Vincent La
903178639-vla6-hw2

Logistic Regression Proofs

1.1.a Batch Gradient Descent

*Derive the gradient of the negative log-likelihood in terms of **w** for this setting.*

We know that:

$$NLL(D,w) = -\sum_{\{i=1\}}^{N} [(1-y_i)\log\big(1-\sigma(w^T x_i)\big) + y_i \log\sigma(w^T x_i)]$$

First, we prove that

$$\frac{\partial}{\partial z}\big(\sigma(z)\big) = \sigma(z)(1-\sigma(z))$$

Note that

$$\frac{\partial}{\partial z}\big(\sigma(z)\big) = -(1+e^{-z})^{-2}(-e^{-z})$$

$$= \frac{e^{-z}}{(1+e^{-z})^2}$$
$$= \frac{1}{1+e^{-z}} * \frac{e^{-z}}{1+e^{-z}}$$
$$= \sigma(z) * (1-\sigma(z))$$

So now that we know this identity we can proceed.

To derive the gradient of the negative log-likelihood we have:

$$\frac{\partial}{\partial w_j} NLL(D,w) = -\sum_{\{i=1\}}^{N} \frac{\partial}{\partial w_j}[(1-y_i)\log\big(1-\sigma(w^T x_i)\big) + \frac{\partial}{\partial w_j} y_i \log\sigma(w^T x_i)]$$

$$-\sum_{\{i=1\}}^{N}\left[\frac{1-\mathrm{y_i}}{1-\sigma(\mathrm{w^T x_i})} + \frac{\mathrm{y_i}}{\sigma(\mathrm{w^T x_i})}\right] * \frac{\partial}{\partial w_j}\sigma(\mathrm{w^T} x_i)$$

$$-\sum_{\{i=1\}}^{N}\left[\frac{\mathrm{y}-\sigma(\mathrm{w^T} x_i)}{\sigma(\mathrm{w^T} x_i)(1-\sigma(w^T x_i))}\right] * (\sigma(\mathrm{w^T} x_i)(1-\sigma(w^T x_i)x^j)$$

$$= -\sum_{\{i=1\}}^{N} \mathbf{x}^{\mathbf{j}}(\boldsymbol{y_i} - \boldsymbol{\sigma}(\boldsymbol{w^T x_i}))$$

1.2 Stochastic Gradient Descent

1.2.a

Show the log likelihood, l, of a single (x_t, y_t) pair

For a single (x_t, y_t) pair, recall that y_t can only take on values 1 or 0 (since we're assuming binary classifier.

Thus, log likelihood, l, of a single pair is

$$\begin{cases} \log \sigma(w^T x_i) & if\ y = 1 \\ \log\big(1 - \sigma(w^T x_i)\big) & if\ y = 0 \end{cases}$$

1.2.b

Show how to update the coefficient vector

For a single training point, recall that

$$\frac{\partial}{\partial w} l(w|y, x) = \big(y - \sigma(w^T x_i)\big)x^j$$

This is taken from question 1.1 of this HW assignment.

Thus, the update rule is

$$w^{t+1} = w^t + \eta\big(y - \sigma(w^t x_i)\big)x^j$$

Where the learning rate is represented by η.

1.2.c

*What is the time complexity of the update rule from **b if x_t is very sparse?***

Notice that if x_t is very sparse, then in the update rule, notice that if $x^j = 0$; then $w^{t+1} = w^t$. Thus in the case where it is very sparse then in the case where the feature is 0, then you don't actually need to update.

Let $D_{non-zero}$ represent the non-zero features. Then, the time complexity is

$$O(D_{non-zero})$$

Or I suppose that if x_t is actually very very sparse, then the time complexity might be argued as $O(1)$ or constant.

1.2.d

Briefly explain the consequence of using a very large learning rate and very small learning rate.

If learning rate is too small, it will take longer to converge; if the learning rate is too large, may fail to converge.

1.2.e

Under the penalty of L2 norm regularization, notice that

$$\frac{\partial}{\partial w_j}[l - \mu \sum_{j=1}^{d} (w_j)^2] = \big(y - \sigma(w^T x_i)\big)x^j - 2\mu w^j$$

Thus the update rule becomes

$$w^{t+1} = w^t + \eta\big[\big(y - \sigma(w^T x_i)\big)x^j - 2\mu w^j\big]$$

Notice that now when x^j is zero, or when there is a sparse matrix, there is still an update that happens. Thus the time complexity is

$$O(D)$$

Where D is the number of features.

