به نام خدا

دانشگاه تهران

پردیس دانشکده‌های فنی

دانشکده برق و کامپیوتر

سیستم های هوشمند

تمرین شماره 1

حمیدرضا علی اکبری خویی

810196514

پاییز 99

فهرست

چکیده

این تمرین از ۴ بخش تشکیل شده است که هرکدام بخشی از درس را مورد پرسش قرار داده است. سوال اول در رابطه با نقطه ایستا و نوع آن سوال دارد که با توجه به نتایج ریاضی حاصل باید به آن پاسخ بدهیم. سوال دوم در رابطه با روش های جستجوی خط با چند متد که در درس به آنها اشاره شده است باید گام هارا پیدا کنیم تا بتوانیم در راستای بهینه کردن تابع خود قدم برداریم. در سوال سوم در رابطه با الگوریتم های SGD و BGD صحبت شده است که باید با توجه به هدف که کمینه کردن تابع است باید مدلسازی ای از این دو روش داشته باشیم تا درک بهتری از کلیت موضوع و نحوه کارکرد این روش برای بهینه ساختن یک سری ویژگی یک تابع به کار رفته است، در ادامه این سوال باید مقدار ضریب یادگیری را به گونه ای در طول یادگیری کم کنیم، چرا که باید در نهایت با کاهش طول گام به یک نقطه بهینه برای جواب خود برسیم. در سوال آخر با توجه به معیار clustering و تشخیص برچسب داده ها از روی داده های مجاور آنها باید مدلسازی داشته باشیم تا به این مهم دست پیدا کند، البته با یک سری بازنگاه هایی که در داده ها باید در نهایت داشته باشیم، باید الگوریتم KNN را بهبود بدیم.

سوال 1

برای بدست آوردن نقاط ایستا باید ماتریس گرادیان تابع را برابر صفر قرار دهیم و با بدست آوردن ماتریس hessian میتوانیم در رابطه با نوع نقطه ایستا تصمیم گیری کنیم.

$$f(x) = 3x_1^2 + 2x_2^2 - 3x_1x_2 + 4x_1^3 + x_1^4$$

$$\nabla f = \begin{bmatrix} \frac{\partial}{\partial x_1} \\ \frac{\partial}{\partial x_2} \end{bmatrix} f = \begin{bmatrix} 6x_1 - 3x_2 + 12{x_1}^2 + 4\,x_1^3 \\ 4x_2 - 3x_1 \end{bmatrix}$$

$$H(f) = \begin{bmatrix} \frac{\partial^2}{\partial x_1^2} & \frac{\partial^2}{\partial x_1 x_2} \\ \frac{\partial^2}{\partial x_2 x_1} & \frac{\partial^2}{\partial x_2^2} \end{bmatrix} f = \begin{bmatrix} 6 + 24x_1 + 12x_1^2 & -3 \\ -3 & 4 \end{bmatrix}$$

در مورد ماتریس هسین میشود با حساب کردن دترمینان این ماتریس راجع به نوع نقطه بحث کرد.

اگر گرادیان تابع را برابر صفر قرار دهیم داریم:

$$\nabla f = 0 \xrightarrow{concludes} \begin{cases} 6x_1 - 3x_2 + 12{x_1}^2 + 4\,x_1^3 = 0 \\ 4x_2 - 3x_1 = 0 \end{cases}$$

$$x_1 = 0\,, -\frac{3}{2} \pm \frac{\sqrt{21}}{4}\,, x_2 = \frac{3x_1}{4}$$

$$DH(x_1, x_2) = \det\big(H(f)\big) = 48x_1^2 + 96x_1 + 15 = 0$$

اگر مقدار دترمینان ماتریس hessian مثبت باشد و مقدار تابع در حول آن نقطه مثبت باشد، مقدار تابع در آن نقطه کمینه محلی است، اگر دترمینان مثبت باشد ولی مقدار تابع منفی باشد، نقطه، نقطه بیشینه محلی است؛ اگر مقدار ماتریس hessian صفر بشود، نقطه حاصل Saddle point است. پس:

$$\begin{cases} DH(0,0) > 0, \quad f > 0 \ : & local\ minimum \\ DH\left(-\frac{3}{2} - \frac{\sqrt{21}}{4}\,, -\frac{9}{8} - \frac{3\sqrt{21}}{16}\right) > 0, f > 0 : & local\ minimum \\ DH\left(-\frac{3}{2} + \frac{\sqrt{21}}{4}\,, -\frac{9}{8} + \frac{3\sqrt{21}}{16}\right) = 0 : & Saddle\ point \end{cases}$$

