به نام خدا

دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

شبکه های عصبی و یادگیری عمیق

تمرین سری اول

نام و نام خانوادگی	محمدعلی شاکردرگاه
شماره دانشجویی	81019487
تاریخ ارسال گزارش	1400/01/13

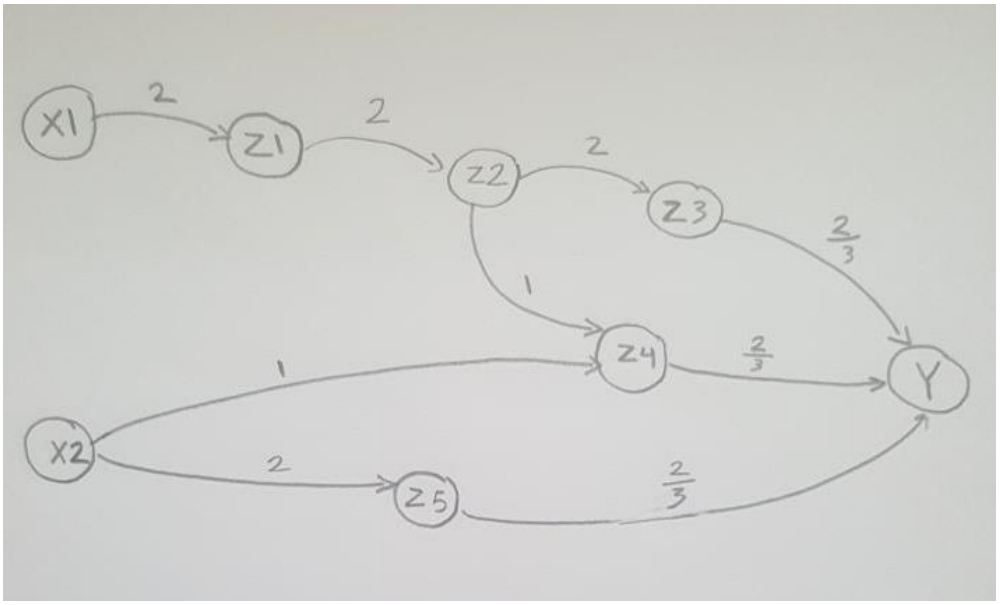
فهرست گزارش

سوال 1 – McCulloch-Pitts

الف:

طبق فرضیات سوال، میدانیم دو بمب با Step 2 زمانی از همدیگر منفجر میشوند. پس از انفجار، نور انفجار دیده میشود و Step1 زمانی بعد، صدای آن شنیده میشود. فرد مورد نظر برای فشار دادن دکمه مورد نظر، باید هم نور انفجار ها را دیده باشد و هم صدایش را شنیده باشد.

در تصویر 1-1-، شبکه مد نظر ترسیم شده است:



شکل 1-1- ترسیم شبکه پرسش اول

در این شبکه، Y خروجی میباشد، x_i ها ورودی میباشند و Z_i ها نورون های لایه های مخفی هستند

ب:

در این قسمت روابط منطقی مربوط به شبکه قسمت قبل، نگاشته میشود:

در روابط منطقی نگاشته شده، Y خروجی میباشد، x_i ها ورودی میباشند و Z_i ها نورون های لایه های مخفی هستند

$$Y(t-4) = Z3(t-3) \;\; AND \;\; Z4(t-3) \;\; AND \;\; Z5(t-3)$$

$$Z3(t-3) = Z2(t-2)$$
$$Z4(t-3) = Z2(t-2) \;\; AND \;\; X2(t-2)$$
$$Z5(t-3) = X(t-2)$$

$$Z2(t-2) = Z1(t-1)$$

$$Z1(t-1) = X1(t-0)$$

ج:

در این بخش به فرایند کار شبکه پرداخته میشود، در این فرایند برای آنکه خروجیY، Fire شود، نیازمند 3 شرط است:

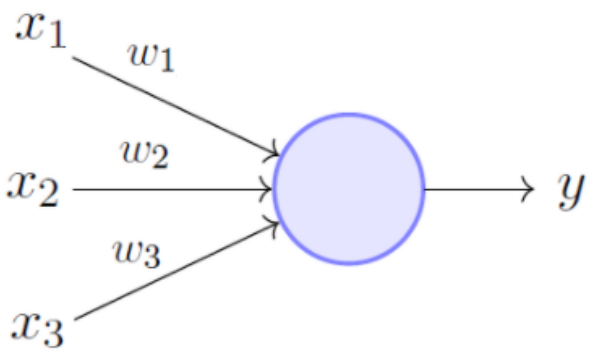
1- شخص صدا و نور بمب اول را شنیده باشد.
2- بمب دوم، دقیقا 2 Step زمانی بعد از Fire شدن بمب اول، Fire شود.
3- شخص صدا و نور بمب اول را شنیده باشد.

این سه اقدام، به ترتیب با نورون های Z3 و Z4 و Z5 مدل شده اند، به ان صورت که:

هنگامی که X1 (بمب اول) فعال میشود، در Step زمانی بعدی Z1 فعال میشود (و همینطور از طریق Z2 و Z3 در Stepهای بعدی خود را به خروجی میرساند که **شرط اول یعنی شنیدن صدا و نور بمب ارضا شده است)**، در استپ زمانی بعدی Z2 و X2 (بمب دوم) فعال میشود و فقط در صورتی Z4 فعال میشود که X2 فعال شده باشد و 2 Step زمانی از فعال شدن X1 گذشته باشد یعنی Z2 فعال شده باشد، (که این **شرط دوم را ارضا میکند چرا که فقط و فقط با 2 Step زمانی از فعال شدن X1 ، و Fire شدن X2 ، Z4 فعال میشود)** در Step زمانی بعد با شرایط گفته شده(Fire شدن به ترتیب X1 و X2 با دو Step زمانی)، Z3 و Z4 و Z5 فعال(**شرط سوم ارضا میشود چرا که در یک Step زمانی بعد از Fireشدن X2 ، صدا شنیده شد و Z5 فعال شد)** میشوند و با همدیگر AND مشوند و خروجی ما ساخته میشود

سوال ۲ – Perceptron

شبکه پرسپترون یک شبکه عصبی تک لایه بوده که عملکرد خطی ای از خود نشان میدهد، نمای یک مثال این شبکه در شکل 2-1 در زیر آمده است:

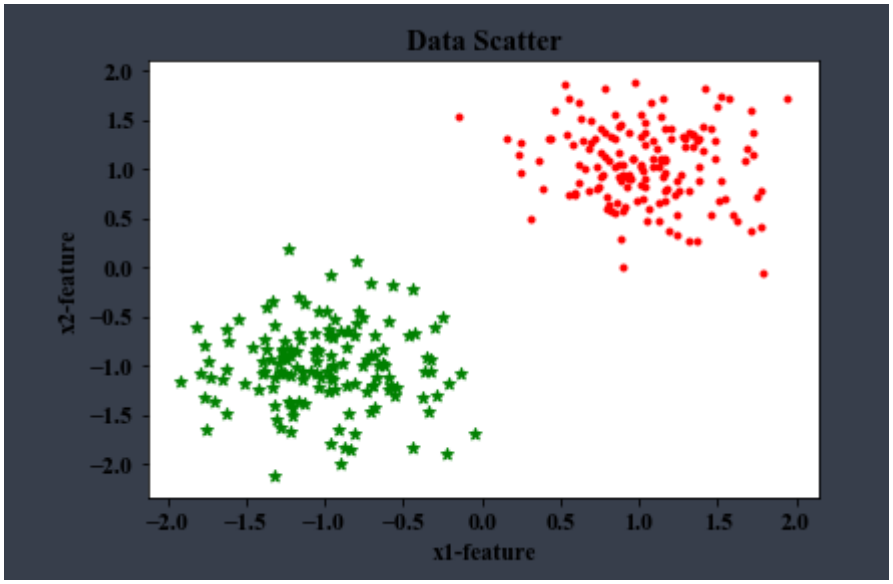


شکل 2-1- نمای یک شبکه پرسپترون

در این پرسش، ابتدا داده ها را بایست به دو گروه Training Set و Test Set تقسیم شوند. در پیاده سازی ما، 75% اول داده ها (300تای اول) به گروه Training Set پیوستند و مابقی (100تای آخر) به Test Set پیوستند. **تمام کد های این سوال در NNDL_HW1_Q2.ipynb موجود میباشد.**

الف:

داده ها را همانطور که در پرسش خواسته شده است با 2 رنگ متفاوت و همچنین شکل های مختلف Scatter plot کرده و در شکل 2-2 در زیر نمایش داده میشود:



شکل 2-2- نمایی از plot داده های دو کلاس

ب:

در این قسمت به Train کردن Train Set به روش پرسپترون پرداخته میشود، در ابتدا با استفاده از فرمول که در زیر آمده است، میبایست net را بدست آورد:

فرمول 2-1:

$$net = \sum_{i=1}^{n} w_i x_i + b$$

که در آن w_i وزن، b بایاس و x_i فضای ورودی که میتواند ویژگی یا ... را تشکیل دهد میباشند.

سپس میتوان تابع hypothesis را با کمک net تشکیل داد:

فرمول 2-2:

$$h = \begin{cases} 1 & \text{if} \quad \text{net} > \theta & \text{Active} \\ 0 & \text{if} \quad |\text{net}| < \theta & \text{non - descision} \\ -1 & \text{if} \quad \text{net} < -\theta & \text{Passive} \end{cases}$$

$$\theta : \text{non} - \text{negative threshold}$$

مشاهده میشود که h حاصل از تابع روی ضابطه تعریف شده روی net و θ است و مقدار میگیرد.

برای تجدید وزن ها و بایاس نیز از فرمول زیر میتوان استفاده نمود:

فرمول 2-3:

$$w_i(new) = w_i(old) + \alpha x_i t$$

$$b_i(new) = b_i(old) + \alpha t$$

که در آن t هدف و α نرخ یادگیری میباشند. نحوه بروزرسانی وزن ها و بایاس بگونه ای رخ میدهد که از روش گرادیان نزولی استفاده کند و رو به همگرا شدن به جواب بهینه شود.

الگوریتم پرسپترون پیاده سازی شده:

ابتدا وزن ها و بایاس و آستانه و نرخ بروزرسانی را مقدار دهی میکنیم در کد، مقدار دهی همانند مقابل انجام گرفت:

$$\alpha = 1$$

$$\theta = 0.1$$

$$b = 0$$

$$w1, w2 = 0,0$$

سپس برای هر جفت x_i ها محاسبه net و h که با استفاده از فرمول های 2-2 , 2-1 محاسبه میشود، حال شرط بدین گونه چک میشود که خطای h-t صفر میشود یا خیر، اگر صفر میشود که به سراغ x_i بعدی باید رفت اگر نه میبایست از طریق فرمول 2-3 آن را آپدیت نمود و این فرایند را برای تمام داده های Train Set انجام داد حال اگر برای تمام جفت تمرین های (s,t) خطا برابر با صفر بود، الگوریتم به پایان میرسد و وزن ها و بایاس مناسب هستند وگرنه به ایپاک بعدی رفت و این فرایند را از بعد از مقدار دهی اولیه تکرار کرد، تا به شرط خاتمه رسد.

نتیجه حاصل از الگوریتم پرسپترون پیاده سازی شده:

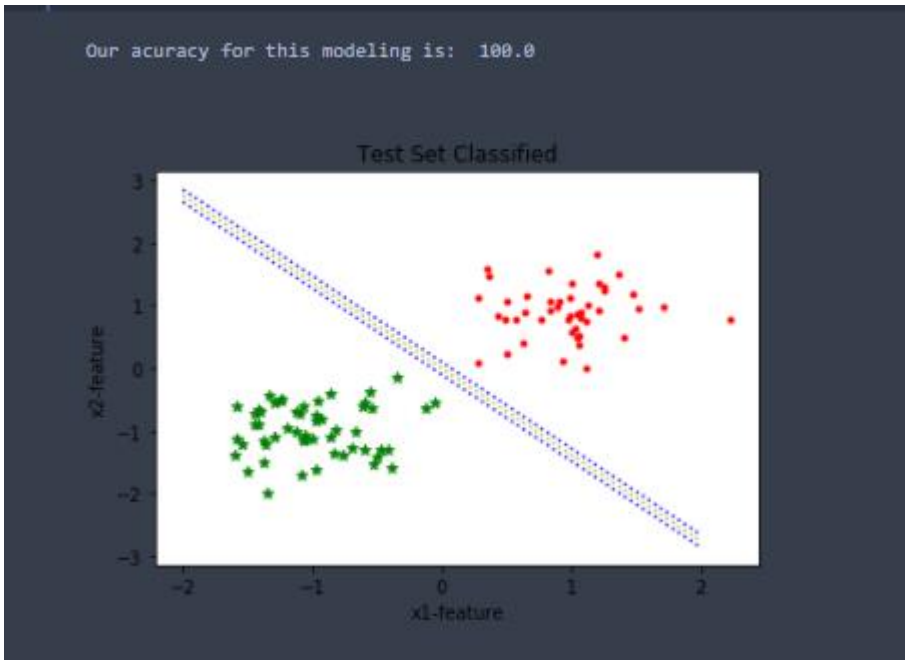
بعد از انجام الگوریتم بالا روی داده های تمرین، به نتیجه زیر برای وزن ها و بایاس میرسیم:

```
w1:  -1.600163552662912
w2:  -1.1626987228737826
b:  0.0
Epochs needed: :  2
```

