Homework Assignment 1

SDS 385 Statistical Models for Big Data

Please upload the HW on canvas before class Oct 11th by 10am. Please type up your homework using latex. We will not accept handwritten homeworks.

1. (10 pts) **Convex functions:** Using the definition of convex function, i.e. $f(tx + (1-t)y) \leq tf(x) + (1-t)f(\beta')$ show that the following functions are convex.

   (a) (3pts) e^x

   (b) (2pts) If $f(x)$ is convex for $x \in \Re^p$, show that so is $f(Ax+b)$ for $A \in \Re^{p\times p}$ and $b \in \Re^p$.

   (c) (2pts) If $f_i(x), i \in [k]$ are convex functions, show that the pointwise maximum, i.e. $g(x) = \max_{i\in[k]} f_i(x)$ is also convex.

   (d) (3 pts) Consider the logistic regression problem. For $x \in \Re^p$, You have

$$y_i \sim Bernoulli\left(\frac{1}{1+e^{-\theta^T x}}\right)$$

   i. (1pt) Write down the log likelihood function.

   ii. (2pt) Show that this is concave. *Hint: for part d you can use first/second order conditions and properties of concave functions.*

Solution:

(a) Since that $f(tx+(1-t)y) \leq tf(x)+(1-t)f(y)$ shows the convexity of a function, we can prove e^x is convex by proving that:

$$e^{tx+(1-t)y} \leq te^x + (1-t)e^y$$

Assume that $x < y$ and divide through e^x to get that:

$$e^{(t-1)x}e^{(1-t)y} \leq t + (1-t)e^{(y-x)}$$

$$e^{(1-t)(y-x)} \leq t + (1-t)e^{(y-x)}$$

Denote $y - x = P$ and $P > 0$, the inequality becomes like:

$$e^{(1-t)P} \leq t + (1-t)e^P \tag{1}$$

Use Taylor series, the left part is

$$e^{(1-t)p} = 1 + (1-t)p + \frac{1}{2}(1-t)^2p^2...$$

the right part is

$$\begin{aligned} t + (1-t)e^p &= t + (1-t)(1 + p + \frac{1}{2}p^2 + ...) \\ &= 1 + (1-t)p + (1-t)\frac{1}{2}p^2 + ... \end{aligned}$$

As $0 \leq t \leq 1$, so $0 \leq 1 - t \leq 1$

$$(1-t)^n \leq (1-t) \forall n \geq 0$$

Therefore,

$$\begin{aligned} e^{(1-t)p} &= 1 + (1-t)p + \frac{1}{2}(1-t)^2p^2... \\ &\leq 1 + (1-t)p + (1-t)\frac{1}{2}p^2 + ... \\ &= t + (1-t)e^p \end{aligned}$$

The equation (1) is hereby proved. So the function e^x is convex.

(b) We change the notation to make it easier to understand, where $A \in \Re^{p \times p}$ and $b \in \Re^p$,

$$g(x) = f(Ax + b)$$

Use the definition that as f is convex $f(tx + (1-t)y) \leq tf(x) + (1-t)f(y)$ for any two points x and y that $x, y \in \Re^p$,

$$\begin{aligned} g(tx + (1-t)y) = f(A(tx + (1-t)y + b) = f(t(Ax + b) + (1-t)(Ay + b)) \\ \leq tf(Ax + b) + (1-t)f(Ay + b) = tg(x) + (1-t)g(y) \end{aligned}$$

So $g(x) = f(Ax + b)$ is also convex where where $A \in \Re^{p \times p}$ and $b \in \Re^p$.