سوال 2

$$f(x) = 3x_1^2 + 2x_1 + 8x_2 + 4x_2^2$$

بخش اول:

$$-\nabla f(x_1, x_2) = \begin{bmatrix} -6x_1 - 2 \\ -8 - 8x_2 \end{bmatrix}_{(0,0)} = \begin{bmatrix} -2 \\ -8 \end{bmatrix}$$

بخش دوم:

$$-\nabla f(x_1, x_2) = \begin{bmatrix} -6x_1 - 2 \\ -8 - 8x_2 \end{bmatrix}_{(0,0)} \stackrel{\text{def}}{=} \begin{bmatrix} r_1 \\ r_2 \end{bmatrix} = r$$

$$f(x + \lambda r) = 3(x_1 + \lambda r_1)^2 + 2(x_1 + \lambda r_1) + 8(x_2 + \lambda r_2) + 4(x_2 + \lambda r_2)^2$$

$$\frac{\partial f(x + \lambda r)}{\partial \lambda} = 6r_1(x_1 + \lambda r_1) + 2r_1 + 8r_2 + 8r_2$$

$$\lambda = \frac{6r_1x_1 + 2r_1 + 8r_2 + 8r_2x_2}{6r_1^2 + 8r_2^2} \rightarrow in\ (x_1, x_2) = (0,0) \rightarrow \lambda = \frac{2r_1 + 8r_2}{6r_1^2 + 8r_2^2}$$

$$(r_1, r_2) = (-2, -8) \rightarrow \lambda = 0.12686$$

$$new\ x_1 = x_1 + \lambda r_1 \rightarrow new\ x_1 = -0.25373$$

$$new\ x_2 = x_2 + \lambda r_2 \rightarrow new\ x_2 = -1.01492$$

بخش سوم:

$$f(x_k + \lambda p_k) \leq f(x_k) + c\lambda \nabla f_k^T p_k$$

$$optimum\ \lambda\ is\ in \frac{1}{|\nabla f|} = \frac{1}{8.2} = 0.121\ , c = 10^{-4} (Based\ on\ Nocedal\ Wright)$$

$$f(x_0 + \lambda p_0) = f(-0.3, -1.1) = -4.29$$

$$f(x_0) + c\lambda \nabla f_0^T p_0 = 0 + 10^{-4}.0.121\ .-8.2 = -0.0085 \rightarrow -4.29 < -0.00085$$

پس شرط آرمیجو صادق است و:

$$new\ x_1 = x_1 + \lambda r_1 \rightarrow new\ x_1 = -0.242$$

$$new\ x_2 = x_2 + \lambda r_2 \rightarrow new\ x_2 = -0.968$$

همانطور که مشاهده میشود برای آرمیجو نزدیک نقطه اپتیمم قبلی بود ولی یکم اختلاف داشت آن هم برای این است که برای آرمیجو روش جستجوی خط باید فقط در یک نا مسوای صدق کند و شرط های محکم تری مثل مساوی وجود ندارد.

بخش چهارم:

