بسمه تعالی

دانشکده مهندسی کامپیوتر

مهر ۱۴۰۰

یادگیری عمیق

نام استاد: دکتر محمدی

تمرین دوم

آرمان حیدری

شماره دانشجویی: ۹۷۵۲۱۲۵۲

۱. پاسخ سوال اول

الف) داشتن کار (x_2) را اگر 1 باشد به منزله داشتن کار و اگر 0 باشد به منزله نداشتن کار در نظر می‌گیریم. و همچنین توان پرداخت را به همین صورت 0 و 1 می‌گیریم.

first epoch:

$$datas = \begin{cases} x_1 = 22, x_2 = 1, y = 0 \\ x_1 = 25, x_2 = 0, y = 0 \end{cases}$$

forward:

$$\hat{y}_1 = \sigma(w_1 x_1^{(1)} + w_2 x_2^{(1)} + b) = \sigma(22 + 1 + 1) = \frac{1}{1 + e^{-24}} = \simeq 0.999999999994$$

$$\hat{y}_2 = \sigma(w_1 x_2^{(2)} + w_2 x_2^{(2)} + b) = \sigma(25 + 0 + 1) = \frac{1}{1 + e^{-26}} = \simeq 0.999999996$$

$$L = \frac{1}{N} \sum_{i=1}^{N} L_i(s_i, y_i) = \frac{-1}{2} (\log loss(y_1, \hat{y}_1) + \log loss(y_2, \hat{y}_2)) \simeq 36.067$$

backward:

$$a = \hat{y} \Rightarrow \frac{dL(a,y)}{da} = \frac{-y}{a} + \frac{1-y}{1-a}$$

$$z = w_1 x_1 + w_2 x_2 + b \rightarrow \frac{da}{dz} = \frac{-(-e^{-z})}{(-1 + e^{-z})^2} = \frac{1}{1 + e^{-z}} \times \frac{e^{-z} + 1 - 1}{1 + e^{-z}} = \sigma(z) \times (1 - \sigma(z)) = a(1 - a)$$

$$\frac{dL}{dz} = \frac{dL}{da} \times \frac{da}{dz} = \left(\frac{-y}{a} + \frac{1-y}{1-a}\right)(a)(1-a) = a-y$$

$$\frac{dL}{dw_1} = \frac{dL}{dz} \times \frac{dz}{dw_1} = \boxed{(a-y)x_1}$$

$$\frac{dL}{dw_2} = \boxed{(a-y)x_2} \qquad \frac{dL}{db} = \boxed{a-y}$$

حالا که روابط گرادیان را به دست آوردیم، به سراغ وزن ها را آپدیت می کنیم: learning rate

$$w_1 = w_1 - 0.05 \times \frac{dL}{dw_1} = 1 - 0.05 \times \sum_{i=1}^{N} (\hat{y}_{(i)} - y_{(i)}) x_1 =$$

$$= 1 - 0.05 \times \left((\hat{y}_1 - y_1)\,22 + (\hat{y}_2 - y_2)\,25\right) = \boxed{-1.35}$$

$$b = b - 0.05 \times (\hat{y}_1 - y_1 + \hat{y}_2 - y_2) = \boxed{0.90}$$

$$w_2 = w_2 - 0.05 \times \left((\hat{y}_1 - y_1)\,x_2^{(1)} + (\hat{y}_2 - y_2)\,x_2^{(2)}\right) \simeq \boxed{0.95}$$

حال باید batch بعدی را برداریم که طبق صورت سوال دو داده بعدی است و چون تمام کارها برای 2 تا iteration کامل می خواهیم، در واقع همه این مراحل باید 4 بار در هر iteration و در کل 8 بار تکرار شود. که این محاسبات را کامپیوتری انجام داده و نتایج آپدیت وزن ها به این صورت بود:

w_1	w_2	b	
-1.35	0.95	0.90	داده های اول و دوم:
1	1	0.95	" " سوم و چهارم:
1	1	0.95	" " پنجم و ششم:
-1.75	1.0	0.90	" هفتم و هشتم: ← epoch 2
-1.75	1	0.9	داده های اول و دوم:
0.6	1.05	0.95	⋮
0.6	1.05	0.95	
-2.15	1.05	0.9	

آغاز سال ۲۰۱۸ میلادی

ب)

- در روش stochastic احتمال گیر کردن الگوریتم در مینیمم های محلی بسیار کمتر است چون احتمال این که یک مینیمم محلی برای همه مقدار های ورودی برسد کم است. درحالی که در حالت عادی احتمالش هست.

