ECE 403 June 23, 2020 Ashlynn Steeves
Assignment 4 V00850631

All MATLAB code used to generate answers for this assignment can be found at
https://github.com/ashlynns/ECE403/tree/master/A4

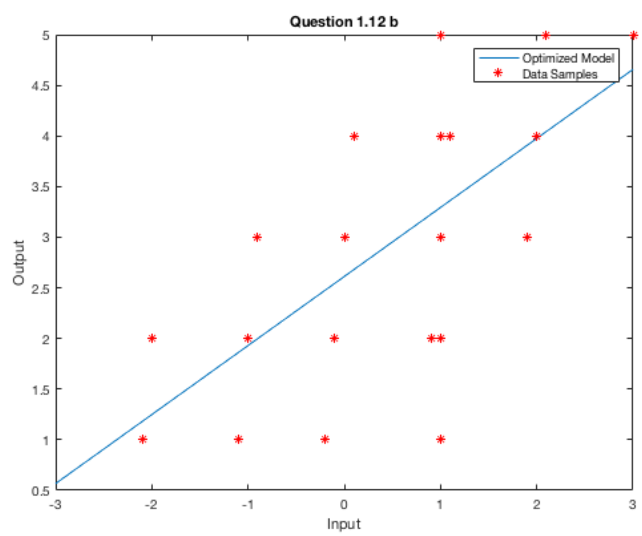
Q1.12

(a) Apply least-squares linear regression to the data to generate a prediction model

```
X = [-2.1 -2 -1.1 -1 -0.9 -0.2 -0.1 0 0.1 0.9 1 1.1 1.9 2 2.1 3];
y = [1 2 1 2 3 1 2 3 4 2 3 4 3 4 5 5]';

X = [X ; ones(1, length(X))];
w = pinv(X')*y;
```

(b) In a single figure, plot the sixteen data samples and the optimized model as a straight line in the x-y space.



(c) Use the model obtained to predict the output for input x = 1.5.

Using the model to predict the output for input x=1.5 returns a y value of y = 3.634.

Q2.1

(a) Follow the above discussion to define a softmax cost function in terms of w^ and z_p defined in (P2.7). Hint: Eq. (1.49) of the Notes provides a counterpart of the cost function in the original input space.

$$E_L(\hat{\boldsymbol{w}}) = \frac{1}{P}\sum_{p=1}^{P} \log\left(1 + e^{-y_p \hat{\mathbf{z}}_p^T \hat{\boldsymbol{w}}}\right)$$

(b) Derive a formula for the gradient of $E_{LR}(\hat{\boldsymbol{w}})$

$$E_{LR}(\hat{\boldsymbol{w}}) = \frac{1}{P}\sum_{p=1}^{P} \log\left(1 + e^{-y_p \hat{\mathbf{z}}_p^T \hat{\boldsymbol{w}}}\right) + \frac{\mu}{2}\|\hat{\boldsymbol{w}}\|_2^2$$

$$\nabla E_{LR}(\hat{\boldsymbol{w}}) = \frac{1}{P}\sum_{p=1}^{P} \nabla\hat{\boldsymbol{w}}(\log\left(1 + e^{-y_p \hat{\mathbf{z}}_p^T \hat{\boldsymbol{w}}}\right)) + \frac{\mu}{2}\nabla\|\hat{\boldsymbol{w}}\|_2^2$$

$$\nabla E_{LR}(\hat{\boldsymbol{w}}) = \frac{1}{P}\sum_{p=1}^{P} \frac{1}{1 + e^{-y_p \hat{\mathbf{z}}_p^T \hat{\boldsymbol{w}}}}\nabla\hat{\boldsymbol{w}}(e^{-y_p \hat{\mathbf{z}}_p^T \hat{\boldsymbol{w}}}) + \mu\hat{\boldsymbol{w}}$$

$$\nabla E_{LR}(\hat{\boldsymbol{w}}) = \frac{1}{P}\sum_{p=1}^{P} \frac{e^{-y_p \hat{\mathbf{z}}_p^T \hat{\boldsymbol{w}}}}{1 + e^{-y_p \hat{\mathbf{z}}_p^T \hat{\boldsymbol{w}}}}(-y_p \hat{\boldsymbol{z}}_p) + \mu\hat{\boldsymbol{w}}$$

$$\nabla E_{LR}(\hat{\boldsymbol{w}}) = -\frac{1}{P}\sum_{p=1}^{P} \frac{y_p \hat{\boldsymbol{z}}_p}{1 + e^{y_p \hat{\mathbf{z}}_p^T \hat{\boldsymbol{w}}}} + \mu\hat{\boldsymbol{w}}$$

(c) Write a MATLAB function for $E_{LR}(\hat{\boldsymbol{w}})$ and $\nabla E_{LR}(\hat{\boldsymbol{w}})$

```
function [E_lr_out] = E_lr(w)
   X = [-1 0 2 0 1 2 ; -1 1 0 -2 0 -1];
   y = [1 1 1 -1 -1 -1]';
   z = zeros(5, length(X));
   for i = 1:length(X)
       x1 = X(1,i);
       x2 = X(2,i);

       z(1,i) = x1;
       z(2,i) = x2;
       z(3,i) = x1^2;
       z(4,i) = x1*x2;
       z(5,i) = x2^2;
   end
   w_padded = repmat(w,1,length(X))
   zdw = dot(z, w_padded, 1)
   exp_val = exp((-y)'.*zdw)
   sum_val = sum(log(1+exp_val))
   E_lr_out = sum_val/length(X)
end
```

```
function [E_lr_grad_out] = E_lr_grad(w)
   X = [-1 0 2 0 1 2 ; -1 1 0 -2 0 -1];
   y = [1 1 1 -1 -1 -1]';
   u = 0.03;
   z = zeros(5, length(X));
   for i = 1:length(X)
       x1 = X(1,i);
       x2 = X(2,i);

       z(1,i) = x1;
       z(2,i) = x2;
       z(3,i) = x1^2;
       z(4,i) = x1*x2;
       z(5,i) = x2^2;
   end
   w_padded = repmat(w,1,length(X));
   zdw = dot(z, w_padded, 1);

   exp_val = exp((y)'.*zdw);
   denom = 1+exp_val; % Denominator of sum term
   num = y'.*z; % Numerator of sum term
   sum_val = sum(num./denom, 2);
   norm = sum_val./-(length(X)); % Sum term normalized by -P
   uw = u*w; % Addition term

   E_lr_grad_out = norm+uw;
end
```

Q2.2

(a) Apply the basic GD to minimize ELR with input ***w****= 0, u = 0.03, and e = 10^-6. Report the numerical result of the minimizer* ***w*** *obtained. Use the optimized model to classify new input samples.*

$$\hat{\boldsymbol{w}}^* = [-1.401, 1.4706, 1.0532, 1.2800, 0.0100]'$$

x_tr_1 is classified as -1

x_tr_1 is classified as 1

(a) Apply momentum-accelerated GD to minimize ELR with input ***w****= 0, u = 0.03, and e = 10^-6. Report the numerical result of the minimizer* ***w*** *obtained. Use the optimized model to classify new input samples.*

Running the provided gd_momentum script under these conditions was not yielding results in a reasonable amount of time (i.e. was not converging). I have tried running it with different conditions such as a smaller epsilon and smaller beta value and was able to run the scrip efficiently using e = 10^-6 and b = 0.75. Which yields the results below (same as a).

$$\hat{\boldsymbol{w}}^* = [-1.401, 1.4706, 1.0532, 1.2800, 0.0100]'$$

x_tr_1 is classified as -1

x_tr_1 is classified as 1