Steepest Decent:

$$f(x+\lambda r) = 3(x_1+\lambda r_1)^2 + 2(x_1+\lambda r_1) + 8(x_2+\lambda r_2) + 4(x_2+\lambda r_2)^2$$

$$\frac{\partial f(x+\lambda r)}{\partial \lambda} = 6r_1(x_1+\lambda r_1) + 2r_1 + 8r_2 + 8r_2$$

$$\lambda = \frac{6r_1x_1 + 2r_1 + 8r_2 + 8r_2x_2}{6r_1^2 + 8r_2^2} \rightarrow in\ (x_1, x_2) = (-0.25373, -1.01492)$$

$$(r_1, r_2) = (-0.47762, 0.11936) \rightarrow \lambda = 0.2334$$

$$new\ x_1 = x_1 + \lambda r_1 \rightarrow new\ x_1 = -0.3652$$

$$new\ x_2 = x_2 + \lambda r_2 \rightarrow new\ x_2 = -1.0534$$

Armijo:

$$f(x_k + \lambda p_k) \leq f(x_k) + c\lambda \nabla f_k^T p_k$$

$$optimum\ \lambda\ is\ in \frac{1}{|\nabla f|} = \frac{1}{0.46} = 2.136\ , c = 10^{-4} (Based\ on\ Nocedal\ Wright)$$

$$f(x_0 + \lambda p_0) = f(-0.3, -0.4) = -2.89$$

$$f(x_0) + c\lambda \nabla f_0^T p_0 = 0 + 10^{-4}.2.136\ .-0.46 = -0.00098 \rightarrow -2.89 < -0.00098$$

پس شرط آرمیجو صادق است و:

$$new\ x_1 = x_1 + \lambda r_1 \rightarrow new\ x_1 = -0.341$$

$$new\ x_2 = x_2 + \lambda r_2 \rightarrow new\ x_2 = -1.013$$

سوال3

محاسبه هایپر پارامتر

برای محاسبه هایپر پارامتر از روش K-fold Cross validation رفته ام. این روش به گونه ای عمل میکند که کل داده هارا به k قسمت تقسیم میکند و به مدت k بار، هر بار با استفاده از k-1 قسمت مدل را بر حسب مقادیر اولیده هایپر پارامتر آموزش میدهد و آن را بر روی یک دسته باقی مانده تست میکند و میزان دقت را استخراج میکند؛ این کار را بعد از انجام K بار برای هر هایپر پارامتر با جایگشتی که روی کل داده ها میزنیم ، میانگین دقت را حساب میکنیم و برای هر قسمت از هایپر پارامتر اولیه انتخاب شده میزان دقت را ذخیره میکنیم و در نهایت آن مقدار اولیه ای که در مجموع و در میانگین بیشترین دقت را داشته است انتخاب میشود.

```
accuracy of 5-fold cross validation for lmbda= 1e0 is : 90.965 %
accuracy of 5-fold cross validation for lmbda= 1e-2 is : 9.035 %
accuracy of 5-fold cross validation for lmbda= 1e-4 is : 9.035 %
accuracy of 5-fold cross validation for lmbda= 1e-6 is : 9.035 %
accuracy of 5-fold cross validation for lmbda= 1e-8 is : 9.035 %
accuracy of 5-fold cross validation for lmbda= 1e-10 is : 9.035 %
```

Figure 1

با توجه به شکل یک میزان دقت برای هایپر پارامتر های مختلف با استفاده از روش توضیح داده شده آمده است که مشاهده میشود میزان دقت برای ۲۰ بار iteration و طول گام 0.1 برای مقدار هایپر پارامتر "۰.۰۱" بیشینه است. البته این امر نیز کمی از قبل میشود حدس زد که برای هایپر پارامتر های با مقدار خیلی کم به خاطر فرابرازش و حتی کمبودن سرعت همگرایی به خاطر فرم اصلی تابع گرادیان، میزان دقت در این مقادیر کم خواهد بود.

One VS All

این روش برای این است که وقتی در SGDیا BGD بیشتر از یک نوع برچسب داشتیم باید مدل را برای هر نوع برچسب جداگانه آموزش دهیم و در نهایت برای آزمودن داده های test باید این گونه عمل کنیم که هر یک از داده ها در هر کدام از تخمین ها به کار رفت آن را با آن کلاس تخمین طبقه بندی کنیم. این روش یک بدی که دارد این است که با توجه به امکان فرابرازش بر روی داده ها یک regulation ای بر حسب تخمین داده شده و بر روی ماتریس وزن ها انجام میدهیم که در تعداد داده های بسیار بزرگ مدل ما دچار فرابرازش نشود.