2.1 Descriptive Statistics

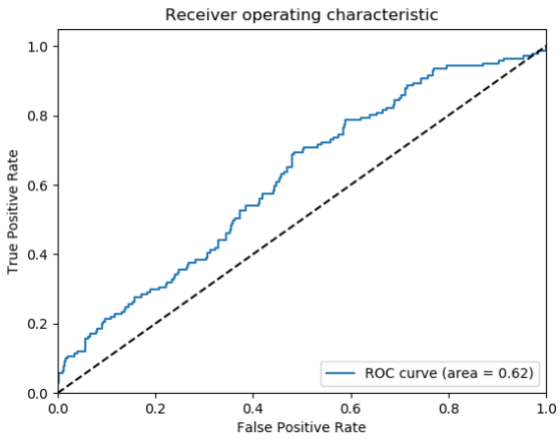
Metric	Deceased Patients	Alive Patients
Event Count		
1. Average Event Count	1027.74	683.16
2. Max Event Count	16829	12627
3. Min Event Count	2	1
Encounter Count		
1. Average Encounter Count	24.84	18.695
2. Max Encounter Count	375	391
3. Min Encounter Count	1	1
Record Length		
1. Average Record Length	157.042	194.702
2. Median Record Length	25.0	16
3. Max Record Length	5364	3103
4. Min Record Length	0	0
Common Diagnosis	DIAG320128 DIAG319835 DIAG313217 DIAG197320 DIAG132797	DIAG320128 DIAG319835 DIAG317576 DIAG42872402 DIAG313217
Common Laboratory Test	LAB3009542 LAB3023103	LAB3009542 LAB3000963

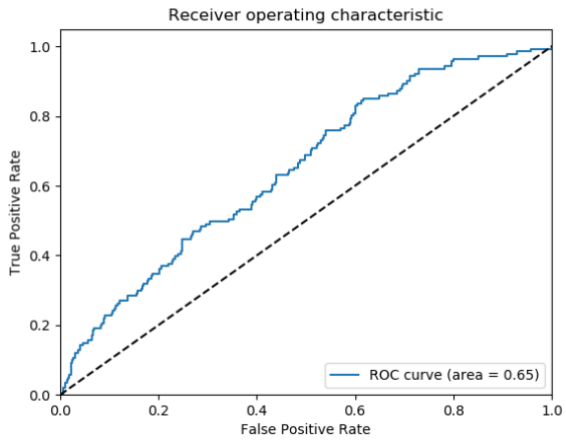
	LAB3000963 LAB3018572 LAB3016723	LAB3023103 LAB3018572 LAB3007461
Common Medication	DRUG19095164 DRUG43012825 DRUG19049105 DRUG956874 DRUG19122121	DRUG19095164 DRUG43012825 DRUG19049105 DRUG19122121 DRUG956874

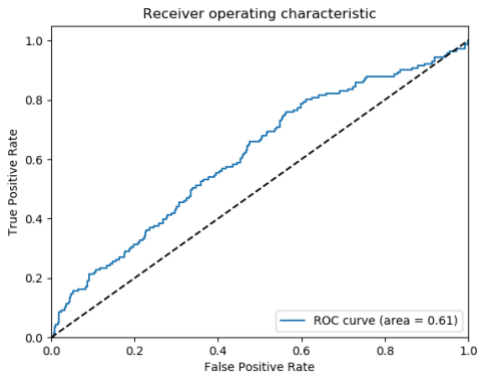
2.3. SGD Logistic Regression

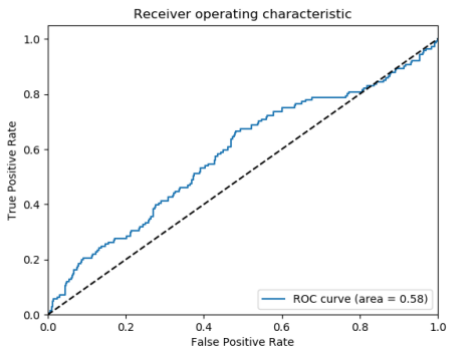
Let C = Regularization Constant
e = Learning Rate

With C = 0 and e = 0.01 (top left): **ROC: 0.62**
With C = 0 and e = 0.3 (top right): **ROC: 0.65**
With C = 0 and e = 0.5 (bottom left): **ROC: 0.61**
With C = 0 and e = 1 (bottom right): **ROC: 0.58**





