Logistic Regression and Stochastic Gradient Ascent Analysis

Paul Laliberte' | CSCI-5622

1. How did the learning rate affect the convergence of your SGA implementation?

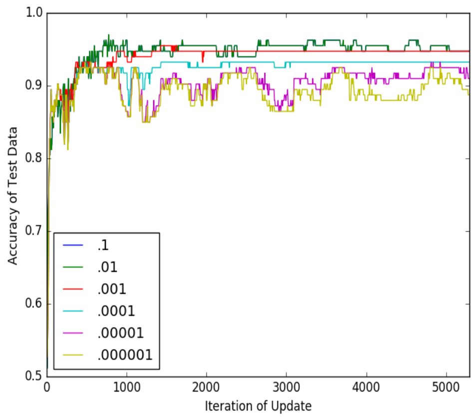

Figure 1: Learn Rate and Convergence

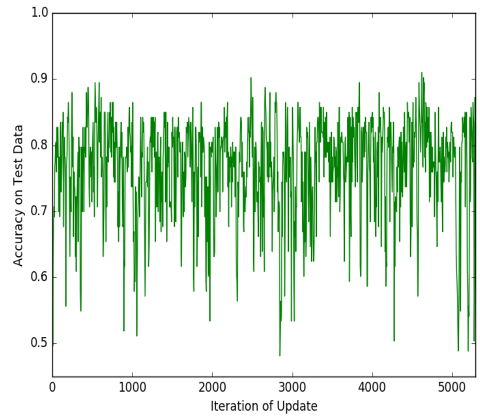

Figure 2: Lack of Converging Solution

The initial learning rate was $\eta = 0.1$ with passes = 5, no other flags were given (non-regularized and tf-idf turned off). We decreased the rate by magnitudes of .1, i.e. .01, .001, .0001, and so on. There are two important aspects to take from Figure 1. The first being that a low initial learning rate is good in a sense that there is minimal overfitting to the data set. However, a learning rate that is low may also not be feasible in a timely matter. Next, we consider a high initial learning rate. From Figure 1, we can infer that a higher learning rate can lead to a much faster convergence. The drawback to this faster rate is that we may be overfitting our data. A $\eta = .1$ achieves almost full convergence in only one pass through the training data, surely that is not enough. A learning rate that is not too slow, but does not converge immediately is ideal.

2. What was your stopping criterion and how many passes over the data did you need before stopping?

The stopping criterion considered two factors: 1) Was convergence a viable possibility? 2) What was the rate of convergence? We first analyzed the issue of whether convergence was possible. To give an example, running the program with a $\eta = .1$ and $\lambda = .25$ produced a divergent solution, no matter the number of passes through the data we assigned (Figure 2). The variance over the data was consistent for both training data (TP, TA) and testing data (HP, HA), where TP and HP are the log probabilities. If there seemed to be a definitive convergence we then consider the rate of that convergence. To be more specific, running the program with a $\eta = .001$, $\lambda = .25$, and passes = 10 we would eventually find a solution that was at, or near, convergence at about 6 to 7 passes through the data. At this point we were getting minimal improvement in the convergence rate of TP, HP, TA, and HA. Hence, we could stop at this point and be confident in the results. In summary, if the convergence rate was minimal in one pass through the data then we considered this a good stopping point.

3. What words are the best predictors of each class? How (mathematically) did you find them?

To classify words as good predictors, we used the tf-idf weight with $\eta = .1$, $\lambda = 0.0$, and passes = 100. We define

$$\text{tf}(t) = \frac{\#\text{ of times term } t \text{ appears in doc}}{\text{total } \#\text{ of term } t \text{ in doc}},$$

and

$$\text{idf}(t) = \log\left(\frac{\text{total } \#\text{ of docs}}{\#\text{ of docs with term } t}\right).$$

Furthermore, the tf-idf weight is given as

$$\text{tf-idf}(t) = \text{tf}(t) \times \text{idf}(t).$$

We then summed every tf-idf calculation as we passed through the data. The highest tf-idf weighted sum scores were the best predictor words. We also took into consideration words that did not appear in the relevant classes and removed them from the rating (assigned a special value to them). The twenty best predictor words of each class can be found in the first to tables, from the left, in Figure 3 (auto and then cycle). We can see that there are several words that overlap between the two, but there are also very distinct words that we would assume to associate with best predictors (car, cars, bike, ride, riding).