SGD

در این روش که نوع خاصی از روش روش گرادیان نزولی است، به گونه ای است که در هر بار برای آموزش دادن مدل خودمان باید یک داده را به طور تصادفی انتخاب بکنیم و مدل را بر حسب آن آموزش دهیم و در هرepoch تا جاییی این کار را انجام میدهیم که داه ای باقی نماند و در ادامه برای بروز رسانی ماتریس های ضرایب آموزش داده شده در epoch

بعدی بجای این که یک initialization ای از ماتریس های ضرایب داشته باشیم، ماتریس های ضرایب بدست آمده در epoch قبلی را به صورت ورودی به سیتم میدهیم تا در این epoch آن ماتریس ها را بهبود ببخشد. میزان دقت در این مدل را میتوان در هر epoch با تست کردن ماتریس های ضرایب بدست آمده بر روی داده های test میزان دقت این مدل را حساب کرد. برای بیان حسرت یا کمبود و نقص مدل نیز میتوان در هر epoch مقدارloss را بر حسب ماتریس های ضرایب بروز شده حساب کرد که این مقدار نشان دهنده آن است که در طول یادگیری، اگر مدل در نهایت به یک optimum بخواهد برسد باید به ازای افزایش epoch ها مقدار loss کاهش بیابد. برای دقت نیز باید به ازای افزایش تعداد epoch ها میزان دقت سیستم افزایش بیابد.

$$Loss\ function = \lambda||w||^2 + \frac{1}{n}\sum_i \max(0,1 - y_i(wx_i - b))$$

$$\frac{\partial}{\partial w}\text{loss} = 2\lambda w + \begin{cases} 0; y_i(w_i x_i - b) \geq 1 \\ -y_i x_i;\ y_i(w_i x_i - b) < 1 \end{cases}$$

برای بروز رسانی وزن ها و بایاس داریم:

$$w = \begin{cases} w - 2\alpha\lambda w; & y_i(w_i x_i - b) \geq 1 \\ w - 2\alpha\lambda w + \alpha y_i x_i; & y_i(w_i x_i - b) < 1 \end{cases}$$

$$b = b - y_i\ only\ if\ y_i(wx_i - b) < 1$$

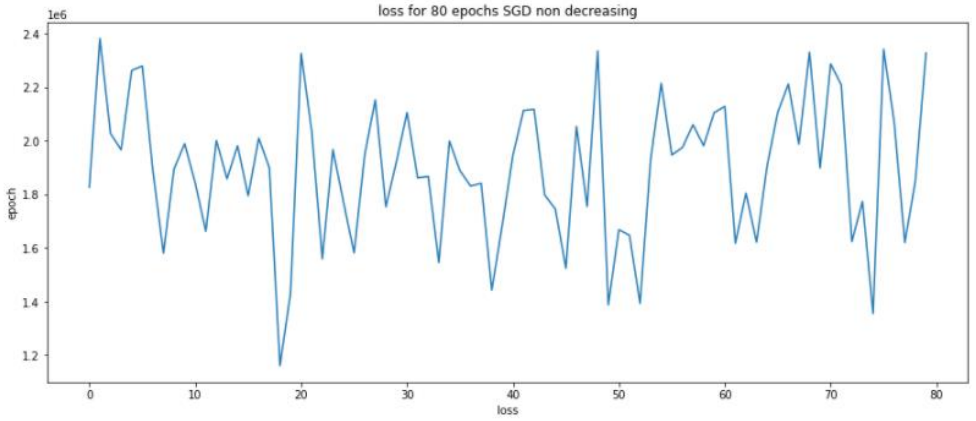


Figure ۲

از شکل شماره دو معلوم است که من طول گام را خوب نگرفته ام و مدل هنوز نمیتواند به نقاط با نرژی کمتر در مدلسازی برود، البته اعداد کمی را امتحان کردم. اگر نمودار درست مسبود باید در رراستای تعداد epoch مقدار loss کمتر میشد.

