به نام خدا

دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

شبکه های عصبی و یادگیری عمیق

تمرین سری اول

نام و نام خانوادگی	فاطمه سلیقه
شماره دانشجویی	۸۱۰۱۹۸۳۰۶
تاریخ ارسال گزارش	۹۸/۱۲/۱۶

فهرست گزارش سوالات (لطفاً پس از تکمیل گزارش، این فهرست را به‌روز کنید.)

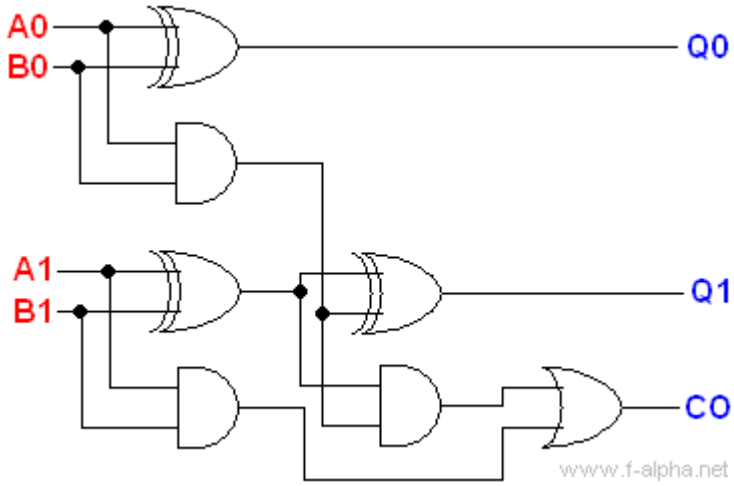
سوال ۱ – طراحی full-adder ۳

سوال ۲ – به روزرسانی دستی شبکه perceptron ۶

سوال ۳ – پیاده سازی perceptron و adaline ۷

سوال ۱ – طراحی full-adder باMcCulloch-Pitts

گیت های یک شبکه full adder دوبیتی به صورت زیر است :



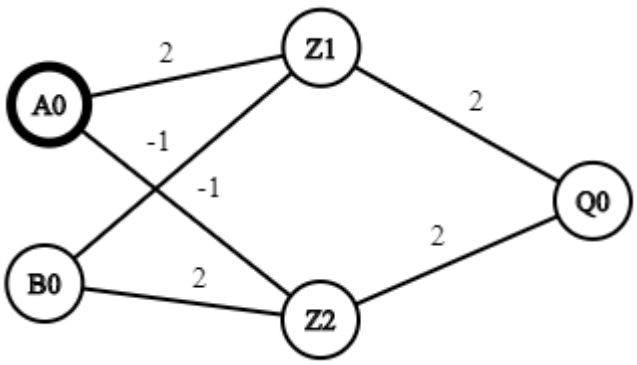
شکل ۱ **full-adder**

همانطور که مشاهده می کنیم بیت اول خروجی (بیت کم ارزش) از xor بیت های کم ارزش ورودی به دست می آید:

Q0 = A0 xor B0

بنابراین با توجه به شبکه عصبی M-P که به صورت زیر است عمل می کنیم :

x_1	x_2	→	y
1	1		0
1	0		1
0	1		1
0	0		0



شکل ۲ شبکه عصبی برای بیت کم ارزش

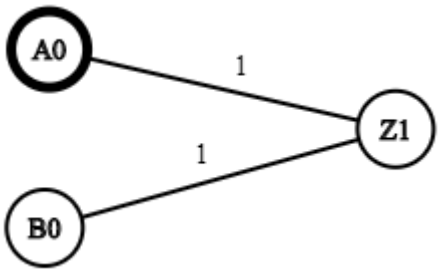
حال برای بیت پر ارزش خروجی (بیت دوم) به صورت زیر است :

Q1 = (A0 and B0) xor (A1 xor B1)

گیت AND به صورت زیر است :

