CS 229, Autumn 2012
Problem Set #1 Solutions: Supervised Learning

Due in class (9:00am) on Wednesday, October 17.

Notes: (1) These questions require thought, but do not require long answers. Please be as concise as possible. (2) If you have a question about this homework, we encourage you to post your question on our Piazza forum, at `https://piazza.com/class#fall2012/cs229`. (3) If you missed the first lecture or are unfamiliar with the collaboration or honor code policy, please read the policy on Handout #1 (available from the course website) before starting work. (4) For problems that require programming, please include in your submission a printout of your code (with comments) and any figures that you are asked to plot. (5) Please indicate the submission time and number of late dates clearly in your submission.

SCPD students: Please email your solutions to `cs229-qa@cs.stanford.edu` with the subject line "Problem Set 1 Submission". If you are writing your solutions out by hand, please write clearly and in a reasonably large font using a dark pen to improve legibility.

1. **[25 points] Logistic regression**

   (a) [10 points] Consider the log-likelihood function for logistic regression:

   $$\ell(\theta) = \sum_{i=1}^{m} y^{(i)} \log h(x^{(i)}) + (1 - y^{(i)}) \log(1 - h(x^{(i)}))$$

   Find the Hessian H of this function, and show that for any vector z, it holds true that

   $$z^T H z \leq 0.$$

   [Hint: You might want to start by showing the fact that $\sum_i \sum_j z_i x_i x_j z_j = (x^T z)^2 \geq 0$.]

   Remark: This is one of the standard ways of showing that the matrix H is negative semi-definite, written "$H \leq 0$." This implies that ℓ is concave, and has no local maxima other than the global one.[1] If you have some other way of showing $H \leq 0$, you're also welcome to use your method instead of the one above.

   Answer: (Note we do things in a slightly shorter way here; this solution does not use the hint.) Recall that we have $g'(z) = g(z)(1 - g(z))$, and thus for $h(x) = g(\theta^T x)$, we have $\frac{\partial h(x)}{\partial \theta_k} = h(x)(1 - h(x))x_k$. This latter fact is very useful to make the following derivations.

   Remember we have shown in class:

   $$\frac{\partial l(\theta)}{\partial \theta_k} = \sum_{i=1}^{m} (y^{(i)} - h(x^{(i)}))x_k^{(i)} \tag{1}$$

[1] If you haven't seen this result before, please feel encouraged to ask us about it during office hours.

$$\begin{aligned} H_{kl} &= \frac{\partial^2 l(\theta)}{\partial\theta_k \partial\theta_l} && (2) \\ &= \sum_{i=1}^m -\frac{\partial h(x^{(i)})}{\partial\theta_l} x_k^{(i)} && (3) \\ &= \sum_{i=1}^m -h(x^{(i)})(1-h(x^{(i)}))x_l^{(i)}x_k^{(i)} && (4) \\ & && (5) \end{aligned}$$

So we have for the hessian matrix H (using that for $X = xx^T$if and only if$X_{ij} = x_i x_j$):

$$\begin{aligned} H &= -\sum_{i=1}^m h(x^{(i)})(1-h(x^{(i)}))x^{(i)}x^{(i)T} && (6) \\ & && (7) \end{aligned}$$

And to prove H is negative semidefinite, we show $z^T H z \leq 0$ for all z.

$$\begin{aligned} z^T H z &= -z^T \left(\sum_{i=1}^m h(x^{(i)})(1-h(x^{(i)}))x^{(i)}x^{(i)T} \right) z && (8) \\ &= -\sum_{i=1}^m h(x^{(i)})(1-h(x^{(i)}))z^T x^{(i)}x^{(i)T} z && (9) \\ &= -\sum_{i=1}^m h(x^{(i)})(1-h(x^{(i)}))(z^T x^{(i)})^2 && (10) \\ &\leq 0 && (11) \end{aligned}$$

with the last inequality holding, since $0 \leq h(x^{(i)}) \leq 1$, which implies $h(x^{(i)})(1-h(x^{(i)})) \geq 0$, and $(z^T x^{(i)})^2) \geq 0$.

(b) [10 points] On the Leland system, the files `/afs/ir/class/cs229/ps/ps1/q1x.dat` and `/afs/ir/class/cs229/ps/ps1/q1y.dat` contain the inputs ($x^{(i)} \in \mathbb{R}^2$) and outputs ($y^{(i)} \in \{0, 1\}$) respectively for a binary classification problem, with one training example per row. Implement[2] Newton's method for optimizing $\ell(\theta)$, and apply it to fit a logistic regression model to the data. Initialize Newton's method with $\theta = \vec{0}$ (the vector of all zeros). What are the coefficients θ resulting from your fit? (Remember to include the intercept term.)