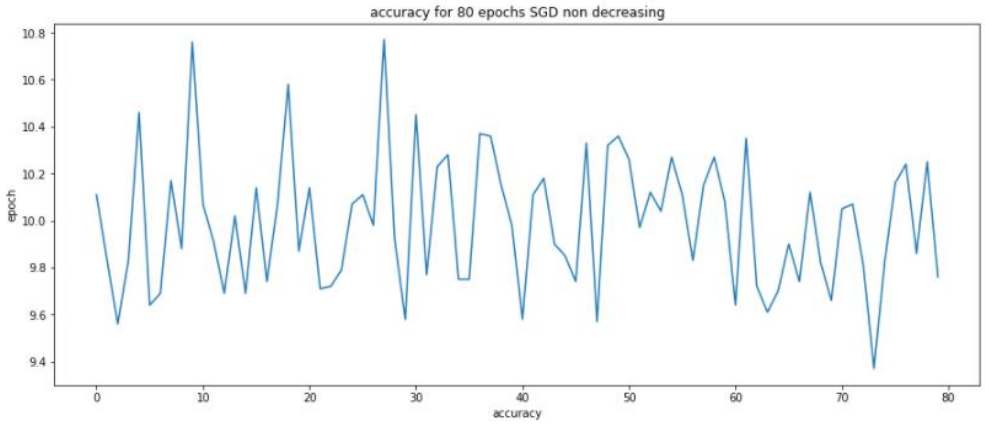


Figure 3

در شکل سه هم مشاهده میشود همچنین برای این شکل هم چون طول گام به اندازه کافی کوچ نبود و بزرگ بود این اتفاق افتاده است و اگر شکل درست میبود و طول گام به اندازه کافی بود باید در راستای افزایش تعداد epoch مقدار دقت افزایش میافت.

BGD

مانند روش SGD است که فقط بجای این که داده هارا تکی تکی بدهیم داده هارا دسته دسته میکنیم و بعد ماتریس های ضرایب را آپدیت خواهیم کرد به گونه ای که این روش باعث میشود سرعت همگرایی روش به مراتب با توجه به اندازه دسته هایی که در نظر گرفته ایم بیشتر بشود, احتمال بسیار کمتری نسبت به روش قبل دارد که در جهت خلاف گرادیان هم این روش حرکت کند به همین خاطر معمولا این روش بهتر نیز عمل میکند و در عمل میزان loss کمتری از خود نمایان میسازد و در نهایت به درصد دقت بیشتری نیز خواهد رسید.

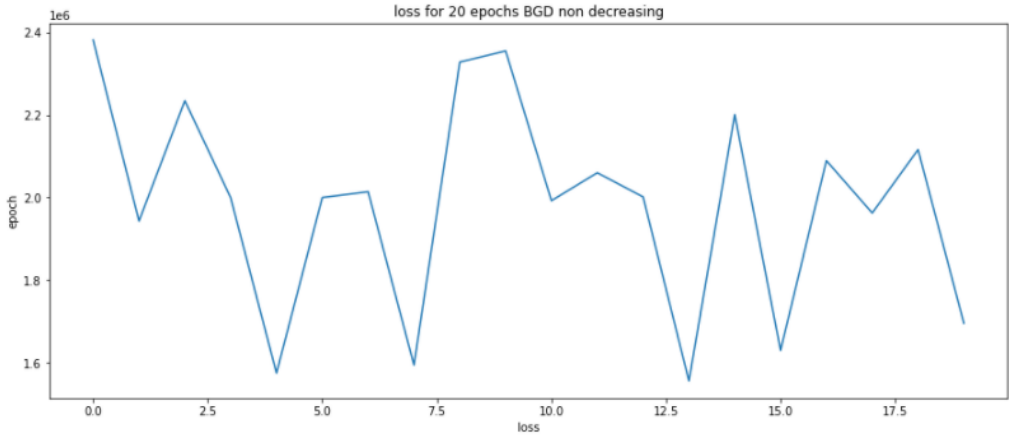
$$Loss\ function = \lambda||w||^2 + \frac{1}{n}\sum_i \max(0,1 - y_i(wx_i - b))$$

$$\frac{\partial}{\partial w}\text{loss} = 2\lambda w + \begin{cases} 0; y_i(w_i x_i - b) \geq 1 \\ -y_i x_i;\ y_i(w_i x_i - b) < 1 \end{cases}$$

