Computer Science 145, Homework 1

Michael Wu
UID: 404751542

January 25th, 2019

1 Linear Regression

Problem 1

a) For the closed form solution without normalization, my results were as follows.

```
Beta:  [ 5.17285600e-01 -1.70173005e-02 -1.25229040e-02 -2.37364105e-02
 -7.26850224e-03 -1.75015833e-03 -2.54105104e-02 -2.95826147e-02
 -1.54205779e-02 -8.11041550e-03 -7.73115897e-03  2.23326880e-02
  1.08377404e-02 -1.78034252e-02 -1.42799326e-02  1.10863872e-03
  5.58962376e-03 -1.66481000e-02  1.74175345e-02 -9.79321083e-03
 -1.13160627e-02 -2.43621012e-02  1.22042094e-02 -2.22008980e-02
 -8.08868427e-03  1.98280275e-02 -1.62549170e-02  1.57163255e-02
  5.55093555e-03  2.52723067e-02 -1.79696813e-02 -3.42412589e-02
  2.33967228e-02 -1.18951150e-02 -8.29832518e-03  1.08683008e-03
  1.07503176e-02  5.89595929e-03 -1.42884432e-02 -7.60366278e-03
 -3.59068468e-03 -2.43039502e-02 -1.50352102e-02 -4.91648480e-05
 -1.75975159e-02 -5.12186137e-03 -6.03505757e-03  2.11963730e-03
  1.84672144e-02  5.97564034e-03  7.70482473e-03 -1.32971032e-02
 -1.56211468e-02  1.64262479e-02 -1.87298040e-02 -2.62080745e-02
  1.98841713e-02 -2.47382511e-02  7.11668306e-03 -2.56090472e-02
 -1.43803106e-02 -1.78350545e-02 -2.34158378e-02 -1.21549137e-02
  2.26194590e-02 -1.35242391e-02  8.88066425e-04 -1.42204055e-02
  2.99114634e-03  5.22524532e-03 -1.79063948e-02  3.83684473e-03
  8.33356729e-03  2.56888081e-02 -1.80756710e-02 -1.99695440e-02
 -2.86138337e-02  2.35867028e-02  1.90433998e-03  1.72159943e-02
  3.03296234e-02  1.74398815e-02 -2.78753941e-02  1.30140929e-02
  2.60430914e-02 -2.59504768e-04  1.74699574e-02  3.43722771e-05
  1.37552942e-02  2.24646356e-02 -1.22617221e-02 -1.82281224e-02
  1.80041301e-02 -7.43819418e-04 -2.84486814e-02 -1.42173525e-02
 -9.10220722e-04 -2.59410878e-02  1.86651575e-02  2.90379883e-02
 -1.63292879e-03]
Training MSE:  0.08693886675396784
Test MSE:  0.11017540281675804
```

For the batch gradient descent without normalization, my results were as follows.

```
Beta:  [ 5.17285600e-01 -1.70173005e-02 -1.25229040e-02 -2.37364105e-02
 -7.26850224e-03 -1.75015833e-03 -2.54105104e-02 -2.95826147e-02
 -1.54205779e-02 -8.11041550e-03 -7.73115897e-03  2.23326880e-02
  1.08377404e-02 -1.78034252e-02 -1.42799326e-02  1.10863872e-03
  5.58962376e-03 -1.66481000e-02  1.74175345e-02 -9.79321083e-03
 -1.13160627e-02 -2.43621012e-02  1.22042094e-02 -2.22008980e-02
 -8.08868427e-03  1.98280275e-02 -1.62549170e-02  1.57163255e-02
  5.55093555e-03  2.52723067e-02 -1.79696813e-02 -3.42412589e-02
  2.33967228e-02 -1.18951150e-02 -8.29832518e-03  1.08683008e-03
  1.07503176e-02  5.89595929e-03 -1.42884432e-02 -7.60366278e-03
 -3.59068468e-03 -2.43039502e-02 -1.50352102e-02 -4.91648480e-05
 -1.75975159e-02 -5.12186137e-03 -6.03505757e-03  2.11963730e-03
  1.84672144e-02  5.97564034e-03  7.70482473e-03 -1.32971032e-02
 -1.56211468e-02  1.64262479e-02 -1.87298040e-02 -2.62080745e-02
  1.98841713e-02 -2.47382511e-02  7.11668306e-03 -2.56090472e-02
 -1.43803106e-02 -1.78350545e-02 -2.34158378e-02 -1.21549137e-02
  2.26194590e-02 -1.35242391e-02  8.88066424e-04 -1.42204055e-02
  2.99114634e-03  5.22524533e-03 -1.79063948e-02  3.83684473e-03
  8.33356729e-03  2.56888081e-02 -1.80756710e-02 -1.99695440e-02
 -2.86138337e-02  2.35867028e-02  1.90433998e-03  1.72159943e-02
  3.03296234e-02  1.74398815e-02 -2.78753941e-02  1.30140929e-02
  2.60430914e-02 -2.59504767e-04  1.74699574e-02  3.43722776e-05
  1.37552942e-02  2.24646356e-02 -1.22617221e-02 -1.82281224e-02
  1.80041301e-02 -7.43819418e-04 -2.84486814e-02 -1.42173525e-02
 -9.10220721e-04 -2.59410878e-02  1.86651575e-02  2.90379883e-02
 -1.63292879e-03]
Training MSE:  0.08693886675396784
Test MSE:  0.1101754028154542
```

