تمرین 1 : مفاهیم زیر را بطور دقیق تعریف کنید

الف) آموزش آنلاین ب) محاسبات Real time ج)آموزش آفلاین

پاسخ :

الف :

آموزش آنلاین یا افزایشی به آموزشی گفته می شود که ما سعی می کنیم پایگاه دانش ماشین مورد نظرمان را به طور افزایشی طی زمان با وارد کردن نمونه های مختلف آموزش دهیم .

منبع :

A review of online learning in supervised neural networks

DOI 10.1007/s00521-013-1534-4

ب:

در علم کامپیوتر به مجموعه سیستمی (نرم افزاری و سخت افزاری) گفته می شود که محدودیت شدید زمانی داشته باشند ، یعنی بین یک اتفاق تا پاسخ سیستم به آن مدت زمان کمی باشد یعنی بین وارد شدن ورودی و پردازش آن و پاسخ به آن زمانی کم باشد ، به بیان دیگر سیستم تضمین دهد که در بازه زمانی خاصی پاسخ دهد وآن پاسخ نیز پاسخی صحیح و دقیق باشد ، معمولا این بازه زمانی بین میلی ثانیه و گاه حتی میکرو ثانیه می باشد .

منبع:

https://en.wikipedia.org/wiki/Real-time_computing

تاریخ مراجعه به سایت :

26 اسفند 1395

ج:

آموزش آفلاین به آموزشی گفته می شود که شامل دو فاز تمرین و تست است ماشین سعی میکند طی مرحله تمرین تابع هزینه را کم کند و به بهترین جواب ممکن برسد و معمولا در مرحله اجرایی به طور رسمی آموزشی نداریم.

منبع :

A review of online learning in supervised neural networks

DOI 10.1007/s00521-013-1534-4

تمرین 2: در طراحی ساختار شبکه های عصبی ممکن است پدیده سرریز پارامتر رخ دهد تحقیق کنید که این اتفاق چیست چه زمانی رخ می دهد و آیا ممکن است کاربردی داشته باشد ؟

پاسخ :

هنگام طراحی شبکه های عصبی ما می توانیم تعداد نورون ها و پارامتر ها را تعیین کنیم و هر چه تعدادشان بیشتر باشد شبکه پیچیده تر است به این کار سرریز پارامتر می گویند اما در این شبکه ها لزوما افزایش پیچیدگی منجر به بهتر شدن پیش بینی ها و نتیجه ها نمی انجامد زیرا اگر داده ای در خارج داده های آموزش داده شده به آن بدهیم جوابی ناصحیح و غیر قابل اطمینان به ما می دهد علاوه بر آنکه زمان تمرین آن زیاد خواهد بود پس همیشه نباید این کار را انجام داد یعنی بایستی مقداری کافی پارامتر داشت هر چند روشی دقیق و قطعی برای دانستن تعداد کافی موجود نیست.

منبع:

Over-parameterisation, a maor obstacle to the use of artificial neural networks in hydrology?

Hydrology & Earth System Sciences,7(5),693-706 (2003)

تمرین 3: یکی از توابعی که از آن به عنوان تابع فعال ساز استفاده می شود تابع $f(net) = \frac{net}{1+|net|}$ است. در مورد این تابع به سوالات زیر پاسخ دهید :

الف) نام علمی این تابع چیست مزایا و معایب استفاده از آن کدامند؟

ب)مشتق این تابع را بصورت تحلیلی حساب کنید .

ج)خود تابع و مشتق آن را به ازای $net \in [-7, 7]$ توسط نرم افزار MATLAB رسم کنید.

پاسخ :الف)

به این تابع ، سیگموید سریع می گویند که تقریب نسبتا خوبی از تابع سیگموید اصلی می باشد و از آنجا که تقریب است دقت محاسبات را کاهش می دهد اما به علت راحتی محاسبه آن در نورون ها (تابع e تابع سنگینی برای محاسبات است و زمانی نسبتا زیاد برای محاسبه خرج خواهد شد) از این تقریب بعه علت نسبتا دقیق بودن آن ، می توان استفاده کرد.در جدول زیر چند تابع سیگموید و مدت زمان محاسبه آن ها را مشاهده می کنید.