برای بروز رسانی وزن ها و بایاس داریم:

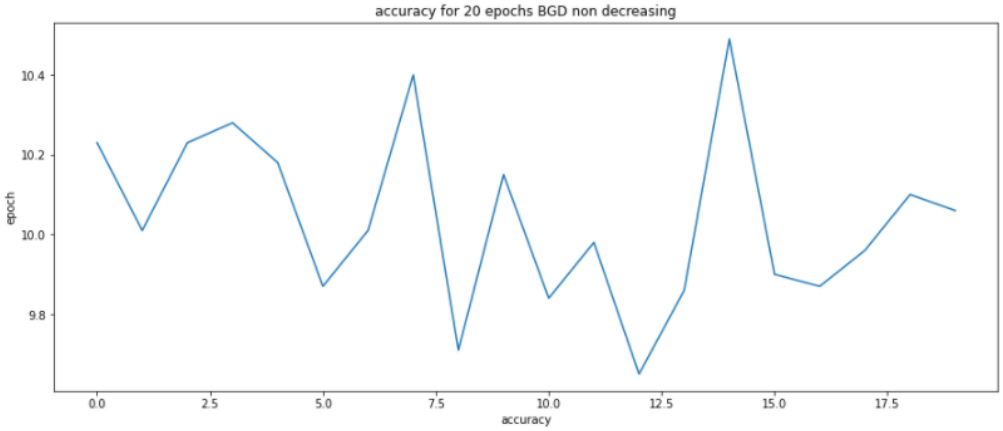
$$for\ any\ k\ in\ bathsize \begin{cases} w = \begin{cases} w - 2\alpha\lambda w; & y_i^k(w_i x_i^k - b) \geq 1 \\ w - 2\alpha\lambda w + \alpha y_i x_i; & y_i^k(w_i x_i^k - b) < 1 \end{cases} \end{cases}$$

$$b = b - y_{\frac{i}{batch_{size}}}\ only\ if\ y_i(wx_i - b) < 1$$



4Figure

از شکل شماره 4 معلوم است که من طول گام را خوب نگرفته ام و مدل هنوز نمیتواند به نقاط با نرژی کمتر در مدلسازی برود، البته اعداد کمی را امتحان کردم. اگر نمودار درست مسبود باید در رراستای تعداد epoch مقدار loss کمتر میشد. البته چون از حالت Batch استفاده میکنم هم باید در نمودار شاهد نرمی بیشتری باشیم چون اصولا این کار باعث نرمی میشود.



5Figure

در شکل ۵ هم مشاهده میشود همچنین برای این شکل هم چون طول گام به اندازه کافی کوچ نبود و بزرگ بود این اتفاق افتاده است و اگر شکل درست میبود و طول گام به اندازه کافی بود باید در راستای افزایش تعداد epoch مقدار دقت افزایش میافت. البته چون از حالت Batch استفاده میکنم هم باید در نمودار شاهد نرمی بیشتری باشیم چون اصولا این کار باعث نرمی میشود.

Decreasing mode

روش من برای کاهش طول بردارد یادگیری به این گونه بوده که اندازه بردار را بعد از تعداد مشخصی epoch تقسیم بر 5 میکردم، البته اول باید یکم فصرت یادگیری به ماشین بدهیم و بعد طول گام را کاهش بدهیم تا سریع تر همگرا بشود و در نقاط مینیمم محلی گیر نکند.