For the stochastic gradient descent without normalization, my results were as follows.

```
Beta:  [ 5.17727561e-01 -1.47842815e-02 -1.20676526e-02 -2.37668895e-02
 -7.32783328e-03 -1.41445479e-03 -2.55251396e-02 -2.93908237e-02
 -1.38775844e-02 -8.59431089e-03 -6.82611718e-03  2.25239053e-02
  1.05100434e-02 -1.66175929e-02 -1.38122374e-02  8.97821998e-04
  5.64471210e-03 -1.77903286e-02  1.69915916e-02 -1.01240826e-02
 -1.14891240e-02 -2.46543082e-02  1.28926195e-02 -2.26955305e-02
 -9.12683142e-03  2.00863340e-02 -1.66615275e-02  1.64042064e-02
  6.26488259e-03  2.55886526e-02 -1.83338710e-02 -3.53007669e-02
  2.39224157e-02 -1.15924739e-02 -9.76131677e-03  1.62107461e-03
  1.10138185e-02  5.47313008e-03 -1.39561895e-02 -7.81630175e-03
 -3.66006941e-03 -2.50854327e-02 -1.57812687e-02  4.63354295e-04
 -1.77176446e-02 -6.30789938e-03 -5.99915938e-03  2.55699483e-03
  1.96424979e-02  5.77555691e-03  7.84005872e-03 -1.28987910e-02
 -1.56818298e-02  1.72374710e-02 -1.86664401e-02 -2.71262493e-02
  2.04028613e-02 -2.44512076e-02  8.10172656e-03 -2.57879568e-02
 -1.51164302e-02 -1.85877657e-02 -2.31486026e-02 -1.13138019e-02
  2.36709941e-02 -1.45172241e-02  1.82397095e-03 -1.32906005e-02
  3.89949952e-03  4.17891060e-03 -1.81996929e-02  5.00152068e-03
  9.03659874e-03  2.37585695e-02 -1.85252329e-02 -1.91261803e-02
 -2.81076966e-02  2.35929552e-02  1.30670030e-03  1.72969880e-02
  3.02645140e-02  1.76422405e-02 -2.72848722e-02  1.34057762e-02
  2.70686451e-02 -6.99976354e-04  1.73402942e-02 -7.75413623e-04
```

```
  1.50461318e-02  2.31099983e-02 -1.14892402e-02 -1.68788189e-02
  1.73091842e-02  4.88273154e-04 -2.86047643e-02 -1.38073724e-02
 -2.96093485e-04 -2.70094076e-02  1.90252359e-02  2.94424674e-02
 -1.63801153e-03]
Training MSE:  0.08727435764353829
Test MSE:  0.11001483005151429
```