On my Core i5-3317U with GCC 4.7.2:

```
% gcc -Wall -O2 -lm -o sigmoid-bench{,.c} -std=c99 && ./sigmoid-bench
atan(pi*x/2)*2/pi   24.1 ns
atan(x)             23.0 ns
1/(1+exp(-x))       20.4 ns
1/sqrt(1+x^2)       13.4 ns
erf(sqrt(pi)*x/2)    6.7 ns
tanh(x)              5.5 ns
x/(1+|x|)            5.5 ns
```

ب)

$$f'(net) = \frac{\partial f}{\partial net} = \frac{1 * (1 + net) - 1 * (net)}{(1 + net)^2} = \frac{1}{(1 + net)^2} \; ; net > 0$$

$$f'(net) = \frac{\partial f}{\partial net} = \frac{1 * (1 - net) - 1 * (-net)}{(1 + net)^2} = \frac{1}{(1 + net)^2} \; ; net \leq 0$$

از آنجا که تابع در صفر پیوسته است پس مشتق در صفر نیز همان است پس مشتق تابع :

$$f'(net) = \frac{\partial f}{\partial net} = \frac{1}{(1+net)^2}$$

ج)

کد:draw.m

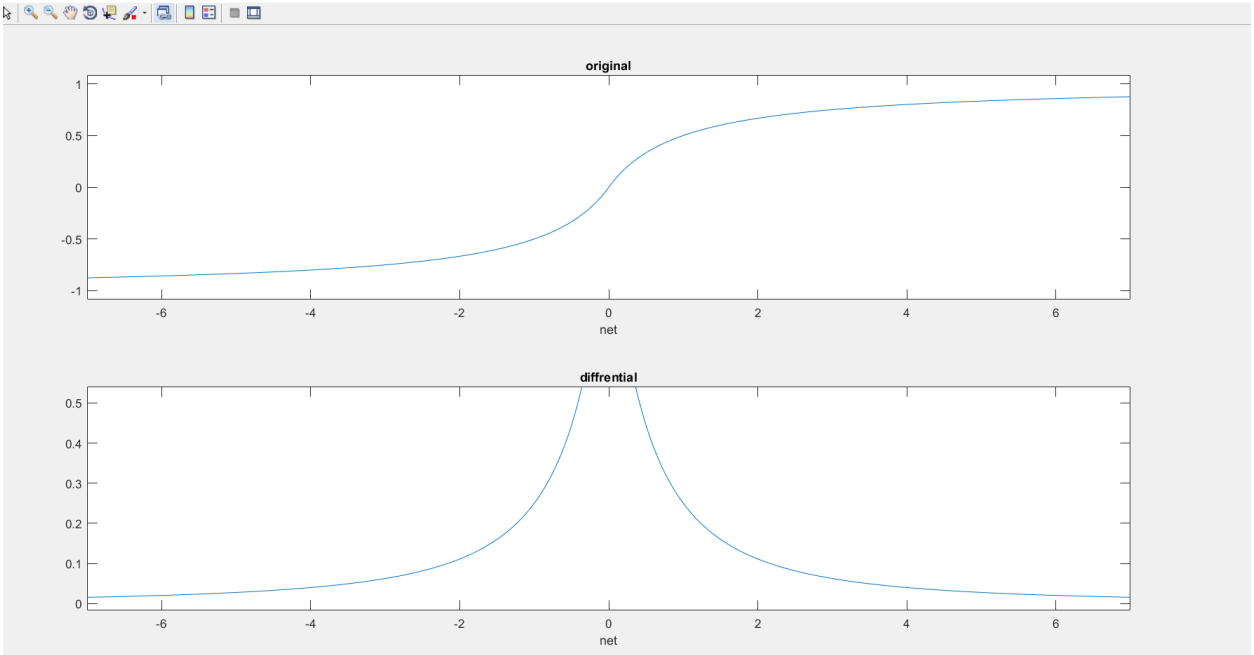
```
Command Window
>> syms net y
y=((net)/(1+abs(net)));
syms y_prime
y_prime=diff(y)

y_prime =

1/(abs(net) + 1) - (net*sign(net))/(abs(net) + 1)^2

>> subplot(2,1,1);ezplot(y,[-7,7]);title('original');
subplot(2,1,2);ezplot(y_prime,[-7,7]);title('diffrential');
>>
```

رسم:


original
1
0.5
0
-0.5
-1
-6
-4
-2
0
2
4
6
net
diffrential
0.5
0.4
0.3
0.2
0.1
0
-6
-4
-2
0
2
4
6
net

تمرین 4 :