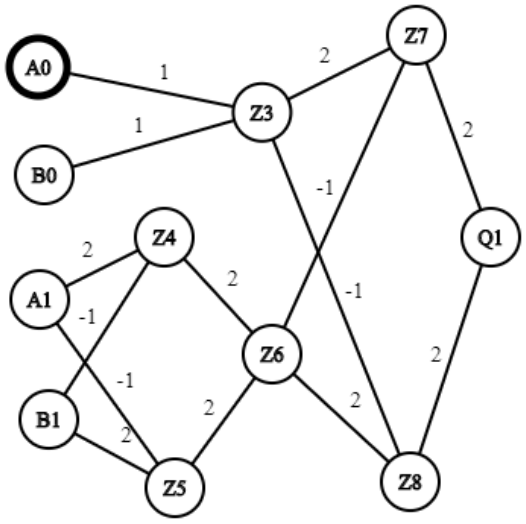
x_1	x_2	→	y
1	1		1
1	0		0
0	1		0
0	0		0

و شبکه عصبی M-P آن به صورت زیر خواهد بود :



شکل ۳ شبکه عصبی گیت **and**

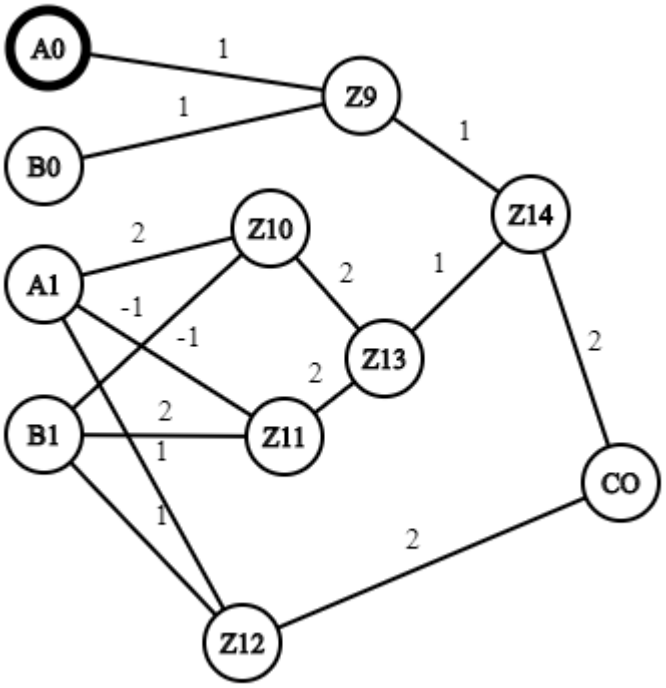
شبکه عصبی برای تولید Q1 :



شکل ۴ شبکه عصبی بیت پر ارزش **full-adder**

برای تولید carry out به صورت زیر است :

CO = ((A0 and B0) and (A1 xor B1)) or (A1 and B1)



شکل ۵ شبکه عصبی **carry out**

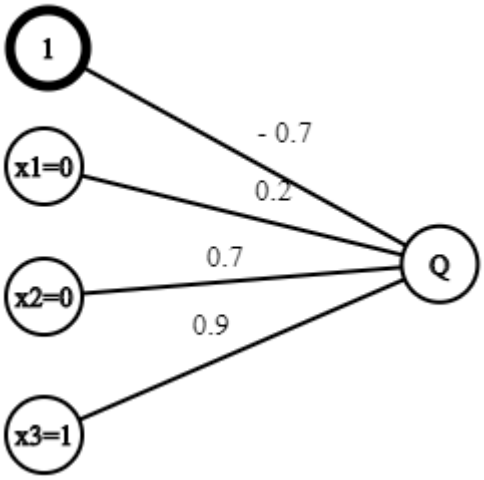
طراحی بالا پیاده سازی شده است و نتیجه به صورت زیر است :

```
00 + 00 = 0 0 0
00 + 01 = 0 0 1
00 + 10 = 0 1 0
00 + 11 = 0 1 1
01 + 00 = 0 0 1
01 + 01 = 0 1 0
01 + 10 = 0 1 1
01 + 11 = 1 0 0
10 + 00 = 0 1 0
10 + 01 = 0 1 1
10 + 10 = 1 0 0
10 + 11 = 1 0 1
11 + 00 = 0 1 1
11 + 01 = 1 0 0
11 + 10 = 1 0 1
11 + 11 = 1 1 0
```