SGD

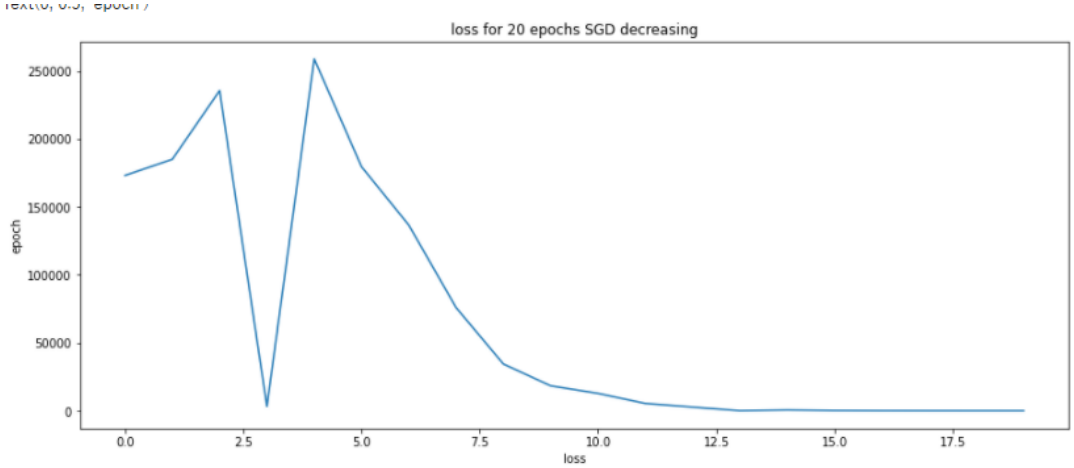
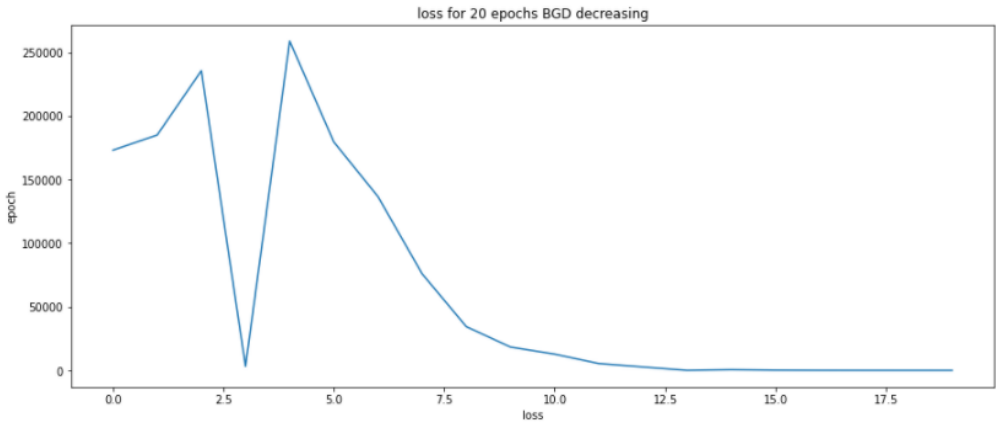


Figure 6

با توجه به شکل ۶ چون من در طول افزایش epoch مقدار طول گام را کاهش دادم این کار باعث شده تا طول گام اولیه ای که در نظر گرفته بودم و به اندازه کافی کوچک نبود در اثر کاهش یافت به مقدار اپتیمم خود برسد و باعث بشود که مقدار LOSS مدل کمتر بشود پس در کل عملکرد مدلسازی برنامه درست است و فقط در طول گام مشکل داشتیم.

BGD



7Figure

با توجه به شکل هفت که مربوط به حالت batch هست، مانند شکل 6 چون طول اولیه گام به اندازه کافی کوچک نبود باعث شده بود تا در epochهای اول یادگیری خوبی نداشته باشیم ولی چون در اثر افزایش epoch مقدار طول گام

کاهش یافت باعث شد تا مقدار loss کل این مدل کمتر بشود و این نشان دهنده این است که در کل سیستم از نظر طول اولیه گام مشکل داشت.

