Лабораторная работа №3

Исследование однослойного персептрона

1 Цель работы

Углубление теоретических знаний в области архитектуры нейронных сетей с пороговыми активационными функциями, исследование свойств однослойного персептрона и правила его обучения, приобретение практических навыков обучения и моделирования однослойной сети при решении простых задач классификации.

2 Основные теоретические положения

2.1 Структура однослойного персептрона

Общая структурная схема персептрона изображена на рисунке 3.1. Персептрон представляет собой однослойную сеть, которая состоит из *S* нейронов с *R* входами, характеризуется матрицей весов $\mathbf{W_{SxR}}$, и вектором смещений $\mathbf{b_{Sx1}}$, использует пороговую активационную функцию с жестким ограничением hardlim.

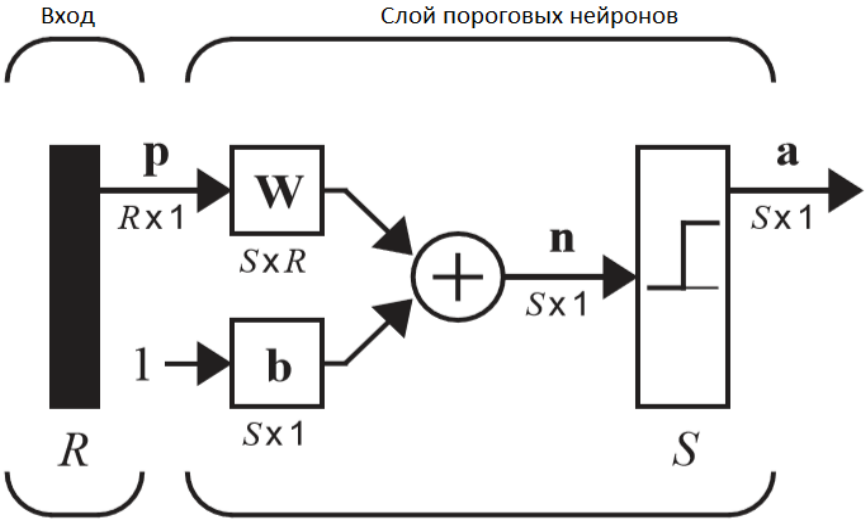


Рисунок 3.1 ─ Структурная схема однослойного персептрона

Каждая срока матрицы W соответствует набору весов одного из нейронов персептрона. Если обозначить *i*-ую строку матрицы как ${}_i\mathbf{w}^T$, то выход отдельного нейрона сети запишется в виде:

$$a_i = hardlim(n_i) = hardlim({}_i\mathbf{w}^T\mathbf{p} + b_i). \quad (3.1)$$

Если скалярное произведение i-ой строки матрицы весов ${}_i\mathbf{w}^T$ на входной вектор $\mathbf{p}$, будет больше или равно $-b_i$, то на выходе нейрона будет 1, иначе ─ 0 . Таким образом, каждый нейрон персептрона делит входное пространство

на две области. Персептрон, состоящий из S нейронов, способен распознать 2^S классов.

2.2 Граница решения

Рассмотрим персептрон из одного нейрона (**простой персептрон**) с 2-мя входами (рисунок 3.2.)