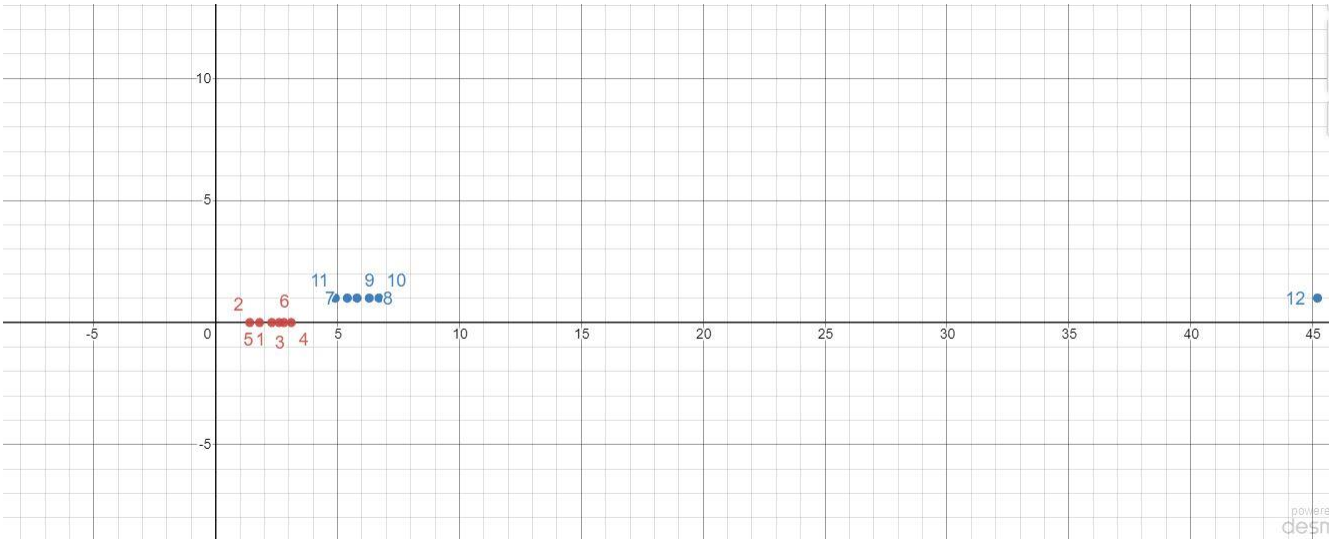
- شاید تابع ضرر در یک راستا تغییرات سریعتری نسبت به راستای دیگر داشته باشد، که در این صورت هردو در یک راستا خیلی آهسته پیش می روند و در یک راستای دیگر نوسان می کنند. راه حل این مشکل می تواند چند بعدی کردن learning rate باشد. تا اندازه گام ها متناسب با میزان تغییرات باشد.

- گرادیان ها در حالت تصادفی نویزی هستند و در واقع با نوسان به سمت نقطه بهینه حرکت می کنیم. اما در حالت عادی حرکتمان صاف است. البته جهت کلی درست است.
- فرق واضح این دو هم کوتاه تر بودن مدت هر epoch در حالت تصادفی است و لذا سریع تر به مقصد می رسیم، یا این که با iteration های بیشتر زودتر به همگرایی می رسیم.
- هردوی آن ها این مشکل را دارند که اگر در ابتدا سرعت بیشتری داشته باشند و کم کم سرعتشان کاهش پیدا کند، می توانند بهتر و زودتر به نتیجه دلخواه برسند. و این قضیه را با الگوریتم های بهینه سازی مختلف مثل adam یا schedule کردن learning rate حل می کنیم.

منابع: اسلاید جلسه چهارم استاد، وبسایت bogotobogo

۲. پاسخ سوال دوم:

ابتدا داده‌ها را رسم میکنیم:



برای linear regression، اگر خطی با رابطه $y' = bx+a$ داشته باشیم، میتوانیم a و b را با رابطه زیر به دست آوریم:

$$a = \frac{(\sum y)(\sum x^2) - (\sum x)(\sum xy)}{n(\sum x^2) - (\sum x)^2}$$

$$b = \frac{n(\sum xy) - (\sum x)(\sum y)}{n(\sum x^2) - (\sum x)^2}$$

پس جدولی تشکیل می‌دهیم تا این موارد را به دست آوریم:

n	x	y	x*y	x^2	y^2
1	2.3	0	0	5.29	0
2	1.4	0	0	1.96	0
3	2.6	0	0	6.76	0
4	3.1	0	0	9.61	0
5	1.8	0	0	3.24	0
6	2.8	0	0	7.84	0
7	5.4	1	5.4	29.16	1
8	6.3	1	6.3	39.69	1