(c) Since it is given that all $f_i(x), i \in [k]$ are convex, for x and y that $x, y \in \Re^p$

$$f_i(tx + (1-t)y) \leq tf_i(x) + (1-t)f_i(y)$$

Take maximum of the both sides,

$$max_i[f_i(tx + (1-t)y)] \leq max_i[tf_i(x) + (1-t)f_i(y)]$$

Clearly,

$$max_i[tf_i(x) + (1-t)f_i(y)] \leq max_i[tf_i(x)] + max_i[(1-t)f_i(y)]$$

So,

$$max_i[f_i(tx + (1-t)y)] \leq max_i[tf_i(x)] + max_i[(1-t)f_i(y)]$$

Thus, we can prove that $g(x) = \max_{i \in [k]} f_i(x)$ is also convex which follows that:

$$g(tx + (1-t)y) \leq tg(x) + (1-t)g(y)$$

(d) **(i)** Since $y_i \sim Bernoulli\left(\frac{1}{1+e^{-\theta^T x}}\right)$, $y_i \in \{0,1\}$,we define that:

$$p(x_i) = \Pr(y_i = 1|x_i)$$
$$\Pr(y_i = 0|x_i) = 1 - p(x_i)$$

Here we view x_i as a row-vector and the coefficient θ^T as a column-vector, so

$$p(x_i;\theta) = \frac{1}{1+e^{-\theta^T x}}$$

Log likelihood for the ith observation is:

$$\begin{aligned} l_i(\theta|x_i) &= (1-y_i)\log[1-p(x_i;\theta)] + y_i \log p(x_i;\theta) \\ &= \begin{cases} \log p(x_i;\theta) & \text{if } y_i = 1 \\ \log[1-p(x_i;\theta)] & \text{if } y_i = 0 \end{cases} \end{aligned}$$

Joint log likelihood for n observations:

$$\begin{aligned} l(\theta|x_1,...x_n) &= \sum_{i=1}^{n} l_i(\theta|x_i) \\ &= \sum_{i=1}^{n} (1-y_i)\log[1-p(x_i;\theta)] + y_i \log p(x_i;\theta) \\ &= \sum_{i=1}^{n} y_i \log\frac{p(x_i;\theta)}{1-p(x_i;\theta)} + \log[1-p(x_i;\theta)] \\ &= \sum_{i=1}^{n} y_i\theta^T x_i - \log(1+e^{-\theta^T x_i}) \end{aligned}$$

(ii) We take the first and second derivative of the log likelihood function, which can get that:

$$l^{'}(\theta) = \sum_{i=1}^{n} x_i(y_i - p(x_i)),$$
$$l^{''}(\theta) = -\sum_{i=1}^{n} x_i^2 p(x_i)(1-p(x_i))$$

The function l is twice-differentiable and its second derivative is negative so that the log likelihood function l is strictly concave.

2. (10 pts) **Convergence of gradient descent:** In class, we used strong convexity to show convergence of GD. In this homework we will revisit this for Lipschitz functions. To be concrete, suppose the function f is convex and differentiable and its gradient is Lipschitz condition with constant $L > 0$, i.e. we have

$$\|\nabla f(\beta) - \nabla f(\beta')\| \leq L\|\beta - \beta'\|_2, \qquad \text{For any } \beta, \beta'$$

In this problem we run GD for t iterations with a fixed step size $\alpha < 1/L$.

(a) (1 pt) First show that for any β',

$$f(\beta') \leq f(\beta) + \nabla f(\beta)^T(\beta' - \beta) + \frac{L}{2}\|\beta' - \beta\|^2$$

(b) (3 pts) Let $\beta_{t+1} = \beta_t - \alpha\nabla f(\beta_t)$. Now show:

$$f(\beta_{t+1}) \leq f(\beta_t) - t\|\nabla f(\beta_t)\|^2/2$$

(c) (3 pts) Now show that $f(\beta_{t+1}) - f(\beta^*) \leq \frac{1}{2\alpha}(\|\beta_t - \beta^*\|^2 - \|\beta_{t+1} - \beta^*\|^2)$

(d) (3 pts) Using this, show that

$$f(\beta_t) - f(\beta^*) \leq \frac{\|\beta_0 - \beta^*\|^2}{2\alpha t}$$

Solution:

(a) According to the characterization and definition of L-Lipschitz gradient,

$$\|\nabla f(\beta) - \nabla f(\beta')\| \leq L\|\beta - \beta'\|_2, \qquad \text{For any } \beta, \beta'$$

Move the right part of norm to left and get

$$\frac{\|\nabla f(\beta) - \nabla f(\beta')\|}{\|\beta - \beta'\|_2} \leq L,$$

$$f''(\beta) \leq L$$

It is easy to Taylor expansion and derive that:

$$f(\beta) = f(\beta') + \nabla f(\beta')(\beta - \beta') + \frac{f''}{2!}(\beta - \beta')^2$$