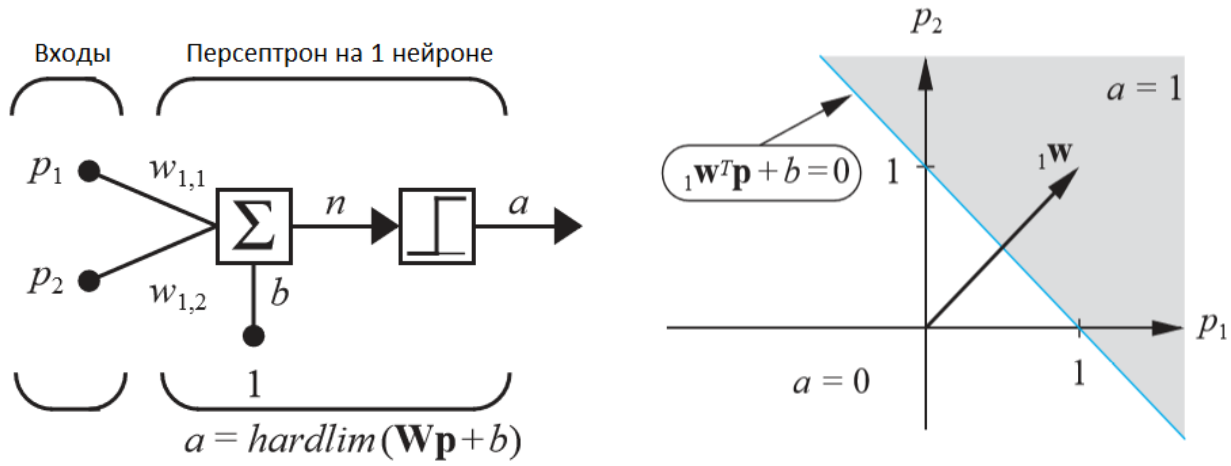


Рисунок 3.2 — Простой персептрон и его граница решения

С учетом только 2-х входов выходной сигнал нейрона запишется в виде:

$$a = hardlim({}_1\mathbf{w}^T\mathbf{p} + b) = hardlim(w_{1,1}p_1 + w_{1,2}p_2 + b) \quad (3.2)$$

Так как функция hardlim(n) при n=0 меняет значение с 0 на 1, то ***граница решения*** будет определяться условием:

$$n = {}_1\mathbf{w}^T\mathbf{p} + b = w_{1,1}p_1 + w_{1,2}p_2 + b = 0. \quad (3.3)$$

Пусть

$$w_{1,1} = 1, w_{1,2} = 1, b = -1.$$

Для этого случая граница решения запишется в виде уравнения прямой линии (рисунок 3.2.):

$$n = {}_1\mathbf{w}^T\mathbf{p} + b = w_{1,1}p_1 + w_{1,2}p_2 + b = p_1 + p_2 - 1 = 0.$$

Чтобы изобразить её, найдеч точки пересечения линии с осями координат (p_1, p_2):

$$p_1 = -\frac{b}{w_{1,1}} = -\frac{-1}{1} = 1 \qquad p_2 = -\frac{b}{w_{1,2}} = -\frac{-1}{1} = 1 \quad .$$

С одной стороны границы выход персептрона равен a=1, а с другой ─ a=0. Чтобы определить область, где выход персептрона будет равен 1, рассмотрим точку

$$\mathbf{p} = \begin{bmatrix} 2 & 0 \end{bmatrix}^T,$$

для которой a=1, действительно

$$a = hardlim\left(\begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 0 \end{bmatrix} - 1 \right) = 1.$$

Таким образом, часть входного пространства, где a=1, будет находиться выше границы решения (на рисунке 3.2. ─ затененная область).

2.3 Графическое построение границы решения

Границу решения можно также построить **графически**. Отметим, что ***граница всегда ортогональна вектору весов*** **w,** как изображено на рисунке 3.2. Действительно, граница решения определяется условием:

$$_1\mathbf{w}^T\mathbf{p} + b = 0. \tag{3.4}$$

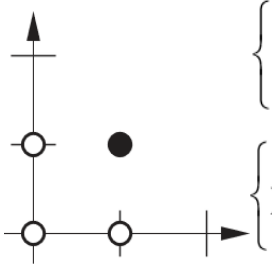
Для всех точек, принадлежащих границе, скалярное произведение входного вектора **p** на вектор весов **w** имеет одно и то же значение, равное -b. Это означает, что соответствующие входные векторы **p** имеют одно и то же значение своей проекции на вектор весов **w**, т.е. *концы таких векторов* **p** *должны лежать на линии ортогональной вектору весов*. Отметим, что *вектор весов всегда направлен в сторону области, где выход нейрона равен 1.*

После того как определено направление вектора весов **w**, значение **смещения** определяется путем выбора точки на границе и решения уравнения (3.4).

Пример.

Построим границу для случая реализации логической функции AND с помощью простого персептрона.