Answer: $\theta = (-2.6205, 0.7604, 1.1719)$ with the first entry corresponding to the intercept term.

```
%%%%%%%   hw1q1.m  %%%%%%%
load('q1x.dat');
load('q1y.dat');

q1x = [ones(size(q1x,1),1) q1x];
[theta, ll] = log_regression(q1x,q1y);

m=size(q1x,1);
```

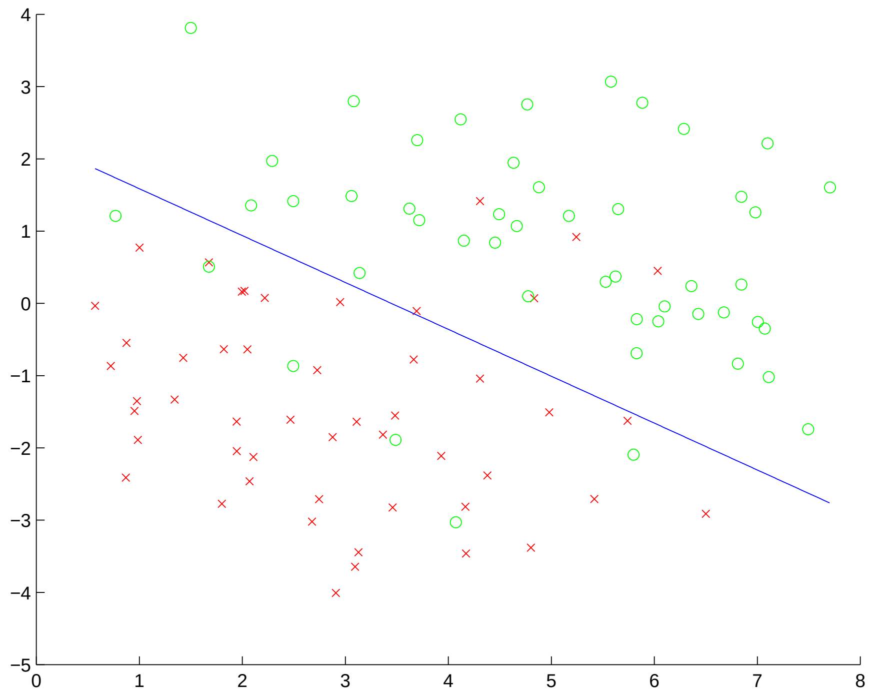
[2]Write your own version, and do not call a built-in library function.

```
figure; hold on;

for i=1:m
   if(q1y(i)==0)
      plot(q1x(i,2),q1x(i,3),'rx');
   else
      plot(q1x(i,2),q1x(i,3),'go');
   end
end

x = min(q1x(:,2)):.01:max(q1x(:,2));
y = -theta(1)/theta(3)-theta(2)/theta(3)*x;

plot(x,y);
xlabel('x1');
ylabel('x2');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%% log_regression.m %%%%%%%

function [theta,ll] = log_regression(X,Y)

% rows of X are training samples
% rows of Y are corresponding 0/1 values

% newton raphson: theta = theta - inv(H)* grad;
% with H = hessian, grad = gradient

m = size(X,1);
n = size(X,2);

theta = zeros(n,1);

max_iters = 50;
for i=1:max_iters
   grad = zeros(n,1);
   ll(i)=0;
   H = zeros(n,n);
   for j=1:m
      hxj = sigmoid(X(j,:)*theta);
      grad = grad + X(j,:)'*(Y(j) - hxj);
      H = H - hxj*(1-hxj)*X(j,:)'*X(j,:);
      ll(i) = ll(i) + Y(j)*log(hxj) + (1-Y(j))*log(1-hxj);
   end
   theta = theta - inv(H)*grad;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%% sigmoid.m %%%%%%%%%%%%%%
```

```
function a = sigmoid(x)

a = 1./(1+exp(-x));
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

(c) [5 points] Plot the training data (your axes should be x_1 and x_2, corresponding to the two coordinates of the inputs, and you should use a different symbol for each point plotted to indicate whether that example had label 1 or 0). Also plot on the same figure the decision boundary fit by logistic regression. (I.e., this should be a straight line showing the boundary separating the region where $h(x) > 0.5$ from where $h(x) \leq 0.5$.)

Answer:



2. **[27 points] Locally weighted linear regression**

Consider a linear regression problem in which we want to "weight" different training examples differently. Specifically, suppose we want to minimize

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m} w^{(i)} \left(\theta^T x^{(i)} - y^{(i)}\right)^2.$$

In class, we worked out what happens for the case where all the weights (the $w^{(i)}$'s) are the same. In this problem, we will generalize some of those ideas to the weighted setting, and also implement the locally weighted linear regression algorithm.

(a) [2 points] Show that $J(\theta)$ can also be written

$$J(\theta) = (X\theta - \vec{y})^T W (X\theta - \vec{y})$$

for an appropriate diagonal matrix W, and where X and $\vec{y}$ are as defined in class. State clearly what W is.

Answer: Let $W_{ii} = \frac{1}{2}w^{(i)}, W_{ij} = 0$ for $i \neq j$, let $\vec{z} = X\theta - \vec{y}$, i.e. $z_i = \theta^T x^{(i)} - y^{(i)}$. Then we have:

$$\begin{aligned}(X\theta - \vec{y})^T W (X\theta - \vec{y}) &= \vec{z}^T W \vec{z} && (12)\\ &= \frac{1}{2}\sum_{i=1}^{m} w^{(i)} z_i^2 && (13)\\ &= \frac{1}{2}\sum_{i=1}^{m} w^{(i)} (\theta^T x^{(i)} - y^{(i)})^2 && (14)\\ &= J(\theta) && (15)\end{aligned}$$

(b) [7 points] If all the $w^{(i)}$'s equal 1, then we saw in class that the normal equation is

$$X^T X\theta = X^T \vec{y},$$