They are not the same in each version, though they are all pretty close to each other. This is because the iterative methods do not exactly converge to the most optimal point given by the closed form solution. The batch gradient descent comes the closest to the closed form solution, while the stochastic gradient descent has more differences because it can move around in random directions as it approaches the optimum. If the learning rate is lower with more iterations, both gradient descent methods could eventually come very close to the closed form solution.

b) For the closed form solution with normalization, my results were as follows.

```
Beta:  [ 5.23000000e-01 -3.95099505e-02 -3.01401932e-02 -5.71438644e-02
 -1.72769796e-02 -4.13700127e-03 -5.86318630e-02 -6.89027284e-02
 -3.56331805e-02 -1.87845537e-02 -1.82888714e-02  5.29276130e-02
  2.53519018e-02 -4.15812928e-02 -3.30193382e-02  2.65867992e-03
  1.34068950e-02 -3.88013327e-02  4.11038867e-02 -2.32239983e-02
 -2.68494719e-02 -5.67582270e-02  2.85948574e-02 -5.22058491e-02
 -1.94232592e-02  4.61988692e-02 -3.87491283e-02  3.82055256e-02
  1.27021593e-02  5.82271850e-02 -4.20937718e-02 -8.05582038e-02
  5.50688227e-02 -2.88202457e-02 -1.94706479e-02  2.58596756e-03
  2.55048685e-02  1.39991237e-02 -3.38312079e-02 -1.80218433e-02
 -8.42135902e-03 -5.61252496e-02 -3.60939866e-02 -1.12787490e-04
 -4.02969672e-02 -1.20851201e-02 -1.41809480e-02  5.11770552e-03
  4.48842190e-02  1.42864924e-02  1.79066117e-02 -3.08841654e-02
 -3.67139837e-02  3.83560781e-02 -4.47435146e-02 -6.08180754e-02
  4.69774181e-02 -5.86346690e-02  1.62361334e-02 -6.06942237e-02
 -3.38205570e-02 -4.24317897e-02 -5.46648364e-02 -2.89378305e-02
  5.33687506e-02 -3.17462303e-02  2.12826319e-03 -3.26837546e-02
  6.84819052e-03  1.25455103e-02 -4.09640271e-02  8.88512549e-03
  1.94883628e-02  6.04797247e-02 -4.23185183e-02 -4.76582979e-02
 -6.69833777e-02  5.66019062e-02  4.63178581e-03  4.13664903e-02
  7.10828556e-02  4.08986579e-02 -6.46605942e-02  3.05062530e-02
  6.11970818e-02 -6.13118531e-04  4.12093831e-02  8.04511196e-05
  3.21203863e-02  5.30651849e-02 -2.83935172e-02 -4.22856651e-02
  4.23271015e-02 -1.72635991e-03 -6.75124152e-02 -3.30151234e-02
 -2.14687553e-03 -6.00152621e-02  4.30059659e-02  6.79904935e-02
 -3.84367853e-03]
Training MSE:  0.08693886675396784
Test MSE:  0.11017540281675804
```