شکل۶ خروجی

سوال ۲ – به روزرسانی دستی شبکهPerceptron

شبکه مورد نظر به صورت زیر است :



شکل ۷ شبکه **perceptron**

t = -1

$$y - in = \sum w_i x_i + b$$

$$y = \begin{cases} +1 & if\ y - in > 0 \\ -1 & if\ y - in < 0 \end{cases}$$

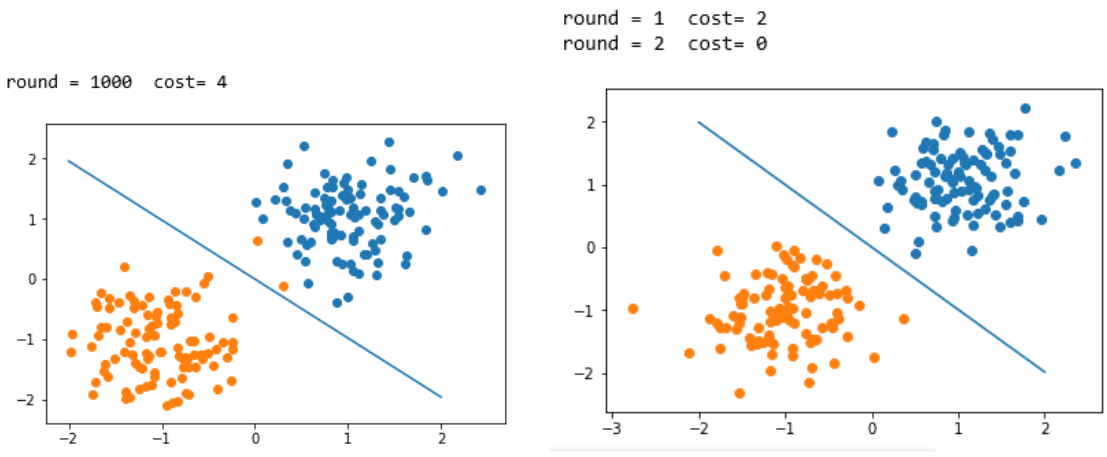
جدول ۲ به روز رسانی وزن ها

W0	W1	W2	W3	y
-0.7	0.2	0.7	0.9	1
-0.9	0.2	0.7	0.7	-1

سوال ۳ – پیاده سازی، بررسی و مقایسه عملکرد دو شبکه Perceptron وAdaline

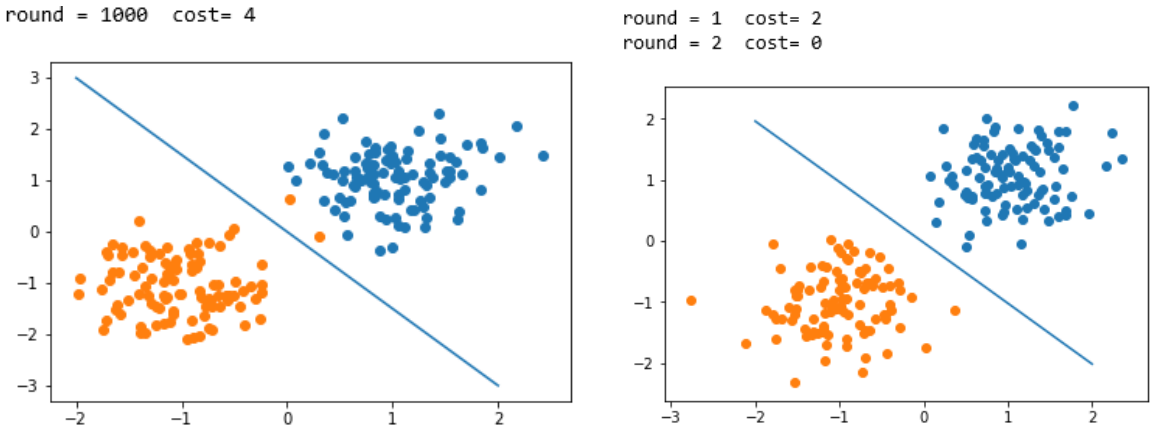
الف)