and that the value of θ that minimizes $J(\theta)$ is given by $(X^T X)^{-1} X^T \vec{y}$. By finding the derivative $\nabla_\theta J(\theta)$ and setting that to zero, generalize the normal equation to this weighted setting, and give the new value of θ that minimizes $J(\theta)$ in closed form as a function of X, W and $\vec{y}$.

Answer:

$$\nabla_\theta J(\theta) = \nabla_\theta(\theta^T X^T W X\theta + \vec{y}^T W \vec{y} - 2\vec{y}^T W X \theta) = 2(X^T W X\theta - X^T W \vec{y}), \qquad (16)$$

so we have $\nabla_\theta J(\theta) = 0$ if and only if

$$X^T W X \theta = X^T W \vec{y} \qquad (17)$$

These are the normal equations, from which we can get a closed form formula for θ.

$$\theta = (X^T W X)^{-1} X^T W \vec{y} \qquad (18)$$

(c) [6 points] Suppose we have a training set $\{(x^{(i)}, y^{(i)});\ i = 1\ldots, m\}$ of m independent examples, but in which the $y^{(i)}$'s were observed with differing variances. Specifically, suppose that

$$p(y^{(i)}|x^{(i)};\theta) = \frac{1}{\sqrt{2\pi}\sigma^{(i)}} \exp\left(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2(\sigma^{(i)})^2}\right)$$

I.e., $y^{(i)}$ has mean $\theta^T x^{(i)}$ and variance $(\sigma^{(i)})^2$ (where the $\sigma^{(i)}$'s are fixed, known, constants). Show that finding the maximum likelihood estimate of θ reduces to solving a weighted linear regression problem. State clearly what the $w^{(i)}$'s are in terms of the $\sigma^{(i)}$'s.

Answer:

$$\begin{aligned}
\arg\max_\theta \prod_{i=1}^m p(y^{(i)}|x^{(i)};\theta) &= \arg\max_\theta \sum_{i=1}^m \log p(y^{(i)}|x^{(i)};\theta) && (19)\\
&= \arg\max_\theta \sum_{i=1}^m \left(\log \frac{1}{\sqrt{2\pi}\sigma^{(i)}} - \frac{(y^{(i)}-\theta^T x^{(i)})^2}{2(\sigma^{(i)})^2}\right) && (20)\\
&= \arg\max_\theta - \sum_{i=1}^m \frac{(y^{(i)}-\theta^T x^{(i)})^2}{2(\sigma^{(i)})^2} && (21)\\
&= \arg\min_\theta \frac{1}{2}\sum_{i=1}^m \frac{1}{(\sigma^{(i)})^2}(y^{(i)}-\theta^T x^{(i)})^2 && (22)\\
&= \arg\min_\theta \frac{1}{2}\sum_{i=1}^m w^{(i)}(y^{(i)}-\theta^T x^{(i)})^2 && (23)
\end{aligned}$$

where in the last step, we substituted: $w^{(i)} = \frac{1}{(\sigma^{(i)})^2}$ to get the linear regression form.

(d) [12 points] On the Leland computer system, the files `/afs/ir/class/cs229/ps/ps1/q2x.dat` and `/afs/ir/class/cs229/ps/ps1/q2y.dat` contain the inputs ($x^{(i)}$) and outputs ($y^{(i)}$) for a regression problem, with one training example per row.

i. [2 points] Implement (unweighted) linear regression ($y = \theta^T x$) on this dataset (using the normal equations), and plot on the same figure the data and the straight line resulting from your fit. (Remember to include the intercept term.)

ii. [7 points] Implement locally weighted linear regression on this dataset (using the weighted normal equations you derived in part (b)), and plot on the same figure the data and the curve resulting from your fit. When evaluating $h(\cdot)$ at a query point x, use weights

$$w^{(i)} = \exp\left(-\frac{(x-x^{(i)})^2}{2\tau^2}\right),$$

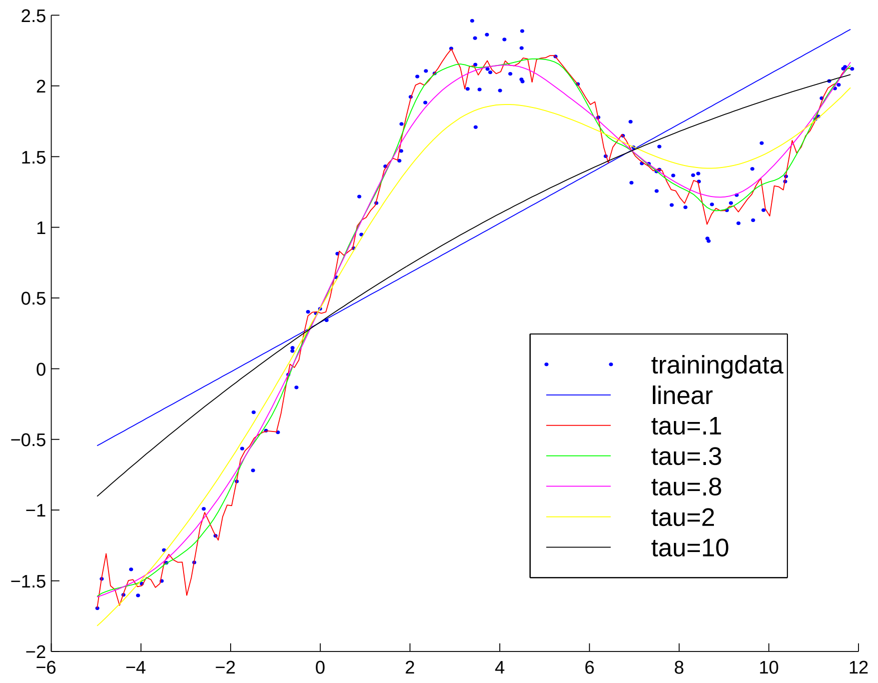
with a bandwidth parameter $\tau = 0.8$. (Again, remember to include the intercept term.)

iii. [3 points] Repeat (ii) four times, with $\tau = 0.1, 0.3, 2$ and 10. Comment **briefly** on what happens to the fit when τ is too small or too large.

Answer: Below is the code for all 3 parts of question 2d:

```
%%%%%% hw1q2d %%%%%%%

load('q2x.dat');
load('q2y.dat');

x = [ones(size(q2x,1),1) q2x];
y = q2y;

%% linear regression
theta = pinv(x'*x)*x'*y;

figure;
```

```
hold on;
plot(x(:,2),y,'.b');
regr_line_x = min(x(:,2)):.1:max(x(:,2));
regr_line_y = theta(2)*regr_line_x + theta(1);
plot(regr_line_x,regr_line_y,'b');




%% locally weighted linear regression

taus = [.1 .3 .8 2 10];
colors = ['r' 'g' 'm' 'y' 'k'];
m = size(q2x,1);

for i=1:size(taus,2)
   tau=taus(i);
   for k=1:size(regr_line_x,2)
      W = zeros(m,m);
      for l=1:m
         W(l,l)=exp(-(regr_line_x(k)-x(l,2))^2/(2*tau^2));
      end
      theta = pinv(x'*W*x)*x'*W*y;
      regr_line_y(k) = theta(2)*regr_line_x(k) + theta(1);
   end
   plot(regr_line_x,regr_line_y,colors(i));
end
legend('trainingdata','linear','tau=.1','tau=.3',...
    'tau=.8','tau=2','tau=10')
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```



(Plotted in color where available.)

For small bandwidth parameter τ, the fitting is dominated by the closest by training samples. The smaller the bandwidth, the less training samples that are actually taken into account when doing the regression, and the regression results thus become very susceptible to noise in those few training samples. For larger τ, we have enough training samples to reliably fit straight lines, unfortunately a straight line is not the right model for these data, so we also get a bad fit for large bandwidths.

3. **[18 points] Poisson regression and the exponential family**

   (a) [5 points] Consider the Poisson distribution parameterized by λ:

$$p(y;\lambda) = \frac{e^{-\lambda}\lambda^y}{y!}.$$

Show that the Poisson distribution is in the exponential family, and clearly state what are $b(y)$, η, $T(y)$, and $a(\eta)$.

Answer: Rewrite the distribution function as:

$$\begin{aligned} p(y;\lambda) &= \frac{e^{-\lambda}e^{y\log\lambda}}{y!} \\ &= \frac{1}{y!}\exp(y\log\lambda - \lambda) \end{aligned}$$

Comparing with the standard form for the exponential family:

$$\begin{aligned} b(y) &= \frac{1}{y!} \\ \eta &= \log\lambda \\ T(y) &= y \\ a(\eta) &= e^{\eta} \end{aligned}$$

(b) [3 points] Consider performing regression using a GLM model with a Poisson response variable. What is the canonical response function for the family? (You may use the fact that a Poisson random variable with parameter λ has mean λ.)

Answer: The canonical response function for the GLM model will be:

$$\begin{aligned} g(\eta) &= E[y;\eta] \\ &= \lambda \\ &= e^{\eta} \end{aligned}$$

(c) [10 points] For a training set $\{(x^{(i)}, y^{(i)}); i = 1, \dots, m\}$, let the log-likelihood of an example be $\log p(y^{(i)}|x^{(i)};\theta)$. By taking the derivative of the log-likelihood with respect to θ_j, derive the stochastic gradient ascent rule for learning using a GLM model with Poisson responses y and the canonical response function.

Answer: The log-likelihood of an example $(x^{(i)}, y^{(i)})$ is defined as $\ell(\theta) = \log p(y^{(i)}|x^{(i)};\theta)$. To derive the stochastic gradient ascent rule, use the results in part (a) and the standard GLM assumption that $\eta = \theta^T x$.

$$\begin{aligned} \frac{\partial \ell(\theta)}{\partial \theta_j} &= \frac{\partial \log p(y^{(i)}|x^{(i)};\theta)}{\partial \theta_j} \\ &= \frac{\partial \log\left(\frac{1}{y^{(i)}!}\exp(\eta^T y^{(i)} - e^{\eta})\right)}{\partial \theta_j} \\ &= \frac{\partial \log\left(\exp((\theta^T x^{(i)})^T y^{(i)} - e^{\theta^T x^{(i)}})\right)}{\partial \theta_j} + \frac{\partial \log\left(\frac{1}{y^{(i)}!}\right)}{\partial \theta_j} \\ &= \frac{\partial\left((\theta^T x^{(i)})^T y^{(i)} - e^{\theta^T x^{(i)}}\right)}{\partial \theta_j} \\ &= \frac{\partial\left((\sum_k \theta_k x_k^{(i)}) y^{(i)} - e^{\sum_k \theta_k x_k^{(i)}}\right)}{\partial \theta_j} \\ &= x_j^{(i)} y^{(i)} - e^{\sum_k \theta_k x_k^{(i)}} x_j^{(i)} \\ &= (y^{(i)} - e^{\theta^T x^{(i)}}) x_j^{(i)} \end{aligned}$$