روزنبلات اثبات کرده است که اگر داده های ورودی قابلیت جداسازی خطی را داشته باشند ، با استفاده از تابع فعالساز hardlimit می توان داده های ورودی را بصورت خطی جدا سازی نمود .الگوریتمی که وی ارائه کرده عبارت است از:

مسیر پیشرو :

$$\mathbf{y} = f\left(\mathbf{WX}\right)$$

آموزش پس انتشاردسته ای (batch) :

$$\mathbf{W}(k+1) = \mathbf{W}(k) + \eta \mathbf{e}(k)\mathbf{X}^T$$

خطای کل هر دوره آموزش :

$$E(k) = \frac{1}{2} tr\left(e(k)e(k)^T\right)$$

بطوریکه $X_{n_1*p} = \left[x^1_{n_1*1} \quad x^2_{n_1*1} \quad \cdots \quad x^p_{n_1*1}\right]_{n_1*p}$ **ماتریس ورودی،** n_1 **بعد هر الگوی ورودی و p تعداد کل الگوهای ورودی است . دقت کنید که** n_1 **شامل ورودی واحد به عنوان بایاس نیز می باشد.**
$W_{n_2*n_1}$ **ماتریس ورودی** n_2 **تعداد خروجی ها ،** e_{n_2*p} **بردار خطا برای تمام داده های ورودی ،** y_{n_2*p}

ماتریس خروجی برای تمام الگو های ورودی ، η نرخ یادگیری ، **k** شمارنده دوره آموزش و **f** تابع فعالساز **hardlimit** است.

با توجه به توضیحات فوق یک شبکه عصبی تک لایه با دو ورودی و دو خروجی طراحی کنید تا نقاط زیر را کلاس بندی کند :

برچسب	ورودی ها			خروجی مطلوب
گروه ۱	$(0.1,1.2)$	$(0.7,1.8)$	$(0.8,1.6)$	$(1,0)$
گروه ۲	$(0.8,0.6)$	$(1.0,0.8)$	-	$(0,0)$
گروه ۳	$(0.3,0.5)$	$(0.0,0.2)$	$(-0.3,0.8)$	$(1,1)$
گروه ۴	$(-0.5,-1.5)$	$(-1.5,-1.3)$	-	$(0,1)$

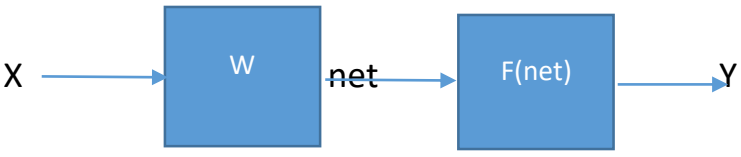
الف) ماتریس وزن را تصادفی انتخاب کنید و تا دو دوره آموزشی آنرا آموزش دهید و ماتریس وزن ، خروجی ، خطا را در هر دوره گزارش کنید .

ب) خروجی شبکه و خطای آنرا به ازای ماتریس وزن زیر بدست آورید :

$$W_{best} = \begin{bmatrix} -5.1679 & 7.4033 & -0.8295 \\ -5.1534 & -4.6383 & 4.0994 \end{bmatrix}$$

ج)برنامه ای نوشته تا آموزش را دقیقا تا خطای صفر ادامه دهد ، چند مرحله نیاز است هر مرحله را نمایش دهید .

پاسخ :



ما دو نورون داریم که هر نورون برای کار خاصیست ، اگر ما ورودی ها را نقاط دستگاه مختصاتی فرض کنیم ، نورون اول برای پیدا کردن x ها و نورون دوم برای پیدا کردن y ها است . که ما آنرا اگر در ماتریس ورودی که به صورت :