نتیجه به دست آمده از perceptron از مجموعه داده اول به ازای lr = 0.001:



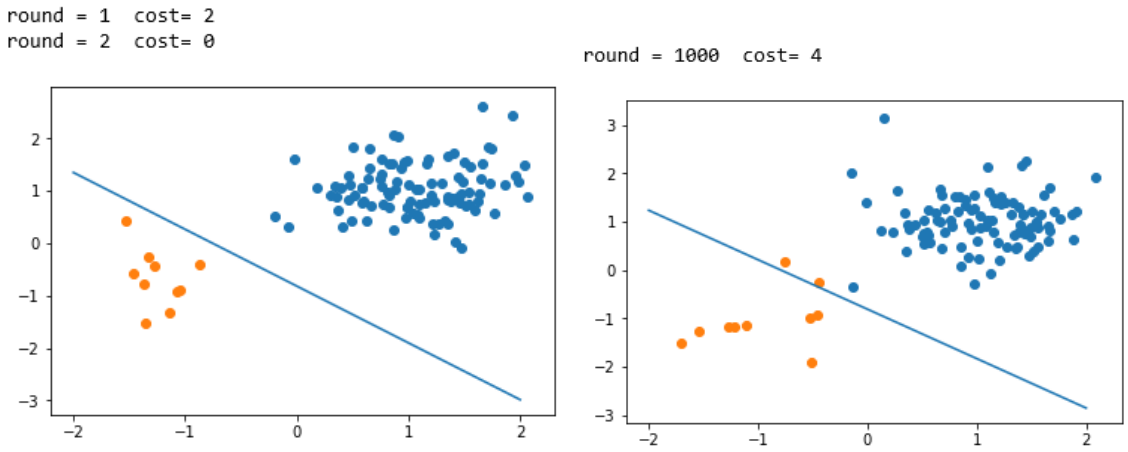
شکل ۸ خرجی **perceptron** دسته اول

نتیجه به دست آمده از adaline از مجموعه داده اول با lr=0.001:



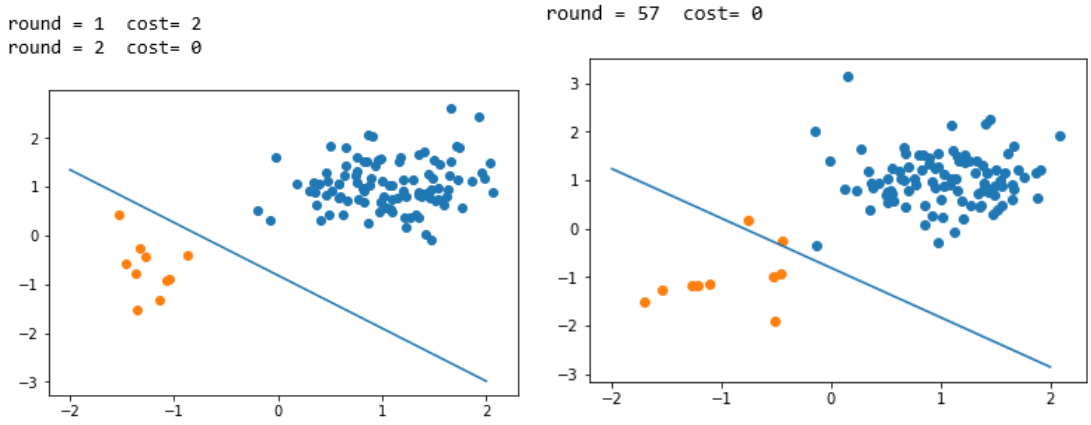
شکل ۹ خرجی **adaline** دسته اول

نتیجه به دست آمده از perceptron از مجموعه داده دوم با lr = 0.001:



شکل ۱۰ خرجی **perceptron** دسته دوم

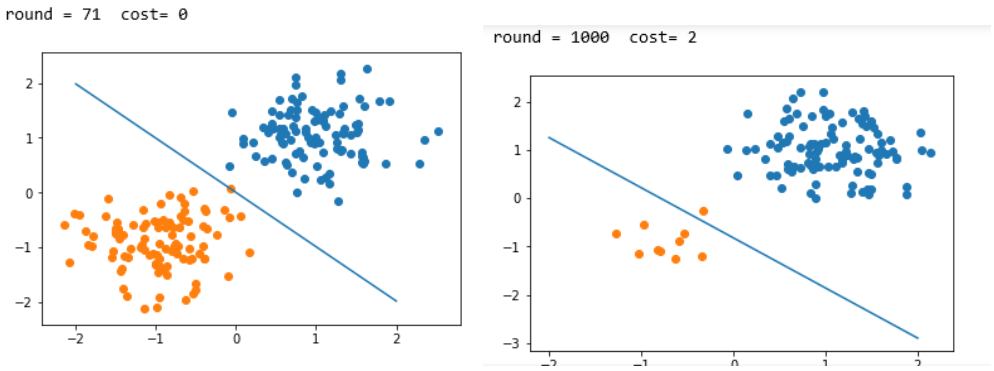
نتیجه به دست آمده از adaline از مجموعه داده دوم با lr = 0.001:



شکل ۱۱ خرجی **adaline** دسته دوم

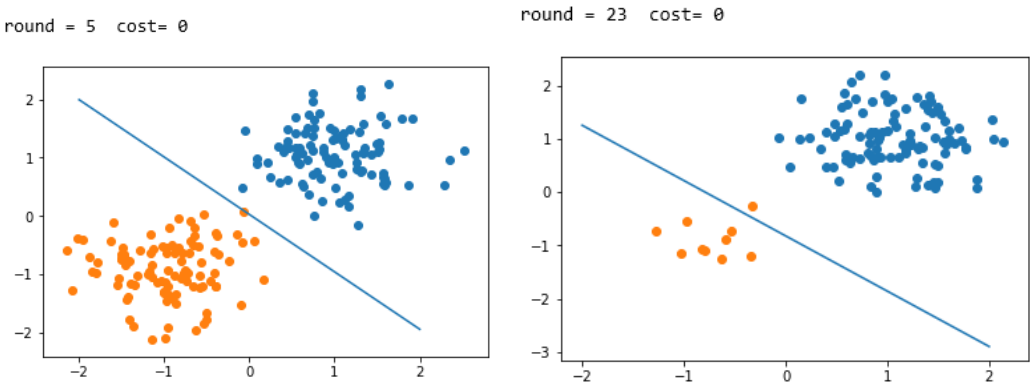
ب)