Thus the stochastic gradient ascent update rule should be:

$$\theta_j := \theta_j + \alpha \frac{\partial \ell(\theta)}{\partial \theta_j}$$

which reduces here to:

$$\theta_j := \theta_j + \alpha(y^{(i)} - e^{\theta^T x})x_j^{(i)}$$

(d) **[5 extra credit points]** Consider using GLM with a response variable from any member of the exponential family in which $T(y) = y$, and the canonical response function for the family. Show that stochastic gradient ascent on the log-likelihood $\log p(\vec{y}|X,\theta)$ results in the update rule $\theta_i := \theta_i - \alpha(h(x) - y)x_i$.

Answer: As in the previous part, consider the derivative of the likelihood of a training example (x, y) with respect to the parameter θ_j:

$$\begin{aligned}
\frac{\partial \ell(\theta)}{\partial \theta_j} &= \frac{\partial \log p(y|x;\theta)}{\partial \theta_j} \\
&= \frac{\partial \log \left(b(y)\exp(\eta^T y - a(\eta))\right)}{\partial \theta_j} \\
&= \frac{\partial \left(\eta^T y - a(\eta))\right)}{\partial \theta_j} \\
&= x_j y - \frac{\partial a(\eta)}{\partial \eta} x_j \\
&= \left(y - \frac{\partial a(\eta)}{\partial \eta}\right) x_j
\end{aligned}$$

Thus, it only remains to show that $\frac{\partial a(\eta)}{\partial \eta} = h(x) = E[y|x;\theta]$. To prove this consider the fact that $p(y|x;\theta)$ is a probability distribution and must thus sum to 1.

$$\begin{aligned}
\int_y p(y|x;\theta)dy &= 1 \\
\int_y b(y)\exp(\eta^T y - a(\eta))dy &= 1 \\
\int_y b(y)\exp(\eta^T y)dy &= \exp(a(\eta))
\end{aligned}$$

Differentiating both sides with respect to η:

$$\begin{aligned}
\int_y b(y) y \exp(\eta^T y)dy &= \exp(a(\eta))\frac{\partial a(\eta)}{\partial \eta} \\
\frac{\partial a(\eta)}{\partial \eta} &= \int_y b(y) y \exp(\eta^T y - a(\eta))dy \\
&= \int_y y p(y|x;\theta)dy \\
&= E[y|x;\theta]
\end{aligned}$$

where the last step follows from the definition of the (conditional) expectation of a random variable. Substituting this into the expression for $\frac{\partial \ell(\theta)}{\partial \theta_j}$ gives the required gradient ascent update rule.

4. **[15 points] Gaussian discriminant analysis**

Suppose we are given a dataset $\{(x^{(i)}, y^{(i)}); i = 1, \ldots, m\}$ consisting of m independent examples, where $x^{(i)} \in \mathbb{R}^n$ are n-dimensional vectors, and $y^{(i)} \in \{0, 1\}$. We will model the joint distribution of (x, y) according to:

$$\begin{aligned}
p(y) &= \phi^y (1-\phi)^{1-y} \\
p(x|y=0) &= \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_0)^T\Sigma^{-1}(x-\mu_0)\right) \\
p(x|y=1) &= \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1)\right)
\end{aligned}$$

Here, the parameters of our model are ϕ, Σ, μ_0 and μ_1. (Note that while there're two different mean vectors μ_0 and μ_1, there's only one covariance matrix Σ.)

(a) [5 points] Suppose we have already fit ϕ, Σ, μ_0 and μ_1, and now want to make a prediction at some new query point x. Show that the posterior distribution of the label at x takes the form of a logistic function, and can be written

$$p(y=1|x;\phi,\Sigma,\mu_0,\mu_1) = \frac{1}{1+\exp(-\theta^T x)},$$

where θ is some appropriate function of $\phi, \Sigma, \mu_0, \mu_1$. (Note: To get your answer into the form above, for this part of the problem only, you may have to redefine the $x^{(i)}$'s to be $n+1$-dimensional vectors by adding the extra coordinate $x_0^{(i)} = 1$, like we did in class.)