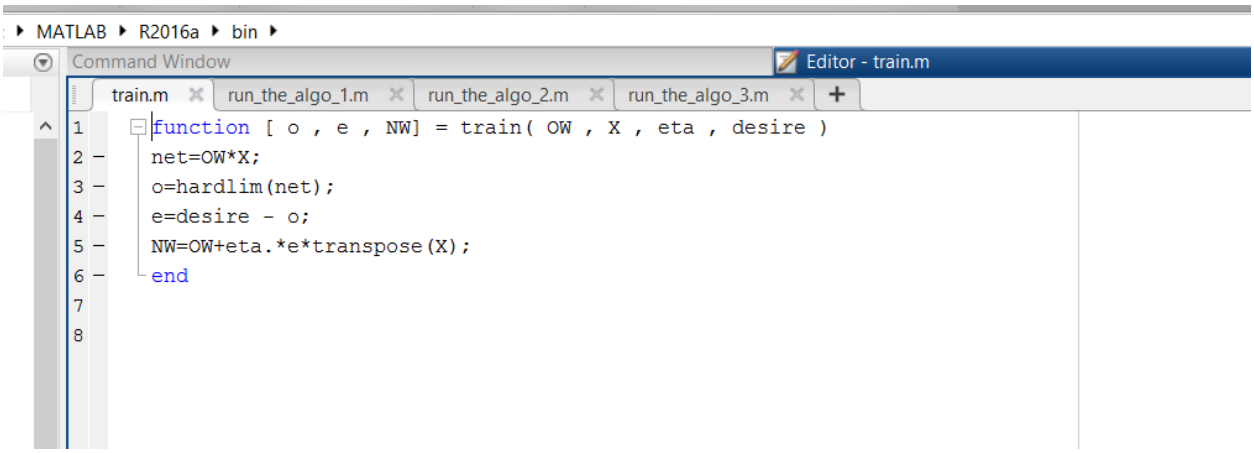
X =

1.0000 1.2000 0.1000
1.0000 1.8000 0.7000
1.0000 1.6000 0.8000
1.0000 0.6000 0.8000
1.0000 0.8000 1.0000
1.0000 0.5000 0.3000
1.0000 0.2000 0
1.0000 0.8000 -0.3000
1.0000 1.5000 -0.5000
1.0000 1.3000 -1.5000

ضرب کنیم و بر روی آن تابع هارد لیمیت را اجرا کنیم و از مقدار دلخواه کم کنیم مقدار خطا را خواهیم داشت .

مقدار یک که در ماتریس ورودی می بینیم در وزن بایاس ضرب شده و آنرا به ما نشان می دهد.

تابع آموزش ما به صورت زیر می شود : (train.m)



که NW نمایانگر وزن جدید و OW نمایانگر وزن قدیم و O نمایانگر خروجی و e مقدار خطا ما می باشد

برای پاسخ به پرسش اول

الف)

اسکریپت (run_the_algo_1.m) را نوشتیم که ابتدا یک وزن رندوم تولید می کند ، سپس برای دو دوره آموزش آنرا برای تمام ورودی ها انجام می دهد.

```
X=[0.1 1.2 1 ; 0.7 1.8 1 ; 0.8 1.6 1 ; 0.8 0.6 1 ; 1 0.8 1 ; 0.3 0.5 1 ; 0 0.2 1 ; -0.3 0.8 1 ; -0.5 -1.5 1 ; -1.5 -1.3 1];
desire=[1 0 ; 1 0 ; 1 0 ; 0 0 ; 0 0 ; 1 1 ; 1 1 ; 1 1 ; 0 1 ; 0 1 ];
eta = 0.8;
OW=rand(2,3)./2;
F=OW
for(j=1:1:2)
s=[];
k=[];
for(i=1:1:10)
[ o , e , NW] = train( OW , transpose(X(i,:,:)) , eta , transpose(desire(i,:,:)) );
s=[s;e(1,1) e(2,1)];
k=[k;o(1,1) o(2,1)];
OW=NW;
end
s
k
NW
E=0.5.*trace(s*transpose(s))
disp('============================');
end
```

که یک نمونه اجرای آن می شود :

```
>> run_the_algo_1
```

سپس بعد اجرا وزن های رندوم ما می شود :

```
F =

    0.0379    0.2654    0.4670
    0.0270    0.3896    0.0650
```

که بعد از یک بار آموزش برای تمامی ورودی ها :

وزن جدید :

```
NW =

    0.0379    1.3854   -0.3330
   -0.0530    0.4696    0.8650
```

ما تریس خطای ما :

```
s =

     0    -1
     0     0
     0     0
    -1     0
     0     0
     1     1
     0     0
     0     1
    -1     0
     0     0
```

ماتریس خروجی ما :

```
k =

     1     1
     1     0
     1     0
     1     0
     0     0
     0     0
     1     1
     1     0
     1     1
     0     1
```

و مقدار خطا :

```
E =

     3
```

می باشد .