For the batch gradient descent with normalization, my results were as follows.

```
Beta:  [ 5.23000000e-01 -3.95101936e-02 -3.01400849e-02 -5.71436016e-02
 -1.72767566e-02 -4.13680968e-03 -5.86319214e-02 -6.89032234e-02
```

```
 -3.56330657e-02 -1.87841217e-02 -1.82887658e-02  5.29276931e-02
  2.53523482e-02 -4.15814596e-02 -3.30189000e-02  2.65905421e-03
  1.34067273e-02 -3.88011854e-02  4.11037614e-02 -2.32240541e-02
 -2.68494945e-02 -5.67579750e-02  2.85951506e-02 -5.22054525e-02
 -1.94231842e-02  4.61989821e-02 -3.87489331e-02  3.82054501e-02
  1.27021239e-02  5.82268258e-02 -4.20940954e-02 -8.05573634e-02
  5.50689558e-02 -2.88196801e-02 -1.94710680e-02  2.58553047e-03
  2.55047214e-02  1.39994226e-02 -3.38314427e-02 -1.80216795e-02
 -8.42163324e-03 -5.61254245e-02 -3.60937290e-02 -1.12375791e-04
 -4.02968579e-02 -1.20849591e-02 -1.41807601e-02  5.11749920e-03
  4.48844305e-02  1.42864598e-02  1.79068013e-02 -3.08838475e-02
 -3.67140506e-02  3.83564459e-02 -4.47432800e-02 -6.08186932e-02
  4.69775375e-02 -5.86348267e-02  1.62358161e-02 -6.06943926e-02
 -3.38207247e-02 -4.24316426e-02 -5.46643054e-02 -2.89379794e-02
  5.33687847e-02 -3.17460535e-02  2.12781890e-03 -3.26837622e-02
  6.84804807e-03  1.25455942e-02 -4.09640721e-02  8.88515498e-03
  1.94879083e-02  6.04800905e-02 -4.23184267e-02 -4.76582901e-02
 -6.69831131e-02  5.66016664e-02  4.63151746e-03  4.13661728e-02
  7.10829186e-02  4.08988871e-02 -6.46605503e-02  3.05063276e-02
  6.11972940e-02 -6.13108962e-04  4.12095655e-02  8.03641186e-05
  3.21205945e-02  5.30652999e-02 -2.83934643e-02 -4.22856178e-02
  4.23273822e-02 -1.72618033e-03 -6.75124367e-02 -3.30146697e-02
 -2.14697095e-03 -6.00147938e-02  4.30057056e-02  6.79898214e-02
 -3.84341156e-03]
Training MSE:  0.0869388667577727
Test MSE:  0.11017549781082567
```

For the stochastic gradient descent with normalization, my results were as follows.

```
Beta:  [ 5.22741019e-01 -3.94120236e-02 -3.00839409e-02 -5.73610123e-02
 -1.75043981e-02 -4.28299899e-03 -5.86020198e-02 -6.88782381e-02
 -3.54447121e-02 -1.87650114e-02 -1.82157832e-02  5.30176045e-02
  2.51799462e-02 -4.19312505e-02 -3.30069940e-02  2.74635496e-03
  1.33798468e-02 -3.91170084e-02  4.10881831e-02 -2.34457506e-02
 -2.68658205e-02 -5.67733793e-02  2.85412906e-02 -5.22164280e-02
 -1.93501407e-02  4.60646773e-02 -3.90170237e-02  3.82006657e-02
  1.30882941e-02  5.81289783e-02 -4.22180476e-02 -8.06102894e-02
  5.51029928e-02 -2.88582300e-02 -1.96433051e-02  2.57845570e-03
  2.55584390e-02  1.38724109e-02 -3.40623635e-02 -1.81011980e-02
 -8.42498301e-03 -5.60598331e-02 -3.59959463e-02  9.38345410e-05
 -4.01352608e-02 -1.22651066e-02 -1.42533806e-02  5.09891016e-03
  4.51153515e-02  1.41921750e-02  1.76191596e-02 -3.08863950e-02
 -3.68347321e-02  3.80046243e-02 -4.46304727e-02 -6.08590168e-02
  4.68289440e-02 -5.86216782e-02  1.60793011e-02 -6.11310317e-02
 -3.40968733e-02 -4.25480999e-02 -5.44811420e-02 -2.88358813e-02
  5.33882331e-02 -3.20266074e-02  2.01279950e-03 -3.24378281e-02
  6.69948426e-03  1.25255183e-02 -4.09354596e-02  8.96241378e-03
  1.93830951e-02  6.04843325e-02 -4.23063725e-02 -4.75919050e-02
 -6.69680975e-02  5.67524813e-02  4.62892117e-03  4.11990177e-02
  7.11877053e-02  4.08196677e-02 -6.48099470e-02  3.04255895e-02
  6.14083660e-02 -5.67508606e-04  4.13467943e-02  2.47771340e-04
  3.21770657e-02  5.30485993e-02 -2.84268198e-02 -4.22707931e-02
  4.23427257e-02 -1.30903299e-03 -6.73548500e-02 -3.32209167e-02
 -1.90140621e-03 -6.01926524e-02  4.29478948e-02  6.76920264e-02
 -4.00731006e-03]
Training MSE:  0.0869420129183826
Test MSE:  0.11014750653569365
```

The normalization affects the weights produced, as each feature is scaled. However, the mean square error is the same because the y values were not changed. So the most optimal solution produces the same predicted values. Effectively, the change in the weights undoes the normalization when comparing the predicted values.