Answer: Since the given formulae are conditioned on y, use Bayes rule to get:

$$\begin{aligned}
p(y=1|x;\phi,\Sigma,\mu_0,\mu_1) &= \frac{p(x|y=1;\phi,\Sigma,\mu_0,\mu_1)p(y=1;\phi,\Sigma,\mu_0,\mu_1)}{p(x;\phi,\Sigma,\mu_0,\mu_1)} \\
&= \frac{p(x|y=1;\ldots)p(y=1;\ldots)}{p(x|y=1;\ldots)p(y=1;\ldots)+p(x|y=0;\ldots)p(y=0;\ldots)} \\
&= \frac{\exp\left(-\frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1)\right)\phi}{\exp\left(-\frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1)\right)\phi+\exp\left(-\frac{1}{2}(x-\mu_0)^T\Sigma^{-1}(x-\mu_0)\right)(1-\phi)} \\
&= \frac{1}{1+\frac{1-\phi}{\phi}\exp\left(-\frac{1}{2}(x-\mu_0)^T\Sigma^{-1}(x-\mu_0)+\frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1)\right)} \\
&= \frac{1}{1+\exp\left(\log(\frac{1-\phi}{\phi})-\frac{1}{2}(x-\mu_0)^T\Sigma^{-1}(x-\mu_0)+\frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1)\right)} \\
&= \frac{1}{1+\exp\left(-\frac{1}{2}\left(-2\mu_0^T\Sigma^{-1}x+\mu_0^T\Sigma^{-1}\mu_0+2\mu_1^T\Sigma^{-1}x-\mu_1^T\Sigma^{-1}\mu_1\right)+\log(\frac{1-\phi}{\phi})\right)}
\end{aligned}$$

where we have simplified the denominator in the penultimate step by expansion, i.e.,

$$\begin{aligned}
&-\frac{1}{2}(x-\mu_0)^T\Sigma^{-1}(x-\mu_0)+\frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1) \\
&= -\frac{1}{2}\left(x^T\Sigma^{-1}x-\mu_0^T\Sigma^{-1}x-x^T\Sigma^{-1}\mu_0+\mu_0^T\Sigma^{-1}\mu_0-x^T\Sigma^{-1}x+\mu_1^T\Sigma^{-1}x+x^T\Sigma^{-1}\mu_1-\mu_1^T\Sigma^{-1}\mu_1\right) \\
&= -\frac{1}{2}\left(\mu_0^T\Sigma^{-1}x-(x^T\Sigma^{-1}\mu_0)^T+\mu_0^T\Sigma^{-1}\mu_0+\mu_1^T\Sigma^{-1}x+(x^T\Sigma^{-1}\mu_1)^T-\mu_1^T\Sigma^{-1}\mu_1\right)
\end{aligned}$$

Recall that the question was to find θ for $\exp(-\theta^T x)$, after adding a constant intercept term $x_0 = 1$, we have θ equal to:

$$\begin{bmatrix} \frac{1}{2}(\mu_0^T\Sigma^{-1}\mu_0 - \mu_1^T\Sigma^{-1}\mu_1) - \log(\frac{1-\phi}{\phi}) \\ \Sigma^{-1}\mu_1 - \Sigma^{-1}\mu_0 \end{bmatrix}$$

(b) [10 points] For this part of the problem only, you may assume n (the dimension of x) is 1, so that $\Sigma = [\sigma^2]$ is just a real number, and likewise the determinant of Σ is given by $|\Sigma| = \sigma^2$. Given the dataset, we claim that the maximum likelihood estimates of the parameters are given by

$$\begin{aligned} \phi &= \frac{1}{m}\sum_{i=1}^m 1\{y^{(i)} = 1\} \\ \mu_0 &= \frac{\sum_{i=1}^m 1\{y^{(i)} = 0\}x^{(i)}}{\sum_{i=1}^m 1\{y^{(i)} = 0\}} \\ \mu_1 &= \frac{\sum_{i=1}^m 1\{y^{(i)} = 1\}x^{(i)}}{\sum_{i=1}^m 1\{y^{(i)} = 1\}} \\ \Sigma &= \frac{1}{m}\sum_{i=1}^m (x^{(i)} - \mu_{y^{(i)}})(x^{(i)} - \mu_{y^{(i)}})^T \end{aligned}$$

The log-likelihood of the data is

$$\begin{aligned} \ell(\phi, \mu_0, \mu_1, \Sigma) &= \log \prod_{i=1}^m p(x^{(i)}, y^{(i)}; \phi, \mu_0, \mu_1, \Sigma) \\ &= \log \prod_{i=1}^m p(x^{(i)}|y^{(i)}; \mu_0, \mu_1, \Sigma)p(y^{(i)}; \phi). \end{aligned}$$

By maximizing ℓ with respect to the four parameters, prove that the maximum likelihood estimates of ϕ, μ_0, μ_1, and Σ are indeed as given in the formulas above. (You may assume that there is at least one positive and one negative example, so that the denominators in the definitions of μ_0 and μ_1 above are non-zero.)

Answer: The derivation follows from the more general one for the next part.

(c) **[5 extra credit points]** Without assuming that $n = 1$, show that the maximum likelihood estimates of ϕ, μ_0, μ_1, and Σ are as given in the formulas in part (b). [Note: If you're fairly sure that you have the answer to this part right, you don't have to do part (b), since that's just a special case.]

Answer: First, derive the expression for the log-likelihood of the training data:

$$\begin{aligned} \ell(\phi, \mu_0, \mu_1, \Sigma) &= \log \prod_{i=1}^m p(x^{(i)}|y^{(i)}; \mu_0, \mu_1, \Sigma)p(y^{(i)}; \phi) \\ &= \sum_{i=1}^m \log p(x^{(i)}|y^{(i)}; \mu_0, \mu_1, \Sigma) + \sum_{i=1}^m \log p(y^{(i)}; \phi) \\ &\simeq \sum_{i=1}^m \left[\frac{1}{2}\log\frac{1}{|\Sigma|} - \frac{1}{2}(x^{(i)} - \mu_{y^{(i)}})^T\Sigma^{-1}(x^{(i)} - \mu_{y^{(i)}}) + y^{(i)}\log\phi + (1 - y^{(i)})\log(1-\phi)\right] \end{aligned}$$

where constant terms indepedent of the parameters have been ignored in the last expression.