4. What words are the poorest predictors of classes? How (mathematically) did you find them?

To classify words as bad predictors, we used the same tf-idf weight that was calculated in (3). The lowest tf-idf weighted sum scores were the worst predictor words. The twenty worst predictor words of each class can be found in the last two tables, from the left, in Figure 3 (auto and then cycle).

	term	tfidf (sum)
0	car	99.304331
1	distribution	72.543926
2	like	61.255098
3	one	58.244466
4	usa	54.896685
5	cars	49.358891
6	reply	48.777984
7	know	48.750605
8	good	47.470158
9	new	46.013241
10	get	45.351627
11	please	42.591214
12	thanks	41.668738
13	also	38.686989
14	think	38.008484
15	time	36.448811
16	ca	32.819943
17	much	32.645049
18	right	30.715554
19	want	30.276095

	term	tfidf (sum)
0	dod	100.652430
1	bike	85.889095
2	one	68.474187
3	like	67.807797
4	distribution	55.874916
5	get	49.700894
6	know	47.163276
7	ca	47.049867
8	new	39.901945
9	reply	39.687749
10	ride	39.408613
11	good	38.269571
12	bikes	36.091646
13	think	35.315823
14	riding	33.196231
15	well	32.578283
16	time	32.033011
17	world	30.886983
18	usa	30.527954
19	much	30.124241

	term	tfidf (sum)
0	characteristics	0.034993
1	eventual	0.034993
2	approved	0.034993
3	pistons	0.034993
4	distances	0.034993
5	boiling	0.034993
6	unlike	0.034993
7	material	0.034768
8	ignoring	0.034768
9	imply	0.034768
10	trained	0.034768
11	corresponding	0.034768
12	empty	0.034768
13	regardless	0.034586
14	requires	0.034433
15	occasional	0.034433
16	preferable	0.034433
17	usual	0.034433
18	training	0.033759
19	cycle	0.033694

	term	tfidf (sum)
0	oriented	0.029919
1	civil	0.029919
2	standing	0.029764
3	signed	0.029764
4	rapidly	0.029764
5	regulations	0.029764
6	regarding	0.029764
7	williams	0.029634
8	crawl	0.029634
9	94	0.029634
10	mini	0.029522
11	fan	0.029522
12	followups	0.029522
13	utility	0.029522
14	seats	0.029336
15	popular	0.029336
16	defense	0.029257
17	thunder	0.028857
18	bird	0.028735
19	network	0.028431

Figure 3: tf-idf Weighted Sums, left-to-right: auto, cycle, auto, cycle

Extra Credit 1.

Extra Credit 2.

We will give two comparisons: 1) How tf-idf slowed the rate of convergence. 2) How the weighted sum of tf-idf compared with tf in determining the best predictors for each class. For the first, we refer to Figure 4a. There is less variance (and hence less overfitting) over the majority of the tf-idf solutions. We now compare the best predictor words using tf-idf and only tf. See Figure 4b and 4c for the results. For auto, we see that there are some common words that ranked similar to both the tf-idf weighting and the flat tf count (car, like, cars). However, there are some terms in the flat tf count that did not appear in the tfidf weight (go, people) and vice versa (ca, want). Similarly, in cycle we have similar word rankings (dod, bike, ride), others that did appear in the tf count and not the tf-idf wight (go, back), and vice versa (usa, world, bikes).

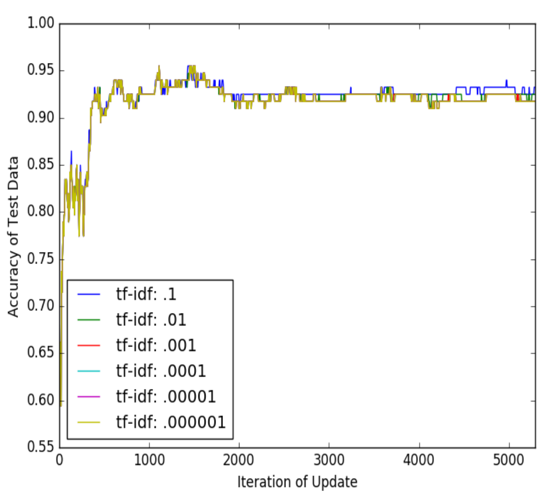

Figure 4: tf-idf

	term	tf
0	car	2950
1	like	1920
2	one	1910
3	distribution	1730
4	cars	1600
5	get	1540
6	good	1490
7	usa	1450
8	know	1420
9	think	1370
10	time	1260
11	also	1260
12	much	1250
13	new	1230
14	reply	1170
15	right	1040
16	go	1030
17	people	1030
18	please	1020
19	really	1010

Figure 5: auto term frequency

	term	tf
0	dod	2780
1	bike	2330
2	one	2080
3	like	2000
4	get	1650
5	know	1410
6	distribution	1290
7	ca	1290
8	good	1190
9	ride	1170
10	think	1120
11	new	1090
12	well	1080
13	time	1080
14	much	980
15	go	960
16	riding	950
17	back	940
18	reply	940
19	right	920

Figure 6: cycle term frequency