بعد از آموزش دوم بر روی تمامی ورودی ها:

وزن جدید :

```
NW =

   -0.3621    1.4654    0.4670
    0.1070   -0.0904    0.8650
```

ماتریس خطای ما :

```
s =

     0    -1
     0     0
     0     0
    -1     0
     0     0
     1     1
     1     0
     0     0
     0     0
     0     0
```

ماتریس خروجی ما :

```
k =

     1     1
     1     0
     1     0
     1     0
     0     0
     0     0
     0     1
     1     1
     0     1
     0     1
```

و مقدار خطای ما :

```
E =

    2.5000
```

می شود .

ب)

برای حل سوال بعدی ما اسکریپت (run_the_algo_2.m) را می نویسیم :

train.m | run_the_algo_1.m | run_the_algo_2.m | run_the_algo_3.m

```
X=[0.1 1.2 1 ; 0.7 1.8 1 ; 0.8 1.6 1 ; 0.8 0.6 1 ; 1 0.8 1 ; 0.3 0.5 1 ; 0 0.2 1 ; -0.3 0.8 1 ; -0.5 -1.5 1 ; -1.5 -1.3 1];
desire=[1 0 ; 1 0 ; 1 0 ; 0 0 ; 0 0 ; 1 1 ; 1 1 ; 1 1 ; 0 1 ; 0 1 ];
eta = 0.8;
OW=[-5.1679 7.4033 -0.8295 ; -5.1534 -4.6383 4.0994];
F=OW
for(j=1:1:1)
s=[];
k=[];
for(i=1:1:10)
[ o , e , NW] = train( OW , transpose(X(i,:,:)) , eta , transpose(desire(i,:,:)) );
s=[s;e(1,1) e(2,1)];
k=[k;o(1,1) o(2,1)];
OW=NW;
end
NW
s
k
E=0.5.*trace(s*transpose(s))
disp('============================');
end
```

که بعد از اجرای آن به صورت :

```
>> run_the_algo_2
```

نتیجه آن به نمایش در می آید که بعد از تحلیل آن متوجه می شویم که وزن های آموزش دیده خوبی هستند زیرا :

ماتریس خطای ما :

```
s =

     0     0
     0     0
     0     0
     0     0
     0     0
     0     0
     0     0
     0     0
     0     0
     0     0
```

همان طور که دیدیم ماتریس خطای ما کاملا صفر است ، پس ماتریس خروجی ما باید با ماتریس دلخواه ما برابر باشد که توجه می شویم چنین است :

```
k =

     1     0
     1     0
     1     0
     0     0
     0     0
     1     1
     1     1
     1     1
     0     1
     0     1
```

پس خطای کل ما نیز صفر می باشد :

```
E =

     0

==================================
```

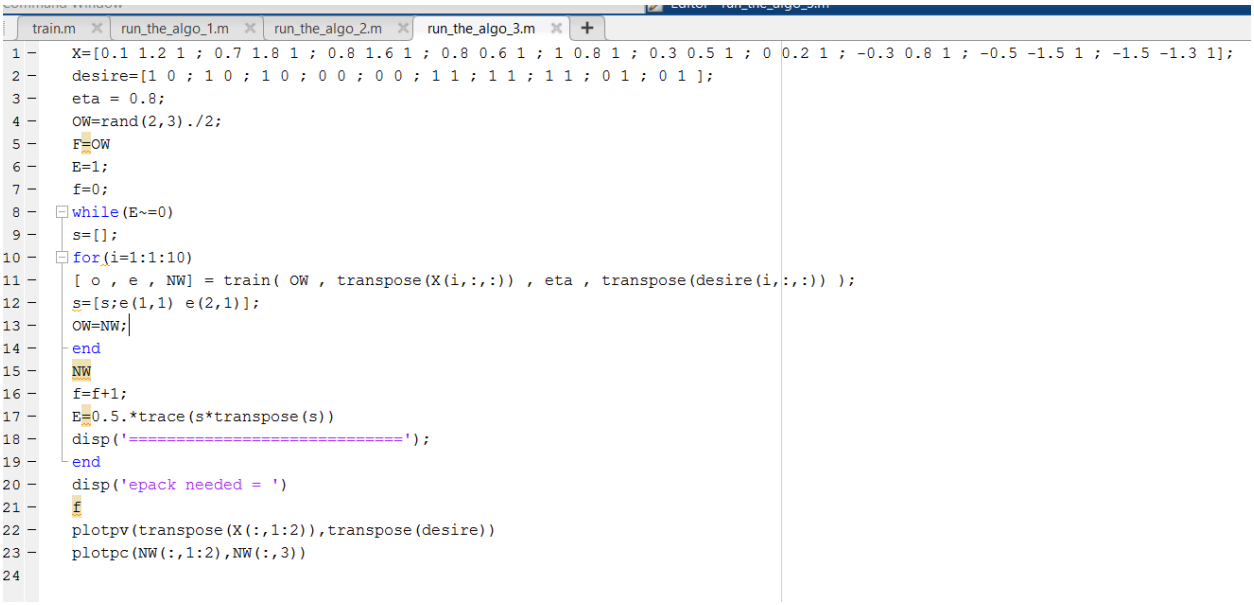
و وزن های ما تغییری نخواهد کرد :

```
NW =

   -5.1679    7.4033   -0.8295
   -5.1534   -4.6383    4.0994
```