: Perceptron



شکل ۱۲ خرجی **perceptron**

:Adaline



شکل ۱۳ خرجی **adaline**

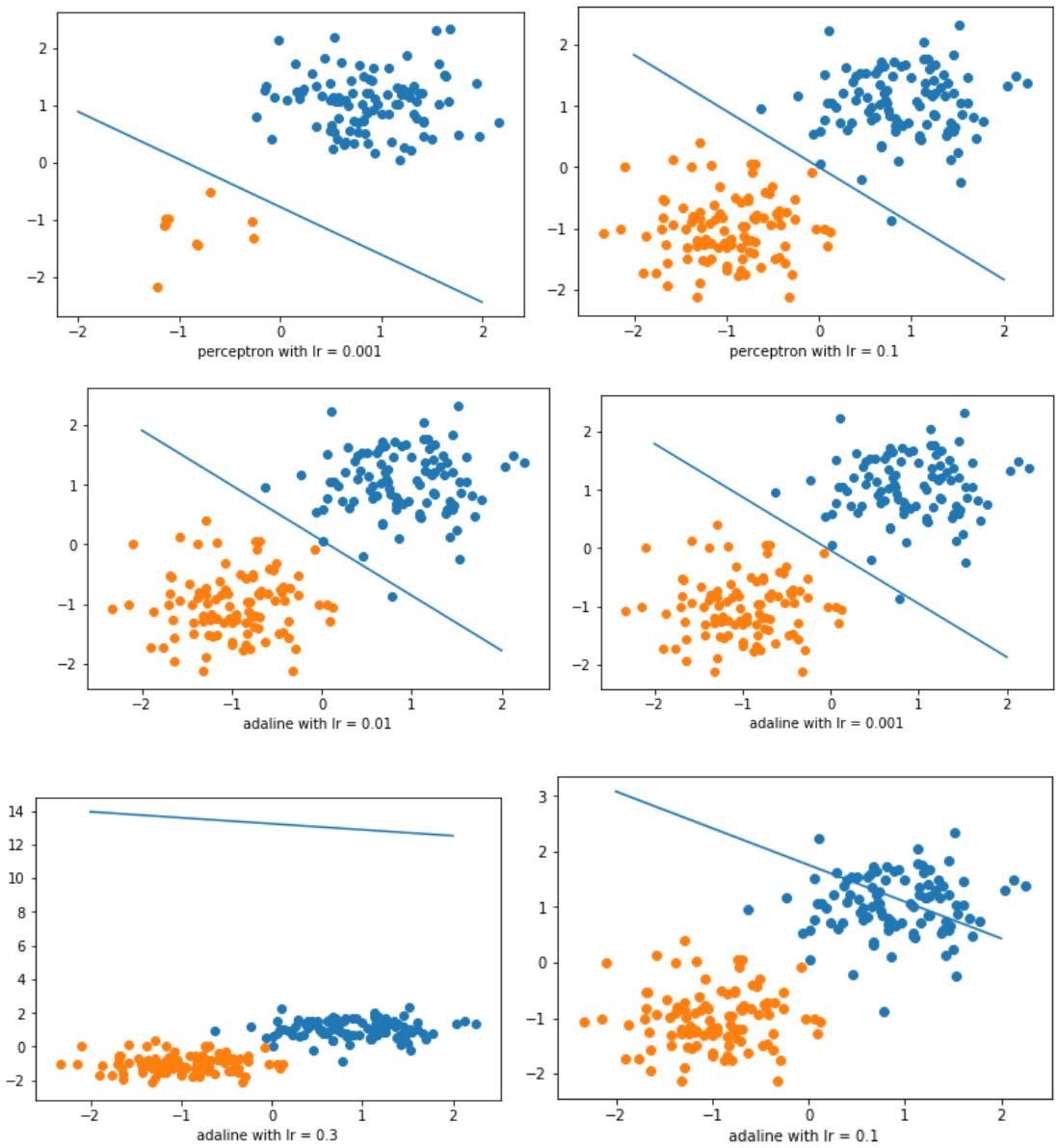
با توجه به نتایج به دست آمده از قسمت الف و عکس های بالا نتایجی که به دست آمده به این صورت است که در داده های balance یعنی در حالتی که تعداد دو نوع داده یکسان است در هر دو الگوریتم سریع تر همگرا می شویم اما در داده های imbalance معمولا زمان همگرایی بیشتر طول می کشد .

از طرفی با توجه با عکس های بالا در این قسمت می بینیم که Adaline سریع تر از perceptron همگرا می شود . مثلا perceptron با ۱۰۰۰ دور ولی adalineبا ۲۳ دور همگرا می شود . و معمولا خط کلاس بندی انها تقریبا یکسان است .

ج)

الگورتم های perceptron , adaline و هم چنین دو دسته داده برای lr های متفاوت بررسی شدند و نتیجه به دست آمده به این صورت است که برای الگوریتم perceptron تغییر lr تاثیری ایجاد نمی کرد .

ولی برای adaline در دسته اول افزایش lr باعث کاهش دقت میشد و کلاس بندی به خوبی انجام نمی شد . و هم چنین زمان اجرای الگوریتم افزایش می یابد .اما adaline در دسته دوم تغییرات زیادی نداشت . بنابراین هر چه lr کمتر بود بهتر عمل می کرد و lr =0.001 برای هر دو الگوریتم خوب عمل می کند.



شکل ۱۴ خرجی برای **lr** های مختلف