Now, the likelihood is maximized by setting the derivative (or gradient) with respect to each of the parameters to zero.

$$\begin{aligned}
\frac{\partial \ell}{\partial \phi} &= \sum_{i=1}^{m} [\frac{y^{(i)}}{\phi} - \frac{1-y^{(i)}}{1-\phi}] \\
&= \frac{\sum_{i=1}^{m} 1\{y^{(i)}=1\}}{\phi} - \frac{m - \sum_{i=1}^{m} 1\{y^{(i)}=1\}}{1-\phi}
\end{aligned}$$

Setting this equal to zero and solving for ϕ gives the maximum likelihood estimate.

For μ_0, take the gradient of the log-likelihood, and then use the same kinds of tricks as were used to analytically solve the linear regression problem.

$$\begin{aligned}
\nabla_{\mu_0} \ell &= -\frac{1}{2} \sum_{i:y^{(i)}=0} \nabla_{\mu_0} (x^{(i)} - \mu_0)^T \Sigma^{-1} (x^{(i)} - \mu_0) \\
&= -\frac{1}{2} \sum_{i:y^{(i)}=0} \nabla_{\mu_0} \left[\mu_0^T \Sigma^{-1} \mu_0 - {x^{(i)}}^T \Sigma^{-1} \mu_0 - \mu_0^T \Sigma^{-1} x^{(i)}\right] \\
&= -\frac{1}{2} \sum_{i:y^{(i)}=0} \nabla_{\mu_0} tr \left[\mu_0^T \Sigma^{-1} \mu_0 - {x^{(i)}}^T \Sigma^{-1} \mu_0 - \mu_0^T \Sigma^{-1} x^{(i)}\right] \\
&= -\frac{1}{2} \sum_{i:y^{(i)}=0} \left[2\Sigma^{-1} \mu_0 - 2\Sigma^{-1} x^{(i)}\right]
\end{aligned}$$

The last step uses matrix calculus identities (specifically, those given in page 8 of the lecture notes), and also the fact that Σ (and thus Σ^{-1}) is symmetric.

Setting this gradient to zero gives the maximum likelihood estimate for μ_0. The derivation for μ_1 is similar to the one above.

For Σ, we find the gradient with respect to $S = \Sigma^{-1}$ rather than Σ just to simplify the derivation (note that $|S| = \frac{1}{|\Sigma|}$). You should convince yourself that the maximum likelihood estimate S_m found in this way would correspond to the actual maximum likelihood estimate Σ_m as $S_m^{-1} = \Sigma_m$.

$$\begin{aligned}
\nabla_S \ell &= \sum_{i=1}^{m} \nabla_S [\frac{1}{2} \log |S| - \frac{1}{2} \underbrace{(x^{(i)} - \mu_{y^{(i)}})^T}_{b_i^T} S \underbrace{(x^{(i)} - \mu_{y^{(i)}})}_{b_i}] \\
&= \sum_{i=1}^{m} [\frac{1}{2|S|} \nabla_S |S| - \frac{1}{2} \nabla_S b_i^T S b_i]
\end{aligned}$$

But, we have the following identities:

$$\nabla_S |S| = |S| (S^{-1})^T$$

$$\nabla_S b_i^T S b_i = \nabla_S tr\left(b_i^T S b_i\right) = \nabla_S tr\left(S b_i b_i^T\right) = b_i b_i^T$$

In the above, we again used matrix calculus identities, and also the commutatitivity of the trace operator for square matrices. Putting these into the original equation, we get:

$$\begin{aligned}\nabla_S \ell &= \sum_{i=1}^{m} \left[\frac{1}{2}S^{-1} - \frac{1}{2}b_i b_i^T\right] \\ &= \frac{1}{2}\sum_{i=1}^{m} \left[\Sigma - b_i b_i^T\right]\end{aligned}$$

Setting this to zero gives the required maximum likelihood estimate for Σ.

5. **[12 points] Linear invariance of optimization algorithms**

Consider using an iterative optimization algorithm (such as Newton's method, or gradient descent) to minimize some continuously differentiable function $f(x)$. Suppose we initialize the algorithm at $x^{(0)} = \vec{0}$. When the algorithm is run, it will produce a value of $x \in \mathbb{R}^n$ for each iteration: $x^{(1)}, x^{(2)}, \ldots$.

Now, let some non-singular square matrix $A \in \mathbb{R}^{n\times n}$ be given, and define a new function $g(z) = f(Az)$. Consider using the same iterative optimization algorithm to optimize g (with initialization $z^{(0)} = \vec{0}$). If the values $z^{(1)}, z^{(2)}, \ldots$ produced by this method necessarily satisfy $z^{(i)} = A^{-1}x^{(i)}$ for all i, we say this optimization algorithm is **invariant to linear reparameterizations**.