ج)

برای حل سوال آخر نیز اسکریپت (run_the_algo_3.m) را می نویسیم که آموزش را تا آنجایی که خطای ما صفر شود ادامه دهد باید توجه داشت که چون تابع اعمالی به net ، تابع hardlimit می باشد در صورت بزرگ بودن ضریب آموزش (η) به مشکل خاصی دچار نخواهیم شد زیرا برد ما فقط یک و صفر می باشد پس تنها سرعت آموزش افزایش می یابد که مشکلی ندارد :



که بعد از اجرای آن به صورت :

```
>> run_the_algo_3
```

وزن های اولیه ی ما :

```
F =

    0.2844    0.0060    0.0811
    0.2347    0.1686    0.3971
```

می باشد .

و وزن های نهایی ما :

```
NW =

   -2.9156    3.6860   -0.7189
   -0.6453   -1.3514    1.1971


E =

     0
```

می باشد .

تعداد ایپاک مورد نیاز 20 می باشد:

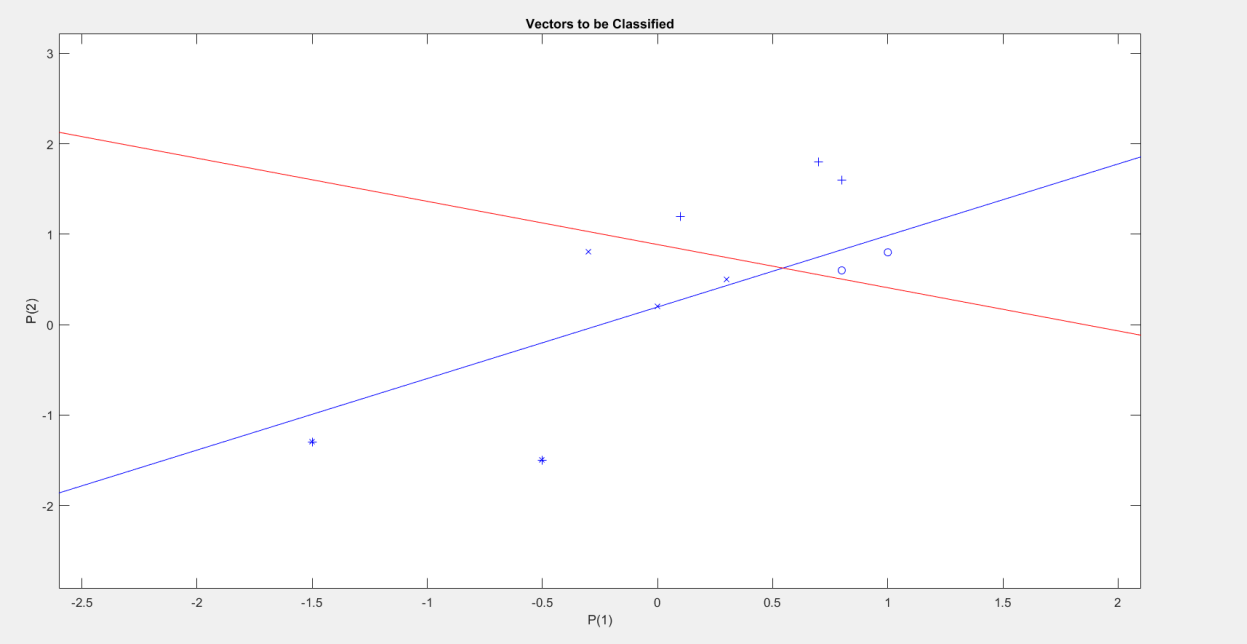
```
==============================
epack needed =

f =

    20
```

می باشد .

و نمودار دو بعدی داده ها به همراه خطوط جدا کننده :



می باشد .