1. Зададим пары входных и выходных значений логического элемента AND в соответствии с рисунком 3.3.



$$\left\{\mathbf{p}_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, t_1 = 0\right\} \left\{\mathbf{p}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, t_2 = 0\right\}$$

$$\left\{\mathbf{p}_3 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, t_3 = 0\right\} \left\{\mathbf{p}_4 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, t_4 = 1\right\}.$$

Рисунок 3.3 ─ Входные и выходные значения логического элемента AND (зачерненная точка соответствует 1)

2. Выберем возможную границу решения (рисунок 3.4).

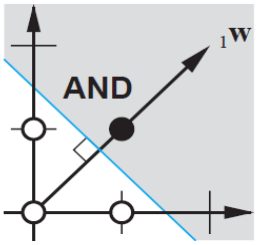


Рисунок 3.4 ─ Граница решения и вектор весов элемента AND

3. Построим вектор весов **w**, ортогональный границе. Вектор может иметь любую длину, например

$$_1\mathbf{w} = \begin{bmatrix} 2 \\ 2 \end{bmatrix}.$$

4. Найдем смещение *b*. Для этого выберем произвольную точку на границе и решим уравнение:

$$\mathbf{p} = \begin{bmatrix} 1.5 & 0 \end{bmatrix}^T \qquad {}_1\mathbf{w}^T\mathbf{p} + b = \begin{bmatrix} 2 & 2 \end{bmatrix}\begin{bmatrix} 1.5 \\ 0 \end{bmatrix} + b = 3 + b = 0 \qquad \Rightarrow \quad b = -3.$$

5. Проверим решение. Например, для $\mathbf{p}_2$ на выходе персептрона должен быть 0. Действительно

$$a = hardlim({}_1\mathbf{w}^T\mathbf{p}_2 + b) = hardlim\left(\begin{bmatrix} 2 & 2 \end{bmatrix}\begin{bmatrix} 0 \\ 1 \end{bmatrix} - 3\right) = hardlim(-1) = 0$$

2.4. Правило обучения персептрона

Правило обучения персептрона представляет собой процедуру **обучения с учителем** и заключается в поиске параметров (весов и смещений), которые обеспечивают совпадения желаемой **t** и действительной реакции персептрона **a** на заданный входной вектор **p**. Обучение осуществляется на основе обучающего **множества примеров** «вход-выход»:

$$\{\mathbf{p}_1, \mathbf{t}_1\}, \{\mathbf{p}_2, \mathbf{t}_2\}, \ldots, \{\mathbf{p}_Q, \mathbf{t}_Q\}.$$

В ходе обучения персептрона на его вход подаются входные векторы **p** из множества примеров, вычисляется реакция **a**, которая сравнивается с требуемым выходом **t** и вычисляется ошибка **e.** Например, для простого персептрона с одним выходом ошибка равна

$$e = t - a.$$

В зависимости от значения ошибки корректируется вектор весов персептрона в соответствии с правилом:

$$_1\mathbf{w}^{new} = {}_1\mathbf{w}^{old} + (t-a)\mathbf{p} = {}_1\mathbf{w}^{old} + e\mathbf{p} \qquad (3.5)$$

Для коррекции (обновления смещений) (смещение эквивалентно некоторому дополнительному входу с весом *b*, для которого p=1) используется упрощенное правило (3.5):

$$b^{new} = b^{old} + e\,. \tag{3.6}$$

В том случае, когда персептрон содержит несколько нейронов приведенные правила обучения можно записать в матричной форме:

$$\begin{aligned} \mathbf{W}^{new} &= \mathbf{W}^{old} + \mathbf{e}\mathbf{p}^{T}, \\ \mathbf{b}^{new} &= \mathbf{b}^{old} + \mathbf{e}\,. \end{aligned} \tag{3.7}$$

В соответствии с правилами (3.7) для каждого очередного вектора **p** из множества примеров новые значения матрицы весов $\mathbf{W}^{new}$ и вектора смещений $\mathbf{b}^{new}$ вычисляются на основе их предыдущих значений $\mathbf{W}^{old}$ и $\mathbf{b}^{old}$. Обычно **начальные значения весов и смещений** инициализируют небольшими случайными числами. Как следует из (3.5), вектор весов отдельного нейрона персептрона в ходе обучения изменяется на величину очередного входного вектора **p** с учетом ошибки, которая равна -1, 0 или 1.