شکل 2-3- نتایج الگوریتم پرسپترون

ج:

بعد از ارزیابی داده های Test و مقایسه آن با Classifier ای که در قسمت "ب" به آن رسیدیم، دقت 100% را خواهیم داشت همانند شکل زیر:



شکل 2-4- نتیجه کلاس بندی داده های تست

د:

برای ارائه یک پاسخ قانع کننده میبایست به فلسفه Threshold در الگوریتم پرسپترون پرداخت، در واقع میبایست معادلات زیر را پی گرفت:

$$\sum_{i=1}^{n} w_i x_i = Threshold$$

$$\sum_{i=1}^{n} w_i x_i - Threshold = 0$$

$$\sum_{i=1}^{n} w_i x_i + b = 0$$

$$\xrightarrow{Obviously} b = -Threshold$$

پر واضح است که در واقع Threshold خود را در قالب بایاس به نمایش میگذارد. حال تاثیر مقادیر متفاوت آن چه میتواند باشد؟ برای شرایط متفاوت میتواند دقت ما دچار تغییر شود، اگر اندازه Threshold بزرگ باشد این به معنای تغییر بزرگی در بایاس در وهله اول میباشد که در صورت

عدم امکان tune کردن مناسب آن در الگوریتم ، قطعا به مشکل خواهیم خورد و با احتمال بالایی از دقت ما کم خواهد شد.

همچنین قابل ذکر است که اگر هم بتوانیم به خوبی tune کنیم، به epoch های بسیار بیشتری نیاز خواهیم داشت، برای نمونه میتوان مثال های زیر را نشان داد:

برای theta=10 :

```
w1:  -16.57379970670919
w2:  -13.21601720667468
b:  1.0
Epochs needed: :  7
```

```
Our acuracy for this modeling is:  97.0
```

شکل 2-5- نتایج با threshold 10

که دیده میشود به epoch 7 نیاز دارد و دقت آن 97% است، درحالی که با انتخاب اولیه مان برای theta با epoch 2 به دقت 100% رسیده شد.

برای theta=5000 :

```
w1:  -7403.099338088021
w2:  -6367.065528748657
b:  372.0
Epochs needed: :  2492
```

```
Our acuracy for this modeling is:  96.0
```

شکل 2-6- نتایج با threshold 5000

که دیده میشود به epoch 2492 نیاز دارد و دقت آن 96% است، درحالی که با انتخاب اولیه مان برای theta با epoch 2 به دقت 100% رسیده شد.

سوال 3 – Adaline

الف:

تفاوت های اساسی:

1- قضیه همگرایی قاعده یادگیری پرسپترون متضمن میشود که اگر w* یک بردار وزن باشد که به ازای هر p روی ورودی ها (t(p) = f(x(p)w* باشد با هر بردار وزنی ای که آغاز کنیم، یادگیری پرسپترون مارا به یک وزن صحیح که پاسخ درست را برای تمام الگوهای یادگیری میدهد.
اما در همگرایی دلتا (Adaline) هیچ تضمینی برای همگرا شدن به جواب بهینه وجود ندارد(هرچند به روشی که در بخش "ج" اشاره میشود میتوان این مهم را براورده کرد)

2- مدل پرسپترون برای تعیین Coefficient های مسئله خود از Class labels استفاده میکند اگر خطایی در این label گذاری نسبت به target اتفاق افتد، بروزرسانی میکند اما مدل Adaline از continuous predicted values برای بروزرسانی Coefficient های مسئله خود استفاده میکند(استفاده از Gradient Descent)

شباهت های اساسی:

1- هر دو classifiers برای binary classification میباشند.
2- هردو مرز تصمیم گیری خطی(linear decision boundary) دارند

ب:

نکته مهم: طبق اصلاحیه، انحراف معیار را برابر 0.5 در نظر میگیریم. **تمام کد های این سوال در NNDL_HW1_Q3.ipynb موجود میباشد.**

در این قسمت از یک نورون Adaline استفاده میشود، طبق خواسته سوال، شرح داده خواهد شد که چگونه این اقدام صورت میگیرد:

در الگوریتم دلتا، با فرض نورون Adaline، میتوان فرمول 3-1 زیر را برای net تشکیل داد:

فرمول3-1:

$$net = \sum_{i=1}^{n} w_i x_i + b$$

که در آن w_i وزن، b بایاس و x_i فضای ورودی که میتواند ویژگی یا ... را تشکیل دهد میباشند.

سپس میتوان تابع hypothesis را با کمک net تشکیل داد:

فرمول 3-2:

$$h = f(net) = \begin{cases} +1 & \text{net } \geq 0 \\ -1 & \text{net } < 0 \end{cases}$$

مشاهده میشود که h حاصل از تابعی همانند ()Sign است که روی ضابطه تعریف شده روی net تعریف شده است مقدار میگیرد. نکته حائز اهمیت، عدم وجود θ در ضابطه تابع است.

برای تجدید وزن ها و بایاس نیز از فرمول زیر میتوان استفاده نمود:

فرمول 3-3:

$$w_i(new) = w_i(old) + \alpha(t_i - net)x_i$$

$$b_i(new) = b_i(old) + \alpha(t_i - net)$$

نکته حائز اهمیت نیز در بروزرسانی وزن ها و بایاس این است که چون ماهیت h برابر با تابع sign است مناسب الگوریتم نیست و اگر قصد استفاده از آن باشد میبایست به جای تابع sign از تابع tanh استفاده کرد که در قسمت "ج" توضیح داده خواهد شد.