Change the notation and use the $f'' \leq L$, following equation can be proved:

$$\begin{aligned} f(\beta') &= f(\beta) + \nabla f(\beta)^T(\beta' - \beta) + \frac{f''}{2}\|\beta' - \beta\|^2 \\ &\leq f(\beta) + \nabla f(\beta)^T(\beta' - \beta) + \frac{L}{2}\|\beta' - \beta\|^2 \end{aligned}$$

(b) According to the equation proved before that

$$f(\beta') \leq f(\beta) + \nabla f(\beta)^T(\beta' - \beta) + \frac{L}{2}\|\beta' - \beta\|^2$$

And also $\beta_{t+1} = \beta_t - \alpha\nabla f(\beta_t)$, thus $\beta_{t+1} - \beta_t = -\alpha\nabla f(\beta_t)$, change the notation to correspond to the equation above:

$$\begin{aligned} f(\beta_{t+1}) &\leq f(\beta_t) - \nabla f(\beta_t)^T\alpha\nabla f(\beta_t) + \frac{L}{2}\|\beta_{t+1} - \beta_t\|^2 \\ &= f(\beta_t) + (-\alpha + \frac{L}{2}\alpha^2)\|\nabla f(\beta_t)\|^2 \end{aligned}$$

Since that $\alpha < 1/L$, so

$$-\alpha + \frac{L}{2}\alpha^2 < -\alpha + \frac{1}{\alpha}\frac{\alpha^2}{2} = -\frac{\alpha}{2}$$

Therefore,

$$\begin{aligned} f(\beta_{t+1}) &\leq f(\beta_t) + (-\alpha + \frac{L}{2}\alpha^2)\|\nabla f(\beta_t)\|^2 \\ &\leq f(\beta_t) - \frac{\alpha}{2}\|\nabla f(\beta_t)\|^2 \end{aligned}$$

(c) It is known that:

$$f(\beta') \leq f(\beta) + \nabla f(\beta)^T(\beta' - \beta)$$

then derive

$$f(\beta_t) \leq f(\beta^*) + \nabla f(\beta_t)^T(\beta_t - \beta^*)$$

where β^* is the minimizer. Also, combine with the proven equation in (b) that:

$$f(\beta_{t+1}) \leq f(\beta_t) - \frac{\alpha}{2}\|\nabla f(\beta_t)\|^2$$

we can get

$$f(\beta_{t+1}) \leq f(\beta^*) + \nabla f(\beta_t)^T(\beta_t - \beta^*) - \frac{\alpha}{2}\|\nabla f(\beta_t)\|^2$$

$$f(\beta_{t+1}) - f(\beta^*) \leq \frac{1}{2\alpha}(2\alpha\nabla f(\beta_t)^T(\beta_t - \beta^*) - \alpha^2\|\nabla f(\beta_t)\|^2 - \|\beta_t - \beta^*\|^2 + \|\beta_t - \beta^*\|^2$$

thus,

$$f(\beta_{t+1}) - f(\beta^*) \leq \frac{1}{2\alpha}(-\|\beta_t - \alpha\nabla f(\beta_t)^T - \beta^*\|^2 + \|\beta_t - \beta^*\|^2)$$

$$f(\beta_{t+1}) - f(\beta^*) \leq \frac{1}{2\alpha}(\|\beta_t - \beta^*\|^2 - \|\beta_{t+1} - \beta^*\|^2)$$

(d) Generate from (c) can get:

$$\begin{aligned} \sum_{i=1}^{t} f(\beta_{t+1}) - f(\beta^*) &\leq \sum_{i=1}^{t}(\frac{1}{2\alpha}(\|\beta_{i-1} - \beta^*\|^2 - \|\beta_i - \beta^*\|^2)) \\ &= \frac{1}{2\alpha}(\|\beta_0 - \beta^*\|^2 - \|\beta_t - \beta^*\|^2) \\ &\leq \frac{1}{2\alpha}\|\beta_0 - \beta^*\|^2 \end{aligned}$$