Можно доказать, что ***правило обучения персептрона обеспечивает схождение весов к требуемым значениям, обеспечивающим правильную классификацию, за конечное число шагов при условии существования решения*** (теорема сходимости) [1, 5].

Сетевая функция **n** персептрона играет роль дискриминантной функции. В общем случае эта функция описывает уравнение гиперплоскости. Поэтому простой персептрон может классифицировать только такие образы входного пространства, которые разделимы с помощью гиперплоскости. Иными словами, задачи классификации, решаемые простым персептроном, – это **линейные сепарабельные задачи**.

3. Варианты заданий и программа работы

3.1. Повторить теоретический материал, относящийся к архитектуре персептрона и его обучению [1, 4, 5].

3.2. Для заданной матрицы входных данных **P** и заданного вектора выходных значений **T** (таблица 3.1) разработать простой персептрон, решающий задачу классификации 2-х классов. Решение найти с помощью графического построения границы решения и вычисления весов и смещения вручную. Протестировать с помощью компьютера полученное решение для всех входных векторов (столбцов матрицы **P**).

Таблица 3.1 – Варианты задания п. 3.2

Вариант	Матрица **P**	Вектор **T**
1	[0 1 1 0; 0 0 1 1]	[0 0 1 1]
2	[0 1 1 0; 0 0 1 1]	[0 0 1 1]
3	[0 1 1 0;	[1 1 0 0]

	0 0 1 1]	
4	[0 1 1 0; 0 0 1 1]	[1 0 0 1]
5	[0 1 1 0; 0 0 1 1]	[0 0 1 0]
6	[0 1 1 0; 0 0 1 1]	[0 0 0 1]
7	[0 1 1 0; 0 0 1 1]	[1 0 0 0]
8	[0 1 1 0; 0 0 1 1]	[0 1 0 0]
9	[0 1 1 0; 0 0 1 1]	[1 1 0 1]
10	[0 1 1 0; 0 0 1 1]	[1 1 1 0]
11	[0 1 1 0; 0 0 1 1]	[1 0 1 1]
12	[0 1 1 0; 0 0 1 1]	[0 1 1 1]
13	[0 1 1 0 2; 0 0 1 1 0]	[0 0 1 0 1]
14	[0 1 1 0 2; 0 0 1 1 0]	[0 1 0 0 1]
15	[0 1 1 0 2; 0 0 1 1 0]	[1 1 0 0 1]

3.3. Даны четыре класса, каждый из которых представлен 2-мя точками (столбцами матрицы **P**), указанными в таблице 3.3. Необходимо:

- разработать структурную схему персептрона, распознающего эти 4 класса;
- выполнить предварительный анализ задачи, изобразив все точки четырех классов и построив графически возможные границы решений персептрона;
- анализируя графическую границу решений, задать допустимые целевые значения выходов персептрона (определить матрицу **T**) для всех входных точек, представленных столбцами матрицы данных **P**;
- изучить функцию ann_PERCEPTRON и, используя её, обучить персептрон правильному распознаванию входных классов;
- написать программу, которая:
  - отображает диаграмму размещения входных точек из **P** на плоскости с координатами (*p1*, *p2*);
  - обучает персептрон;
  - накладывает на диаграмму входных точек границы решения после обучения персептрона;
  - выполняет тестирование полученного решения для всех заданных входных данных, а также для дополнительно выбранных точек из областей принадлежности входных классов.