الگوریتم Adaline پیاده سازی شده:

ابتدا وزن ها و بایاس و آستانه و نرخ بروزرسانی را با مقادیری کوچک مقدار دهی میکنیم در کد، مقدار دهی همانند مقابل انجام گرفت:

$$\alpha = 0.01$$

$$\theta = 0.4$$

$$b = 0.2$$

$$w1, w2 = 0.1$$

سپس برای هر جفت x_i ها محاسبه net و که با استفاده از فرمول 3-1 محاسبه میشود، حال الگوریتم بدین گونه جلو میرود که وزن ها و بایاس را بدون چک کردن شرطی به روز رسانی میکند، سپس یک تابع هزینه به صورت مقابل تعریف میشود:

فرمول 3-4:

$$J = 0.5 * (t_i - net)^2$$

که این این مقدار ثبت میگردد اگر تابع هزینه محاسبه شده برای همه جفت های (s,t) کتر از θ بود، الگوریتم به پایان میرسد وگرنه باید دوباره از ابتدا الگوریتم از بعد از مقدار دهی اولیه شروع به پردازش کند.

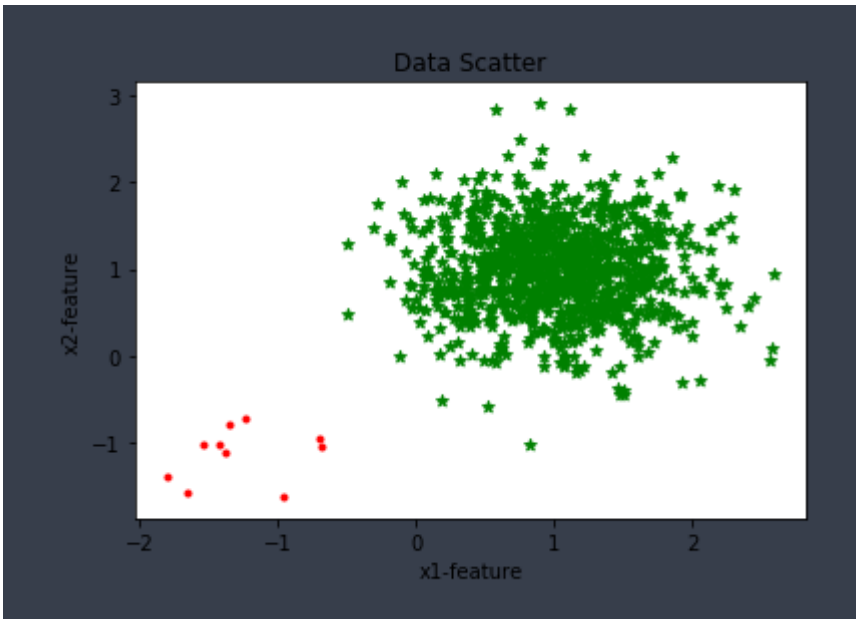
نتیجه حاصل از الگوریتم Adaline پیاده سازی شده:

بعد از انجام الگوریتم بالا روی داده های تمرین، به نتیجه زیر برای وزن ها و بایاس میرسیم:

- **شکل اول:**

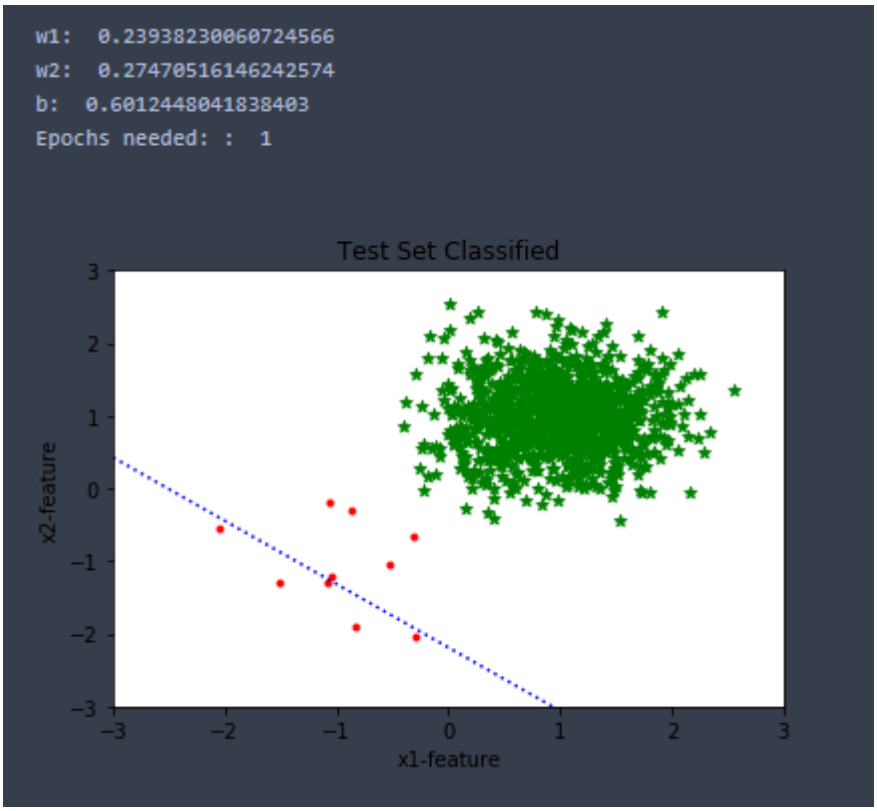
همانطور که در درسنامه هم اشاره شد، روش دلتا، تضمینی برای همگرایی نمیدهد، دلیل آن این است که این net است که دارد به t نزدیک میشود و نه h. در این داده ها توزیع از نظر Population آن ها واضح است که الگوریتم لزوما Converge نمیکند اما برای آن که صحت این امر نشان داده شود، به بررسی گذاشته میشود:

داده ها همانند شکل زیر میباشند:



شکل 3-1 نمایش داده های شکل اول

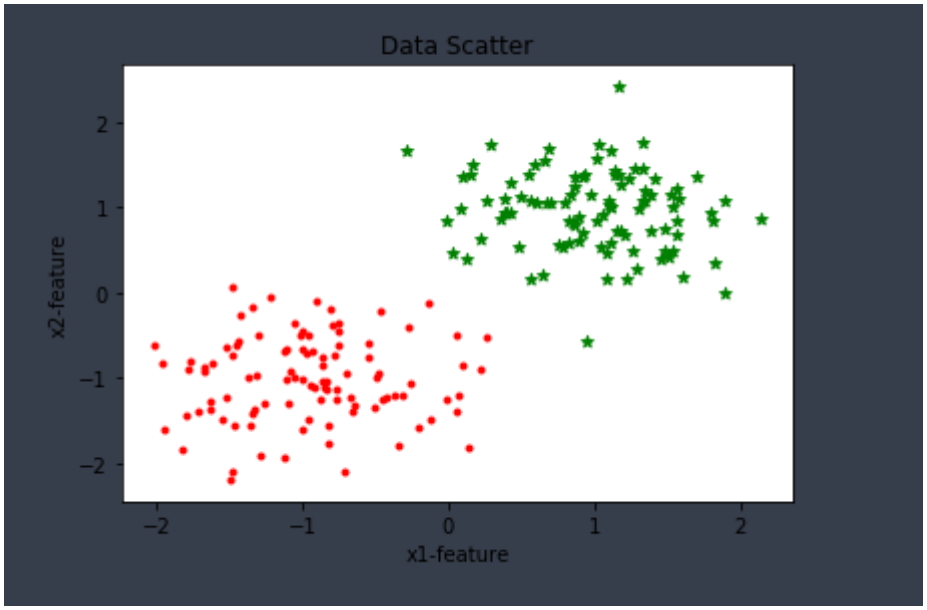
حال طبق الگوریتم اشاره شده پیاده سازی انجام میشود اما بعد از گذشت 1500000 Epoch به مرحله ای که دقیقا بتواند به Convergence برسد، نمیتواند برسد در حالی که از 1010 داده، خواستاردقت 0.992 شده بودیم، لذا دقت را به 1000/1010 گذاشته و حاصل به مانند زیر میشود:



شکل 3-2- نتایج الگوریتم Adaline روی شکل اول

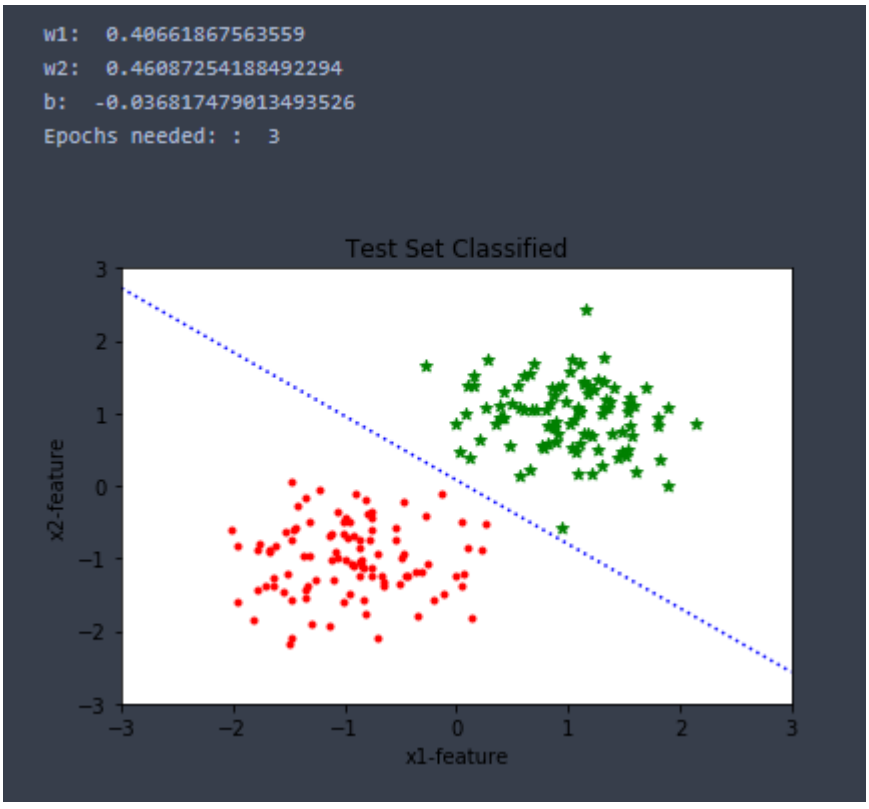
- **شکل دوم:**

در این قسمت population داده ها و Distribution آن ها یکسان است پس به Classify شدن میتوان امیدوار بود. داده ها همانند شکل زیر میباشند:



شکل 3-3 نمایش داده های شکل دوم

حال طبق الگوریتم اشاره شده پیاده سازی انجام میشود و پس از انجام 3 epoch به نتیجه زیر میرسد:



شکل3-4- نتایج الگوریتم Adaline روی شکل دوم

ج:

همانطور که در قسمت های "الف" و "ب" هم بحث شد، ویژگی همگرایی روش دلتا به این صورت است که متضمن همگرایی نمیشود و این امر را برای داده شکل اول قسمت "ب" دیده شد، اما دلیل آن اکنون مفصل تر شرح داده خواهد شد:

همانگونه که در الگوریتم دیده شد، در حل از تفاضل net و t استفاده شد و نه t-h بدان معنا که این net بود که به هدف نزدیک میشد و نه h که علت ناچاری ما از استفاده از آن ماهیت Sign() بودن h بود. برای رفع این مشکل، **راه حلی وجود دارد و آن استفاده از تابع Soft-Sign مثل tanh() به جای sign برای h است**. که تابع هزینه را به شکل زیر تغییر میدهد:

فرمول 3-5:

$$J_p(w,b) = 0.5 * \left(t_p - \tanh\left(\gamma * net(x_p, w, b)\right)\right)^2$$

فقط اشاره به این نکته نیز میتواند مفید باشد که در انتخاب γ اگر خیلی بزرگ باشد، tanh شبیه به sign عمل میکند و اگر خیلی کوچک باشد، رفتار خطی را مشاهده میتوان نمود $(\gamma * net = net)$

د:

برای روشن شدن موضوع میبایست به سراغ تعاریف عملکرد اصلی الگوریتم اشاره کرد، روش دلتا بر اساس Gradient descent عمل میکند، و تابع هزینه کلی ای مانند رو به رو را اگر مفروض شویم:

فرمول 3-6:

$$J_p(w,b) = 0.5 * \left(t_p - net(x_p, w, b)\right)^2$$

بر اساس فرمول های به روز رسانی 3-3، میتوان بر اساس قاعده گرادیان، به فرمول های زیر رسید:

فرمول های 3-7:

$$\delta w_i = W_i^+ - W_i^- = -\alpha \frac{d J_p(w,b)}{dw_i} = \alpha(t - net)x_i$$

$$\delta b = b^+ - b^- = -\alpha \frac{d J_p(w,b)}{db} = \alpha(t - net)$$

که به وضوح نمایانگر تاثر آن در بروز رسانی وزن ها و بایاس میباشد (به همین علت نام آن "نرخ یادگیری" یا "Learning rate" میباشد) و با تنظیم مناسب آن میتوانیم Variation های وزن ها و بایاس را در هر حلقه کنترل کنیم.