سوال 4

قسمت اول

Correctness of KNN for norm 2 and k = 2 is; [15.65714286]%

Correctness of KNN for norm 2 and k = 5 is; [14.77142857]%

Correctness of KNN for norm 2 and k = 10 is; [13.88571429]%

Correctness of KNN for norm 2 and k = 50 is; [14.28571429]%

8Figure

با توجه به شکل ۸ میزان دقت برای k=2و5 بیشتر از همه است. البته باید توجه داشت که دقت پایین به این دلیل است که بعد های ویژگی موجود در داده ها نرمالیزه نشده اند و از نظر اندازه با هم همخوانی ندارند به خاطر همین بعد از نرمال کردن هر یک از ویژگی ها دوباره این تست را گرفتم که نتیجه را در شکل 9 مشاهده میکنید:

Correctness of KNN for norm 2 and k = 2 is; [76.68571429]%

Correctness of KNN for norm 2 and k = 5 is; [76.22857143]%

Correctness of KNN for norm 2 and k = 10 is; [74.34285714]%

Correctness of KNN for norm 2 and k = 50 is; [62.28571429]%

Figure 9

مطابق شکل 9 میزان دقت همچنان برای k=2,5 بیشتر از بقیه است با توجه نوع قرار گرفتن داده ها و میزان پراکندگی و فرکانس تغییرات هر داده این مقدار بهینه k برای هر داده ای فرق خواهد داشت و اینجا به مقدار 2یا5 برای مقدار تقریبا بهینه رسیده ایم.

قسمت دوم

برای این قسمت مدل را برای k =1 به خاطر کوچک بودن و k=100 به خاطر بزرگ بودن امتحان کردیم و نتیجه را در شکل 10 میتوان مشاهده کرد.

```
[15] knn(1, 1, ntrain_data, ntest_data, train_labels, test_labels)

     77.51428571428572

[16] knn(100, 1, ntrain_data, ntest_data, train_labels, test_labels)

     54.028571428571425
```

Figure 10

میتوان شکل 10 برداشت را داشت که با توجه به نزدیک بودن توابع هم خوان به یکدیگر هر چقدر مقدار k بیشتر میشود از دقت مدل کم میشود و درحقیقت میتوان به این نتیجه رسید که داده ها تقریبا clustering هستند و خوشه ای جمع شده اند؛ به ازای افزای k های خیلی بزرگ چون داده های سایر خوشه هارا نیز دخالت میدهیم باعث میشود که مقدار دقت پایین بیاید و هرچقد مقدار k را کمتر بکنیم میزان دقت افزایش میابد.

قسمت سوم

```
Correctness of KNN for norm:1 and k = 2 is; [76.68571429]%

Correctness of KNN for norm:2 and k = 2 is; [54.25714286]%

Correctness of KNN for norm:3 and k = 2 is; [77.11428571]%
```

11Figure

مطابق شکل 11 مشاهده میشود که برای متد های d_1, d_2 که در قسمت سوم سوال معرفی شد برای k بدست امده در سوال های قبل مقادیر دقت بالا بدست آمده است که با توجه به اعداد مدلی اقلیدسی و مدل نرم 1 با قدر مطلق فاصله در رابطه است بیشترین دقت ها را دارند.

قسمت چهارم

همانطور که در قسمت های ابتدایی هم گفته شد به دلیل قابل مقایسه نبودن بعد ها نسبت به هم علی الخصوص 4 بعد ویزگی آخر، باید هر بعد ویژگی را نرمای میکردیم یا آن بعد وِزگی را به اعداد بین دو عدد انتخابی a,b نگاشت میدادیم و این عمل را برای بقیه بعد ها نیز انجام میدادیم. نتیجه انجام عمل در قسمت یک این سوال در شکل شماره 9 آمده است و بعد از نرمال کردن بقیه قسمت های این سوال با نرمال شده ویژگی های این داده ها انجام گرفته است.

نحوه اجرای برنامه

برای سوال سوم فایل IS_HW1_Q3.ipynb را باید در بستر Jupyter اجرا بکینم و البته من داده های Train,Test ر در Google Drive خود گذاشته بودم تا با سرویس Colab برنامه نویسی بکنم. فایل ها در پوشه IS_HW1 در Google Drive باید باشد تا اجرا بشود.

برای سوال چهارم فایل IS_HW1_Q4.ipynb را باید در بستر Jupyter اجرا بکینم و البته من داده های Train,Test ر در Google Drive خود گذاشته بودم تا با سرویس Colab برنامه نویسی بکنم. فایل ها در پوشه IS_HW1 در Google Drive باید باشد تا اجرا بشود.

منابع و مراجع

1.Nocedal & Write, Optimization problems, 2008