 0.8 for different randomization seeds. Here, I am keeping an accuracy of 0.812. Again, there is a possibility that my code is going south somewhere.

(c) I got the gradient and hessian matrices from the same function I used in the last part, and they seemed to perform ok compared to the last part when they were fed to the maximization function. Here, I'm keeping an accuracy of 0.952. Although, the same thing about variation that I said for the first part, held true here.

3.