9	5.8	1	5.8	33.64	1
10	6.7	1	6.7	44.89	1
11	4.9	1	4.9	24.01	1
12	45.2	1	45.2	2043	1
sum	**88.3**	**6**	**74.3**	**2249.1**	**6**

پس داریم:

a = 0.36 , b = 0.018 ➔ y' = 0.018x + 0.36 ➔0.5 = 0.018x+0.36 ➔ x=7.78

همه این محاسبات به صورت خودکار نیز در فایل Q2.ipynb انجام شده و پیوست شده است. همچنین مدل logistic regession را با استفاده از sklearn، وزن ها و پایه‌اش را حساب میکنیم.

که به این صورت به دست می‌آید:

```
w = [1.43516077]
b = -5.924735403729955
```

حال برای محاسبه X که مرز تصمیم باشد:

0.5 = sigmoid(1.43*x -5.92) ➔ ½ = 1/1+e^(5.92-1.43x) ➔ 5.92-1.43x = 0 ➔ x = 5.92/1.43 = 4.14

که x ها اعداد نزدیک به همی به دست نیامدند چون شکل دو تابع بسیار متفاوت است. حالا دقت این دو روش را مقایسه میکنیم و میبینیم که مطابق انتظار دقت logistic regression بسیار بیشتر است.

```
linear score: 0.18945201475354911
logistic score: 0.9166666666666666
```

چون جداسازی این داده ها با یک خط دقت خوبی ندارد و linear اصولا برای مسائل classification مناسب نیست. در حالی که لجستیک با توابع فعالسازی مختلفی که میتواند داشته باشد و شکل بسیار منعطف تر از خطی میتواند به ما کمک کند. البته linear regression هم در برخی مسائل، به خصوص مسائل از نوع تخمین زدن کاربردی است.

منابع: وبسایت statisticshowto

۳. پاسخ سوال سوم:

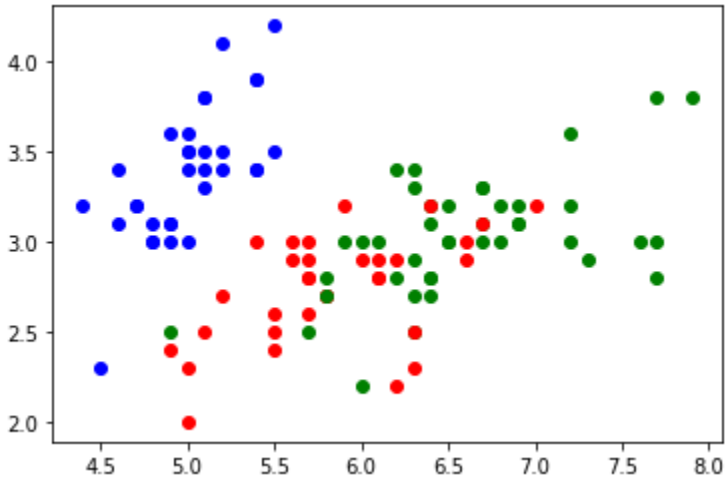
پاسخ این سوال در فایل Q3.ipynb پیوست شده است. و همچنین در گوگل‌کولب موجود است. اما بخش های توضیحی و پاسخ‌هایی خواسته بودید که در ادامه پاسخ خواهم داد.

الف) Iris دیتاستی است که ۴ ویژگی از ۱۵۰ زنبق را در بر دارد. این ویژگی‌ها طول و عرض گلبرگ‌ها و کاسبرگ‌ها هستند. همچنین label های این دیتاست، سه نوع مختلف از زنبق است. و درواقع هدف این است که شبکه‌ای داشته باشیم که با ورودی گرفتن طول و عرض کاسبرگ و گلبرگ، نوع زنبق را تعیین کند.

در این دیتاست ۵۰ داده از هر نوع وجود دارد و داده تستی موجود نیست. و در صورت نیاز (مانند همین سوال) باید بخشی از داده را خودمان به صورت تست در نظر بگیریم و در آموزش شبکه از آن‌ها استفاده نکنیم.

این دیتاست تحت کتابخانه scikit-learn است. و استفاده از آن رایگان است. اما حجم داده‌های آن کم است و دیتاست سبکی محسوب می‌شود.