As $f(\beta_t)$ decreases in each iteration, thus:

$$\begin{aligned} f(\beta_t) - f(\beta^*) &\leq \frac{1}{t}\sum_{i=1}^{t}(f(\beta_i) - f(\beta_*)) \\ &\leq \frac{1}{2t\alpha}\|\beta_0 - \beta^*\|^2 \end{aligned}$$

3. (20 pts) **Programming question** Logistic regression is a simple statistical classification method which models the conditional distribution of the class variable y being equal to class c given an input $x \in \mathbb{R}^p$. We will examine two classification tasks, one classifying newsgroup posts, and the other classifying digits. In these tasks the input x is some description of the sample (e.g. word counts in the news case) and y is the category the sample belongs to (e.g. sports, politics). The Logistic Regression model assumes the class distribution conditioned on x is log-linear. For C classes, the goal is to learn $\beta_1, \dots \beta_{C-1} \in \mathbb{R}^p$. We use the K^{th} class as a pivot.

$$\log \frac{p(Y = 1|X = x; \beta_1, \dots, \beta_{C-1})}{p(Y = C|X = x; \beta_1, \dots, \beta_{C-1})} = \beta_1^T x$$

Another way to think about this is to take β_C as all zeros. Thus,

$$P(Y = c|X = x, \beta_1, \dots, \beta_C) = \frac{\exp(\beta_c^T x)}{\sum_{j=1}^C \exp(\beta_j^T x)}. \tag{2}$$

Once the model is learned, one can classify a new point by picking the class that maximizes the posterior probability of belonging to that class (see Eq 2). You can measure convergence by the relative error of the concantenated parameter vector $\beta = [\beta_1^T \dots \beta_{K-1}^T] \in \mathbb{R}^{p(C-1)}$. You should write your loss function as an average, and you can use the regularization parameter to be $1/n$.

(a) Write down the likelihood of this model for n datapoints.

Extra credit Is the logarithm of this concave? Why?

(b) For the two datasets in the provided zip file, implement the following four methods. You will use ℓ_2 regularization.

i. Gradient descent

ii. Stochastic gradient descent

iii. Newton Raphson

(c) For each method, plot the loglikelihood as a function of number of iterations.

(d) For gradient descent try different step-sizes and provide a discussion on the effect of stepsize on the convergence.

(e) For SGD , how are you choosing your step-size?

(f) Finally compute the test set error and compare the GD and SGD methods on both of the datasets.

(g) Show the test error of NR on the small dataset. How does it perform compared to the other algorithms on the small dataset?

Solution:
(a) Likelihood of this model for n datapoints:

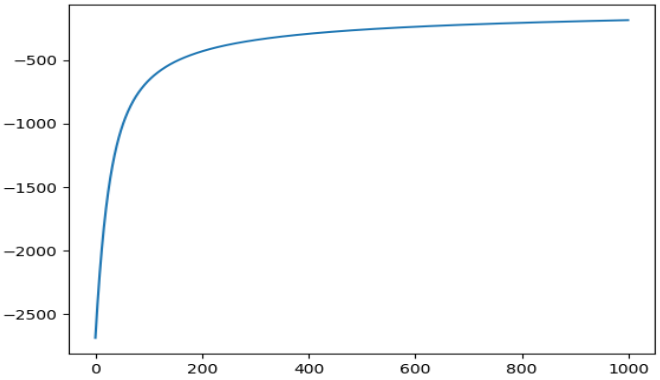
$$L(x; \beta_1, \beta_2, ..., \beta_C) = \prod_{i=1}^n \frac{\exp(\beta_c^T x_i)}{\sum_{j=1}^C \exp(\beta_j^T x_i)}$$

Take the log of joint likelihood:

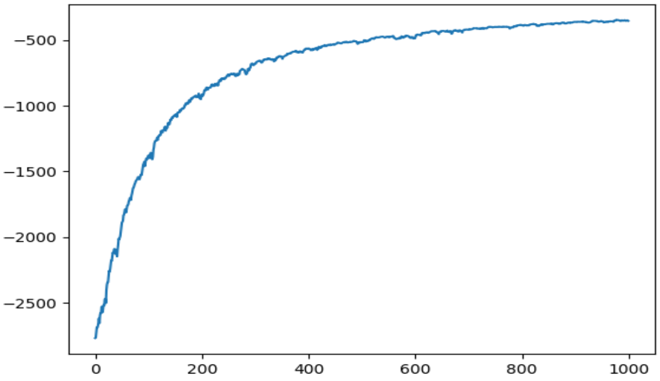
$$\begin{aligned} lgL(x;\beta_1,\beta_2,...,\beta_C) &= \sum_{i=1}^{n} lg\frac{\exp(\beta_c^T x_i)}{\sum_{j=1}^{C}\exp(\beta_j^T x_i)} \\ &= \sum_{i}^{n} lg\exp(\beta_c^T x_i) - lg(\sum_{j=1}^{C}\exp(\beta_j^T x_i)) \\ &= \sum_{i=1}^{n}(\beta_c^T x_i - lg\sum_{j=1}^{C}\exp(\beta_j^T x_i)) \end{aligned}$$

(b) The Python code is attached.

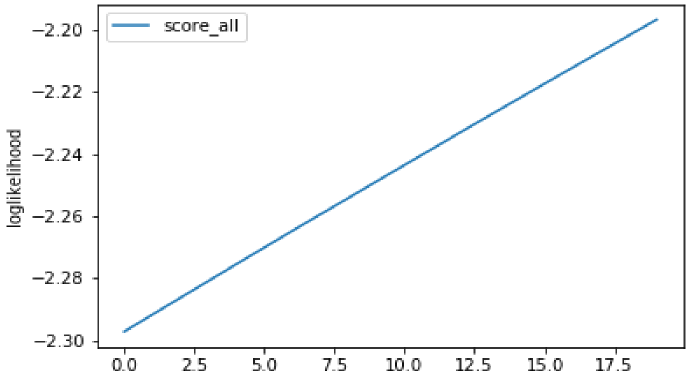
(c) for digit training set:its Gd log-likelihood plot:



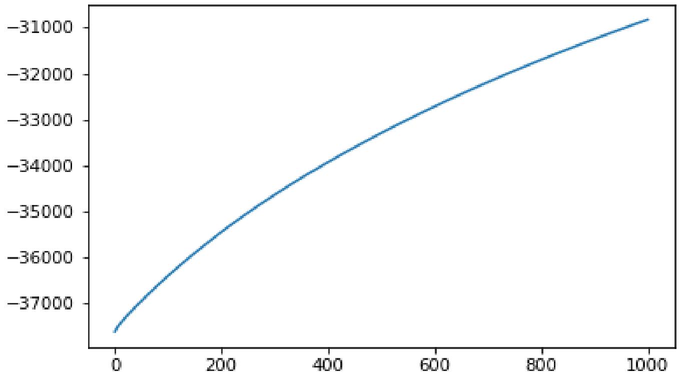
its Sgd log-likelihood plot:



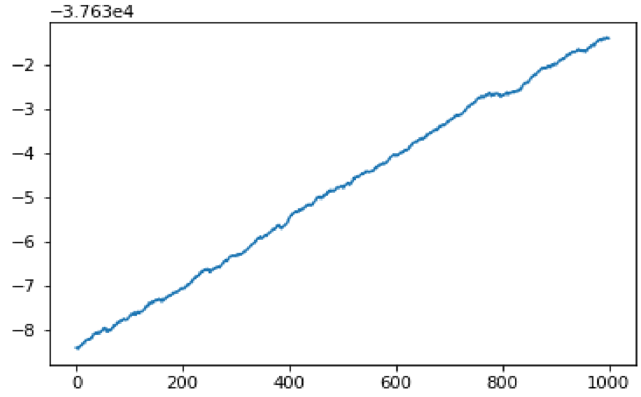
its Newton Raphson loglikehood plot(I change the epoch times from 1000 to 20 in this case to test if it works and normalize the data):



for news training set: its gd log-likelihood plot:

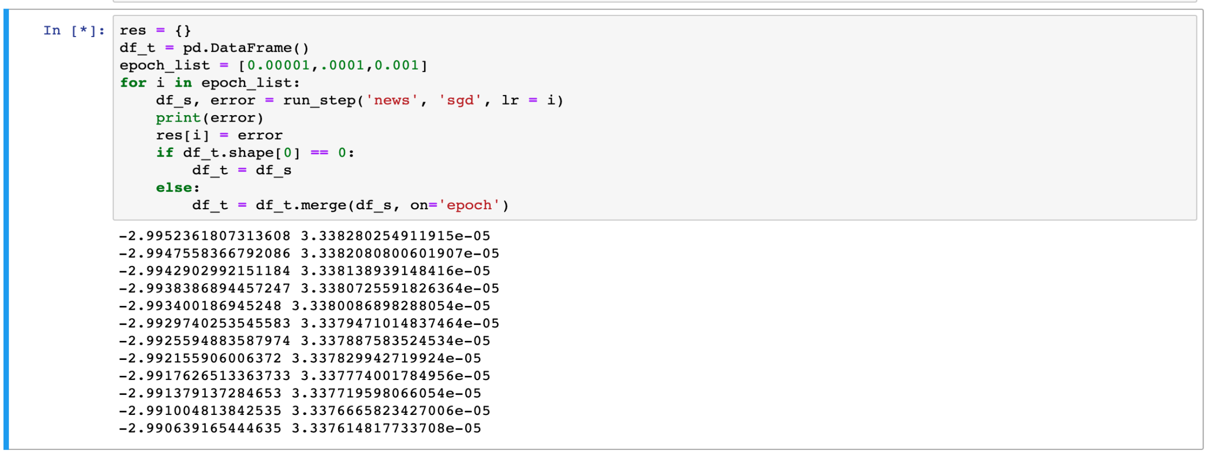


its sgd log-likelihood plot:



(d) For gradient decent, I respectively chose step size 0.03,0.001 and 1e-4 which the 0.03 fails to converge and 1e-4 converge slower than 0.001 but more accurate. So within the computing power and time consumption limit, I use 1e-4 to get the optimal answer.

(e) I use 1e-4 as the step size after using line search and comparing loss function based on fixed learning rate for different values.

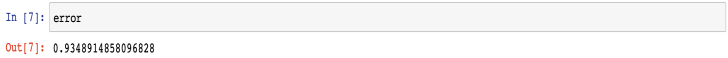


```
In [*]: res = {}
        df_t = pd.DataFrame()
        epoch_list = [0.00001,.0001,0.001]
        for i in epoch_list:
            df_s, error = run_step('news', 'sgd', lr = i)
            print(error)
            res[i] = error
            if df_t.shape[0] == 0:
                df_t = df_s
            else:
                df_t = df_t.merge(df_s, on='epoch')

        -2.9952361807313608 3.338280254911915e-05
        -2.9947558366792086 3.3382080800601907e-05
        -2.9942902992151184 3.338138939148416e-05
        -2.9938386894457247 3.3380725591826364e-05
        -2.993400186945248 3.3380086898288054e-05
        -2.9929740253545583 3.3379471014837464e-05
        -2.9925594883587974 3.337887583524534e-05
        -2.992155906006372 3.337829942719924e-05
        -2.9917626513363733 3.337774001784956e-05
        -2.991379137284653 3.337719598066054e-05
        -2.991004813842535 3.3376665823427006e-05
        -2.990639165444635 3.337614817733708e-05
```

(f) For the test error, I changed the iteration times of news dataset from 1000 to 100 to save time, while the small dataset keeps the iteration of 1000. The accuracy changes not a lot even though the iteration times is changed lower in the big news dataset.
Here is the accuracy of digits data:
sgd test acc: 0.92321; gd test acc: 0.96661
on the news data:
sgd test acc: 0.5571; gd test acc: 0.57479
I find out that in the small "digit" dataset, test error is very low - in other words, both GD and SGD work very well. Nevertheless, in the larger news dataset, neither GD nor SGD has good performance. For comparison, GD works better in both of the datasets.

(g) for the test error of Newton Raphson:



```
In [7]: error

Out[7]: 0.9348914858096828
```

In my program, the NR works better than SGD while not as good as GD in the small dataset of digit test.

All the codes are included in the zip file which conclude the ipynb file and hw1.pdf file.