Problem 2

This cost function is the same except with an additional $\frac{\lambda}{2n}\beta^T\beta$ term. Since the derivative is linear we know that the derivative of the cost function is the following.

$$\frac{\partial J}{\partial \beta} = \frac{X^T X\beta - X^T y + \lambda\beta}{n}$$

Setting this equal to zero yields the following.

$$\begin{aligned} X^T X\beta - X^T y + \lambda\beta &= 0 \\ (X^T X + \lambda I)\beta &= X^T y \\ \beta &= (X^T X + \lambda I)^{-1} X^T y \end{aligned}$$

2 Logistic Regression

a) For the batch gradient descent without normalization, my results were as follows.

```
Beta:  [ 0.533796    0.0046305   0.21859055  0.603253   -0.10366392  0.78285059]
Training avgLogL:  -2.253178474041476
Test accuracy:  0.5149105367793241
```

For the batch gradient descent with normalization, my results were as follows.

```
Beta:  [ 0.27910793  2.15368204  1.23434665  1.32102078 -0.66170494  0.41580785]
Training avgLogL:  -0.4601003753508531
Test accuracy:  0.7534791252485089
```

For the Newton-Raphson method without normalization, my results were as follows.

```
Beta:  [-2.00331994e+07  6.40900006e+07  4.79768070e+08  5.65078628e+06
 -1.45081644e+09  1.46521965e+06]
Training avgLogL:  -26937895719.032852
Test accuracy:  0.4970178926441352
```

For the Newton-Raphson method with normalization, my results were as follows.

```
Beta:  [ 0.27910793  2.15368204  1.23434665  1.32102078 -0.66170494  0.41580785]
Training avgLogL:  -0.46010037535085296
Test accuracy:  0.7534791252485089
```

They are not the same, the results from the non-normalized versions performed significantly worse than the normalized versions. The normalized versions behaved the same between the two methods, and produced similar results. The Newton-Raphson method without normalization seemed to diverge. The large differences in magnitude between the features makes the method behave unpredictably. Without normalization both methods produce almost meaningless results since the accuracy is very bad. For this data set normalization makes sense.

A pro of the standard batch gradient descent method is that it is simpler to implement and faster to execute than the Newton-Raphson method, since it does not need to calculate the second derivative. It also converged on the non-normalized data when I set the learning rate to be very small, while the Newton-Raphson method did not converge at all. A con is that the learning rate takes a lot of testing to tune properly. A pro of the Newton-Raphson method is that there is no need to set the learning rate, but a con is that the data must be normalized for it to work well.

b) The left term is simply a scaled version of the log likelihood that is multiplied by a negative. The right term has only one factor dependent on β_j which is $\lambda\beta_j^2$. Thus the first derivative is shown below.

$$\frac{\partial J(\beta)}{\partial \beta_j} = -\frac{1}{n}\sum_{i=1}^{n} x_{ij}(y_i - p_i(\beta)) + 2\lambda\beta_j$$

For gradient descent we want to minimize this objective function, which can be implemented in the code by dividing the log likelihood derivative by n and subtracting $2\lambda\beta$.

For the regularized batch gradient descent without normalization, my results were as follows.

```
Beta:  [ 0.62840718  0.00210163  0.78134471  0.91813354 -0.16413573  0.10951218]
Training avgLogL:  -1.1156340680708068
Test accuracy:  0.6083499005964215
```

For the regularized batch gradient descent with normalization, my results were as follows.

```
Beta:  [ 0.09380685  0.72848483  0.24504096  0.44285312 -0.26543956  0.33631754]
Training avgLogL:  -0.5315832206246545
Test accuracy:  0.7534791252485089
```

Regularization slightly increased test accuracy when there was no normalization, but that may have been due to chance because my results varied each time I ran the regression. It is hard to make conclusions regarding the non-normalized data since its performance changes a lot. Regularization decreased the average log likelihood for the normalized case, since the regularization would make the weights not fit the training data as well. Since I set λ to be small, the test accuracy for the normalized case remained approximately the same.