ب) در نوت‌بوک رسم کرده‌ام و به این شکل شد:



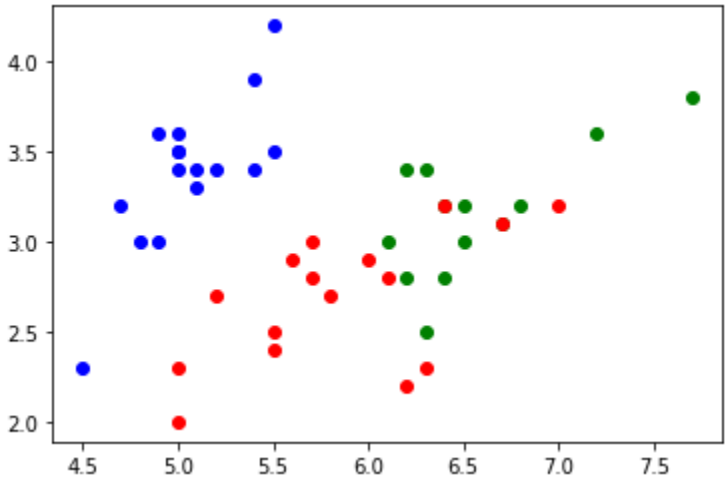
داده‌های آبی: کلاس اول، داده‌های قرمز: کلاس دوم، داده‌های سبز: کلاس سوم

محور افقی: ویژگی اول (طول کاسبرگ)، محور عمودی: ویژگی دوم (عرض کاسبرگ)

*ویژگی های سوم و چهارم در حل این سوال به کل در نظر گرفته نشده‌اند.

مهم‌ترین نکته‌ای که می‌توانیم بفهمیم این است که با این ویژگی‌ها، داده‌های کلاس اول به راحتی قابل تفکیک هستند اما داده‌های کلاس دوم و سوم احتمالا خطا دارند. هر چند می‌توان آن‌ها را نیز تا حد قابل‌قبولی تفکیک کرد.

ج) مشابه قسمت قبل است چون داده‌های تست و آموزش فرق خاصی ندارند و فقط نسبت ۷۰ به ۳۰ رندوم هستند.

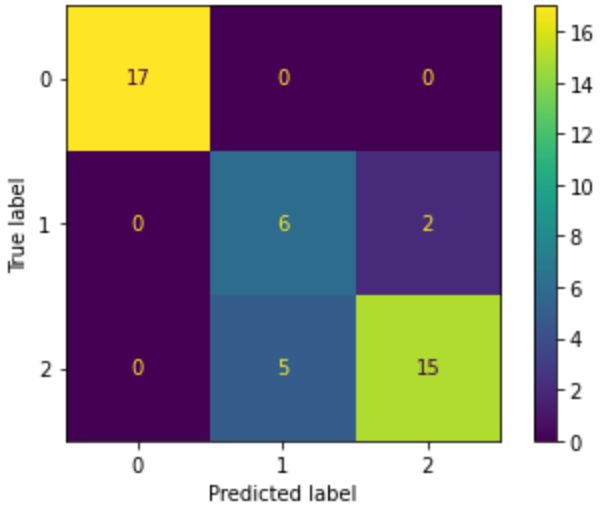


د) حدودا هر دو ۸۰ درصد هستند. اما چون به صورت رندوم انتخاب می‌شود که کدام داده در train و کدام در test باشد، با هر بار اجرای کد کمی خروجی دقت شبکه متفاوت است.

```
train accuracy: 0.8476190476190476
test accuracy: 0.8444444444444444
```

برای الگوریتم logistic regression که ساده است این دقت بدی نیست، اما هرچقدر تعداد iteration ها را زیاد میکنیم میبینیم که دقت تغییر خاصی نمی‌کند. این یعنی با این شبکه و این ورودی‌ها نمیتوانیم به دقت فوق‌العاده‌ای دست پیدا کنیم. همچنین نزدیک بودن دقت train و test نشان میدهد که شبکه دچار overfit نشده است و با توجه به دقت معقول آموزش یعنی underfit هم نداشته‌ایم.

ه) confusion matrix برای داده‌های تست به این شکل است:



مطابق انتظار شبکه در داده‌های تستی که از کلاس اول (رنگ قرمز که به خوبی تفکیک پذیر بود) هستند، هیچ اشتباهی نداشته است. اما در ۵ مورد زنبق ها از نوع ۲ بوده اند و شبکه آن ها را نوع ۱ تشخیص داده است که در بین ۴۵ داده عدد زیادیست. و ۲ مورد هم برعکس این مشکل را داشته‌اند.