نکته دیگری نیز وجود دارد که میتواند مورد توجه قرار گیرد، اینکه نرخ یادگیری باید کمتر از بزرگترین مقدار ویژه ماتریس کورلیشن باشد :

فرمول 3-8:

$$\alpha < \frac{1}{2P}\sum_{p=1}^{P} x(p)^T x(p)$$

که میتوان به صورت غیر مستقیم نتیجه گیری کرد که مطلوب خواهد بود اگر $0.1 < \alpha * n < 1$ وقتی n برابر با تعداد ورودی هاست.

سوال 4 – Madaline

الف:

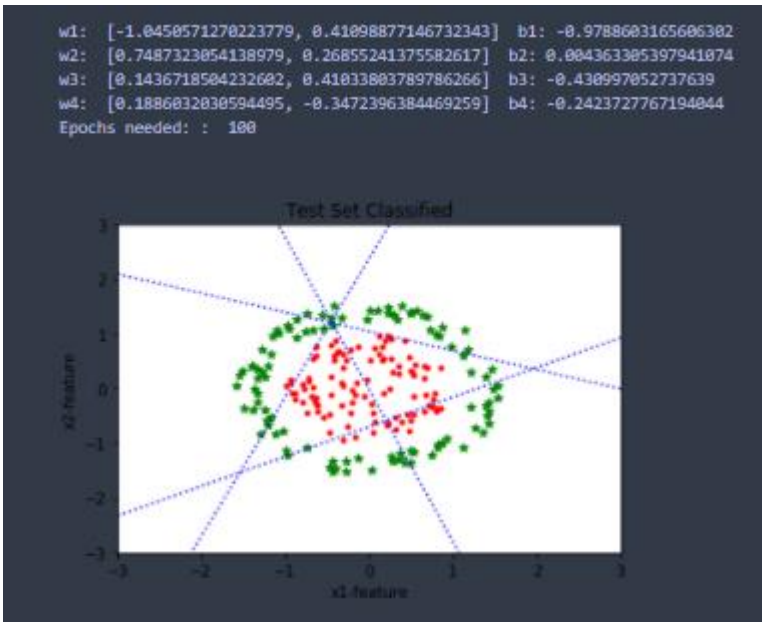
در این پرسش با استفاده از سیستم MAdaline به بررسی کلاس بندی داده ها پرداخته شد، در این مثال بخصوص خواسته شده است با به ترتیب با 4، 6 و 8 نورون شبکه را طراحی و خروجی نمایش داده شود.

ابتدا به بررسی پیاده سازی MAdaline پرداخته میشود، همانطور که در مرجع این درس هم نمایان است، MAdaline با دو روش MR-I و MR-II قابل پیاده سازی است، تفاوت این دو الگوریتم، در تعداد پارامتر هایی است که به روز رسانی میشوند. در روش MR-I لایه نهان را Train میکند اما الگوریتم MR-II تمام وزن ها و بایاس ها به روز رسانی میشوند.

در این پیاده سازی از روش MR-I استفاده میشود.این روش میتواند 2 نوع شرط خاتمه داشته باشد، شرط خاتمه استفاده شده اینجا، تعداد iteration هایی(ایپاک هایی) است که الگوریتم در جهت بهینه سازی بر میدارد. **تمام کد های این سوال در NNDL_HW1_Q4.ipynb موجود میباشد.**

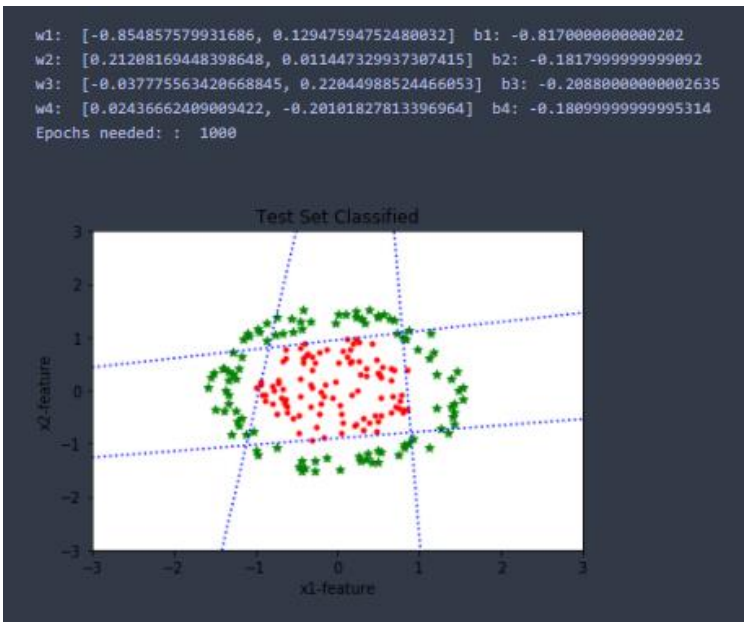
ب:

1- برای شبکه با 4 نورون با 100 ایپاک:



شکل 4-1- نتایج 4 نورون با 100 ایپاک

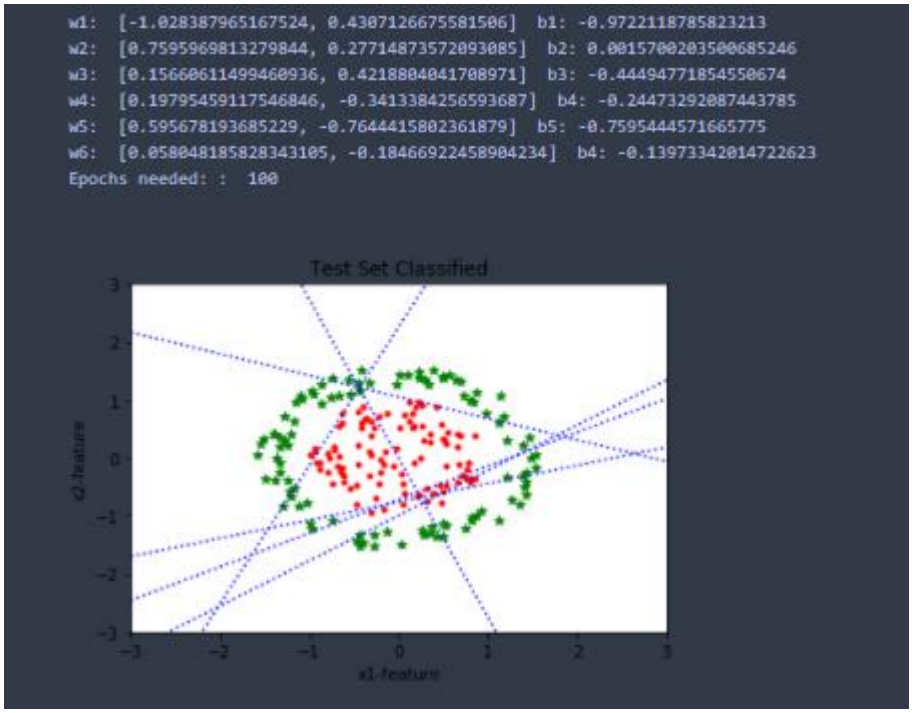
2- برای شبکه با 4 نورون با 1000 ایپاک:



شکل 4-2- نتایج 4 نورون با 1000 ایپاک

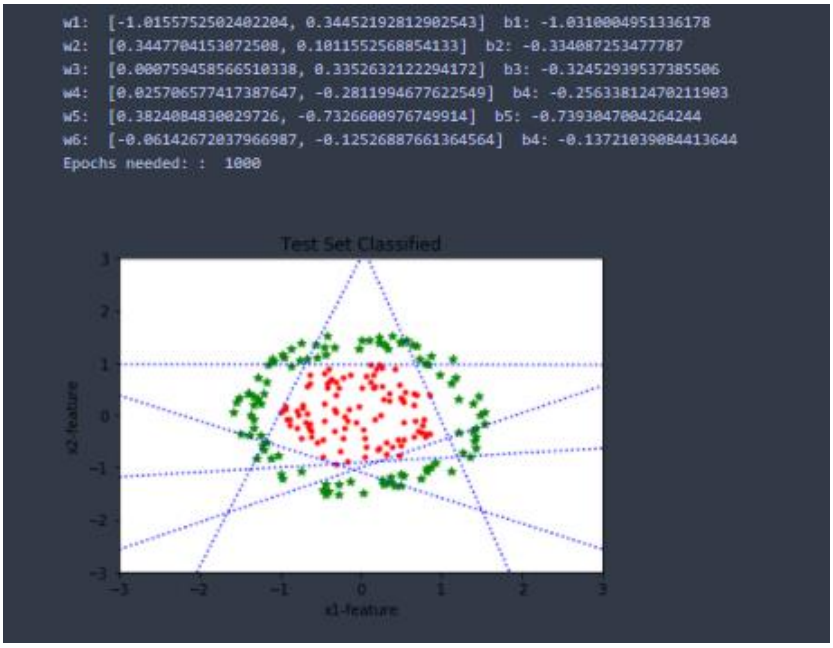
برای شبکه با 6 نورون:

1- برای شبکه با 6 نورون با 100 ایپاک:



شکل 4-3- نتایج با 6 نورون با 100 ایپاک

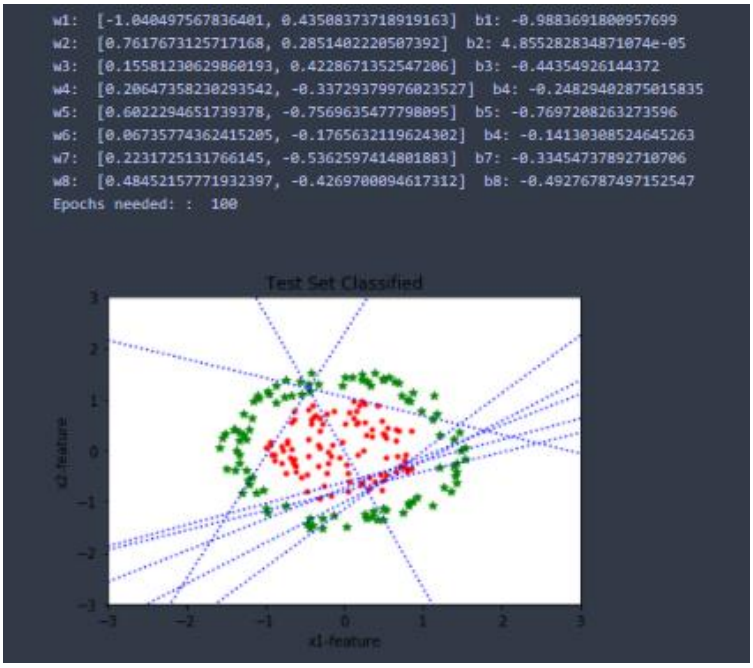
2- برای شبکه با 6 نورون با 1000 ایپاک:



شکل 4-4- نتایج با 6 نورون با 1000 ایپاک

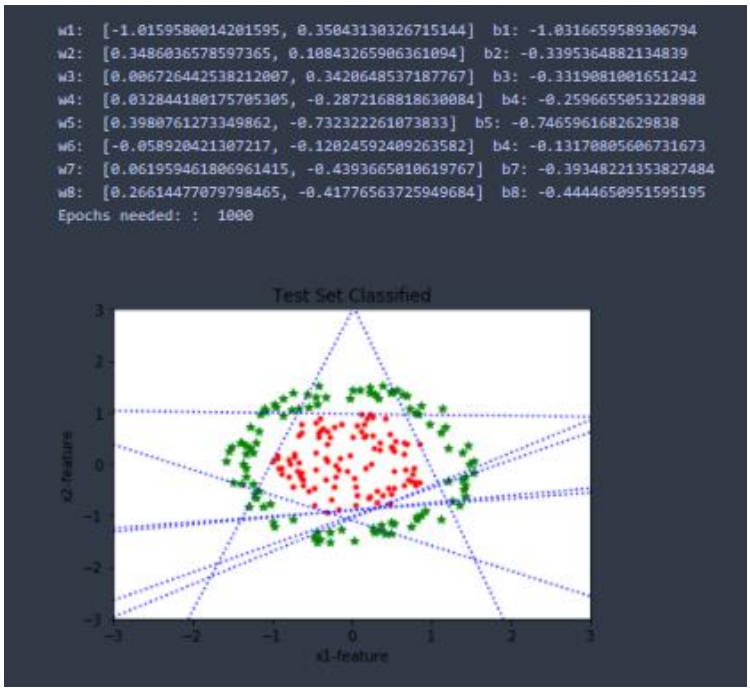
برای شبکه با 8 نورون:

1- برای شبکه با 8 نورون با 100 ایپاک:



شکل 4-5- نتایج با 8 نورون با 100 ایپاک

2- برای شبکه با 6 نورون با 1000 ایپاک:

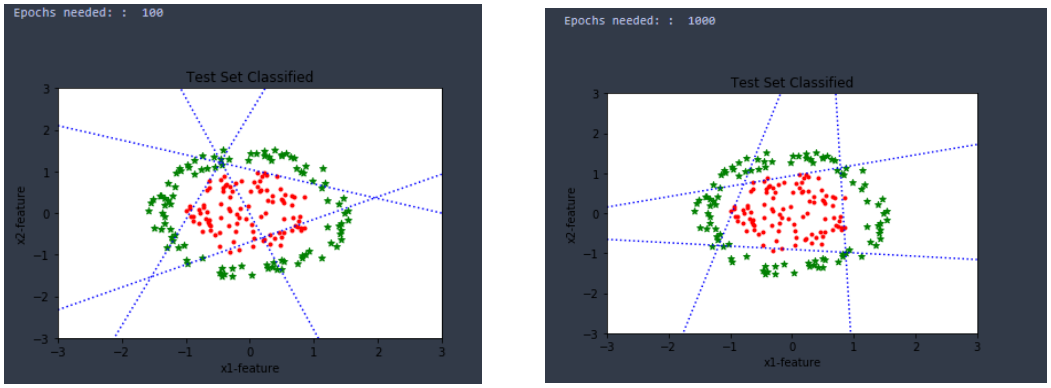


شکل 4-6 نتایج با 8نورون با 1000 ایپاک

ج:

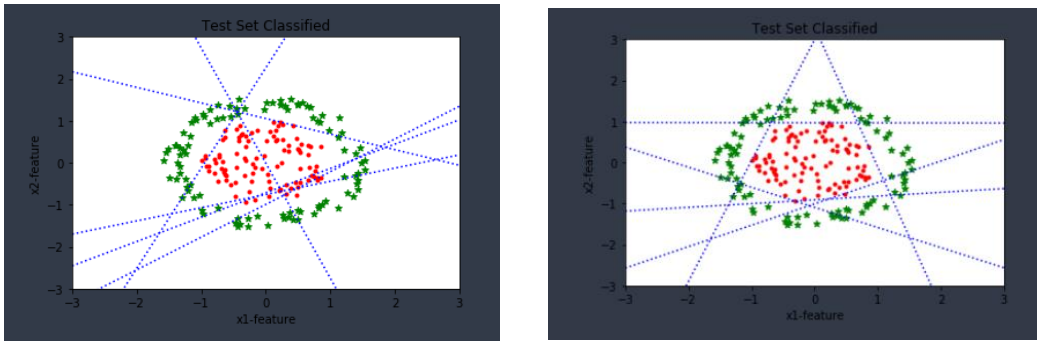
در این بخش به بررسی دقت (accuracy) و رابطه آن با تعداد ایپاک (epoch) پرداخته میشود، همانگونه که در بخش قبل قابل ملاحظه است، ابتدا با 100 ایپاک و سپس با 1000 ایپاک برای تمامی حالت های 4 ، 6 و 8 نورون الگوریتم اجرا شده است، و طبق پیشبینی با افزایش تعداد ایپاک ها، دقت نیز افزایش داشته است اما زمانبر تر نیز بوده است.

برای 4 نورون(سمت راست 1000 و سمت چپ 100 ایپاک):



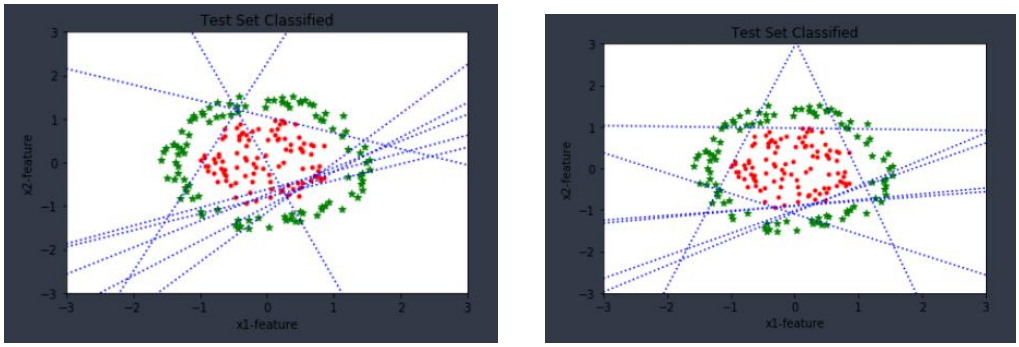
شکل 4-7 نتایج با 4 نورون

برای6 نورون(سمت راست 1000 و سمت چپ 100 ایپاک):



شکل 4-8 نتایج با 6 نورون

برای 8 نورون(سمت راست 1000 و سمت چپ 100 ایپاک):



شکل 4-9 نتایج با 8 نورون

در واقع همیشه یک Trade off بین دقت و تعداد ایپاک ها خواهیم داشت، هرچه دقت بشتری بخواهیم باید هزینه زمان بیشتری(تعداد ایپاک بیشتری) هم پرداخت کنیم. واظح است که میتوان شرط توقف ایپاک بسته به نوع مسئله متفاوت خواهد بود ولی در نهایت این مصالحه بین تعداد ایپاک ها و دقت لازم همواره میباسیت برقرار شود