(a) [9 points] Show that Newton's method (applied to find the minimum of a function) is invariant to linear reparameterizations. Note that since $z^{(0)} = \vec{0} = A^{-1}x^{(0)}$, it is sufficient to show that if Newton's method applied to $f(x)$ updates $x^{(i)}$ to $x^{(i+1)}$, then Newton's method applied to $g(z)$ will update $z^{(i)} = A^{-1}x^{(i)}$ to $z^{(i+1)} = A^{-1}x^{(i+1)}$.[3]

Answer: Let $g(z) = f(Az)$. We need to find $\nabla_z g(z)$ and its Hessian $\nabla_z^2 g(z)$.

By the chain rule:

$$\begin{aligned}\frac{\partial g(z)}{\partial z_i} &= \sum_{k=1}^{n} \frac{\partial f(Az)}{\partial (Az)_k}\frac{\partial (Az)_k}{\partial z_i} && (24)\\ &= \sum_{k=1}^{n} \frac{\partial f(Az)}{\partial (Az)_k} A_{ki} && (25)\\ &= \sum_{k=1}^{n} \frac{\partial f(Az)}{\partial x_k} A_{ki} && (26)\end{aligned}$$

Notice that the above is the same as :

$$\frac{\partial g(z)}{\partial z_i} = A_{\bullet i}^{\top}\nabla_x f(Az) \qquad (27)$$

where $A_{\bullet i}$ is the i'th column of A. Then,

$$\nabla_z g(z) = A^{\top}\nabla_x f(Az) \qquad (28)$$

[3]Note that for this problem, you must explicitly prove any matrix calculus identities that you wish to use that are not given in the lecture notes.

where $\nabla_x f(Az)$ is $\nabla_x f(\cdot)$ evaluated at Az.

Now we want to find the Hessian $\nabla_z^2 g(z)$.

$$\begin{aligned}\frac{\partial^2 g(z)}{\partial z_i \partial z_j} &= \frac{\partial}{\partial z_j}\sum_{k=1}^{n}\frac{\partial f(Az)}{\partial (Az)_k}A_{ki} && (29)\\ &= \sum_l\sum_k \frac{\partial^2 f(Az)}{\partial x_l \partial x_k}A_{ki}A_{lj} && (30)\end{aligned}$$

If we let $H_f(y)$ denote the Hessian of $f(\cdot)$ evaluated at some point y, and let $H_g(y)$ be the Hessian of $g(\cdot)$ evaluated at some point y, we have from the previous equation that:

$$H_g(z) = A^\top H_f(Az)A \tag{31}$$

We can now put this together and find the update rule for Newton's method on the function $f(Ax)$:

$$\begin{aligned} z^{(i+1)} &= z^{(i)} - H_g(z^{(i)})^{-1}\nabla_z g(z^{(i)}) && (32)\\ &= z^{(i)} - (A^\top H_f(Az^{(i)})A)^{-1}A^\top \nabla_x f(Az^{(i)}) && (33)\\ &= z^{(i)} - A^{-1}H_f(Az^{(i)})^{-1}(A^\top)^{-1}A^\top\nabla_x f(Az^{(i)}) && (34)\\ &= z^{(i)} - A^{-1}H_f(Az^{(i)})^{-1}\nabla_x f(Az^{(i)}) && (35)\end{aligned}$$

Now we have the update rule for $z^{(i+1)}$, we just need to verify that $z^{(i+1)} = A^{-1}x^{(i+1)}$ or equivalently that $Az^{(i+1)} = x^{(i+1)}$. From Eqn. (**??**) we have

$$\begin{aligned} Az^{(i+1)} &= A\left(z^{(i)} - A^{-1}H_f(Az^{(i)})^{-1}\nabla_x f(Az^{(i)})\right) && (36)\\ &= Az^{(i)} - H_f(Az^{(i)})^{-1}\nabla_x f(Az^{(i)}) && (37)\\ &= x^{(i)} - H_f(x^{(i)})^{-1}\nabla_x f(x^{(i)}) && (38)\\ &= x^{(i+1)}, && (39)\end{aligned}$$

where we used in order: Eqn. (**??**); rewriting terms; the inductive assumption $x^{(i)} = Az^{(i)}$; the update rule $x^{(i+1)} = x^{(i)} - H_f(x^{(i)})^{-1}\nabla_x f(x^{(i)})$.

(b) [3 points] Is gradient descent invariant to linear reparameterizations? Justify your answer.

Answer:

No. Using the notation from above, gradient descent on $g(z)$ results in the following update rule:

$$z^{(i+1)} = z^{(i)} - \alpha A^\top \nabla_x f(Az^{(i)}). \tag{40}$$

The update rule for $x^{(i+1)}$ is given by

$$x^{(i+1)} = x^{(i)} - \alpha\nabla_x f(x^{(i)}). \tag{41}$$

The invariance holds if and only if $x^{(i+1)} = Az^{(i+1)}$ given $x^{(i)} = Az^{(i)}$. However we have

$$\begin{aligned} Az^{(i+1)} &= Az^{(i)} - \alpha AA^\top \nabla_x f(Az^{(i)}) && (42) \\ &= x^{(i)} - \alpha AA^\top \nabla_x f(x^{(i)}). && (43) \end{aligned}$$

The two expressions in Eqn. (??) and Eqn. (??) are not necessarily equal ($AA^T = I$ requires that A be an orthogonal matrix), and thus gradient descent is not invariant to linear reparameterizations.

Reminder: Please include in your submission a printout of your code and figures for the programming questions.