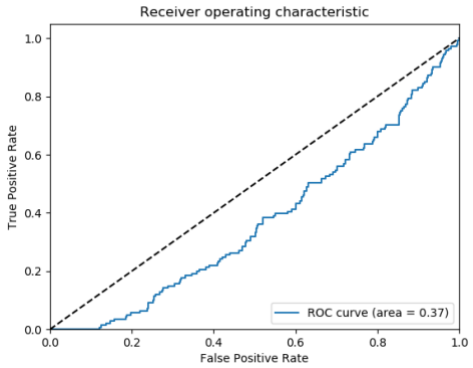


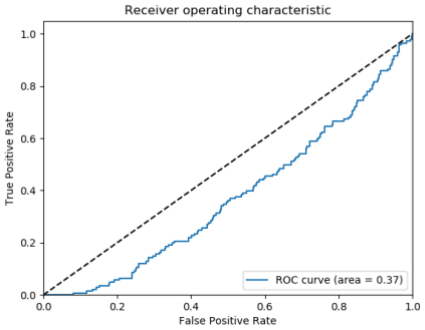


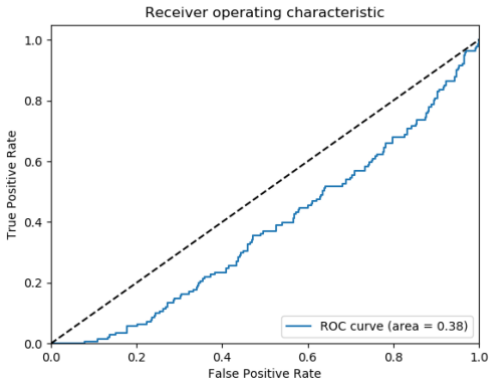
Notice that when we increasing learning rate parameter we generally got better ROC which means that perhaps with very low learning rate, we are overfitting. However, notice that when learning rate is too large performance, ROC goes down again. Note if too large, the implementation may actually fail to converge.

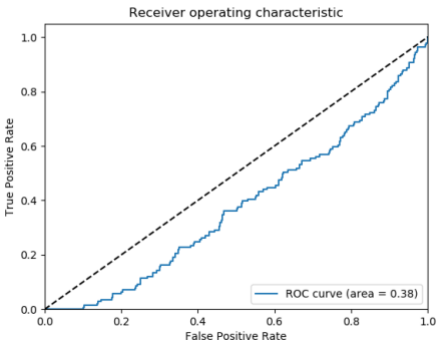
In the previous example, learning rate of e = 0.3 seemed to have yielded the best ROC. Thus let's take that and now start varying the regularization parameter, C

With C = 0.1 and e = 0.3 (top left): **ROC: 0.63 (ROC Flipped)**
With C = 0.3 and e = 0.3 (top right): **ROC: 0.63 (ROC Flipped)**
With C = 0.5 and e = 0.3 (bottom left): **ROC: 0.62 (ROC Flipped)**
With C = 0.7 and e = 0.3 (bottom right): **ROC: 0.62 (ROC Flipped)**





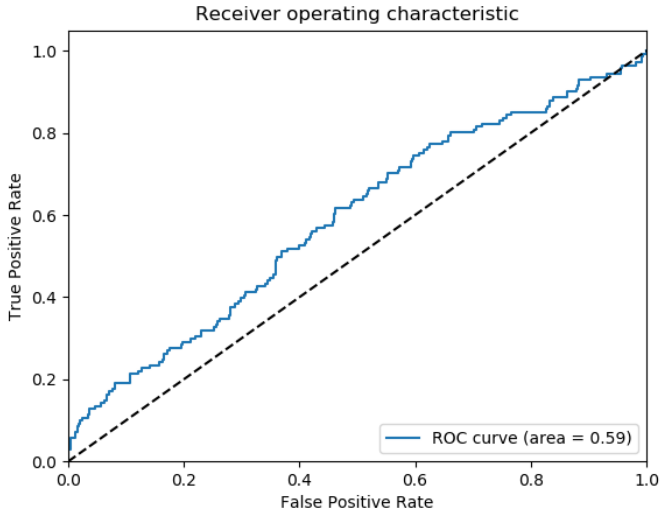




Interestingly enough notice that varying the regularization term doesn't change performance too much.

2.4. Hadoop

Using the default parameters, C=0; e = 0.01, the ROC curve looks like the following, with **AUROC = 0.59**. Which is actually worse



When we use the best parameters from the previous part, C = 0; e = 0.3, the ROC curve looks like this AUROC = 0.59, which is about the same.