Таблица 3.3 – Варианты задания п.3.3

Вариант	Класс 1 **P(:,1), P(:,2),**	Класс 2 **P(:,3), P(:,4),**	Класс 3 **P(:,5), P(:,6),**	Класс 4 **P(:,7), P(:,8),**	
1	[1; 1], [2; 2]	[-1; 1], [-2; 2]	[-1; -1], [-2; -2]	[1; -1], [2; -2]	
2	[2; 1], [2; 2]	[-2; 1], [-2; 2]	[-2; -1], [-2; -2]	[2; -1], [2; -2]	
3	[1; 2], [2; 2]	[-1; 2], [-2; 2]	[-1; -2], [-2; -2]	[1; -2], [2; -2]	
4	[1; 1], [2; 2]	[-1; 1], [-2; 2]	[-1; 0], [-2; -1]	[1; 0], [2; -1]	
5	[2; 1], [2; 2]	[-2; 1], [-2; 2]	[-2; 0], [-2; -1]	[2; 0], [2; -1]	
6	[1; 2], [2; 2]	[-1; 2], [-2; 2]	[-1; -1], [-2; -1]	[1; -1], [2; -1]	
7	[1; 0], [2; 1]	[-1; 0], [-2; 1]	[-1; -1], [-2; -2]	[1; -1], [2; -2]	
8	[2; 0], [2; 1]	[-2; 0], [-2; 1]	[-2; -1], [-2; -2]	[2; -1], [2; -2]	
9	[1; 1], [2; 1]	[-1; 1], [-2; 1]	[-1; -2], [-2; -2]	[1; -2], [2; -2]	
10	[2; 1], [3; 2]	[-1; 1], [-2; 2]	[-1; -1], [-2; -2]	[2; -1], [3; -2]	
11	[3; 1], [3; 2]	[-2; 1], [-2; 2]	[-2; -1], [-2; -2]	[3; -1], [3; -2]	
12	[2; 2], [3; 2]	[-1; 2], [-2; 2]	[-1; -2], [-2; -2]	[3; -2], [3; -2]	
13	[1; 1], [2; 2]	[-2; 1], [-3; 2]	[-2; -1], [-3; -2]	[1; -1], [2; -2]	
14	[2; 1], [2; 2]	[-3; 1], [-3; 2]	[-3; -1], [-3; -2]	[2; -1], [2; -2]	
15	[1; 2], [2; 2]	[-2; 2], [-3; 2]	[-2; -2], [-3; -2]	[1; -2], [2; -2]	

3.4. Выполнить анализ полученных результатов, обратив внимание на то, что при каждом новом запуске функции обучения границы решения меняются. Сравнить результаты нескольких запусков программы. Сохранит ли программа работоспособность, если матрицу целевых значений выходов **T** формировать произвольным образом (например, используя в качестве целевого выходного вектора номер его класса в двузначном двоичном коде)?

3.5. Подготовить и защитить отчет по работе.

4. Методические рекомендации по выполнению работы

4.1. Модуль NeuralNetwork 3.0 пакета Scilab содержит встроенные функции для обучения и моделирования персептрона:

[w,b]= ann_PERCEPTRON(P,T) – функция обучения персептрона, код которой с комментариями приведен в приложении А;

a= ann_PERCEPTRON_run(P,w,b) – функция моделирования персептрона.

Ниже приведен пример использования этих функций для обучения и моделирования простого персептрона, реализующего логическую операцию AND:

```
//обучающие данные для элемента AND
P = [0 0 1 1 ;
       0 1 0 1];
 T = [0 0 0 1];
 // вызов функции обучения
[w,b] = ann_PERCEPTRON(P,T);
 Epoch: 1
 Epoch: 2
 Epoch: 3
 Epoch: 4
 Epoch: 5
 Epoch: 6
 // отображение рез-тов
 --> w
            w = 2.2113249 1.7560439
 --> b
           b = -2.9997789
 // проверка результатов обучения
 // вызов функции моделирования персептрона:
 // a = ann_hardlim_activ(w*P+repmat(b,1,size(P,2));
 a = ann_PERCEPTRON_run(P,w,b)
            a = 0. 0. 0. 1.
```

4.2. При выполнении задания 3.2 для графического построения границы решения простого персептрона и вычисления его весов и смещения следует руководствоваться примером, рассмотренным в п. 2.3 настоящей лабораторной работы. Для тестирования полученного решения необходимо использовать функцию моделирования персептрона:

a = ann_PERCEPTRON_run(P,w,b).

4.3 При выполнении задания 3.3 необходимо провести предварительный анализ расположения точек, представляющих классы, на плоскости ($p1$, $p2$) и нарисовать вручную возможные границы решения персептрона. Поскольку задано 4 класса, то персептрон должен состоять из 2-х нейронов, каждый из которых создаёт границу, которая делит входное пространство на 2 области. При этом первая граница разделяет множество входных примеров на 2 подмножества, а вторая граница разделяет эти подмножества еще раз на две части. Это позволяет персептрону успешно распознавать 4 заданных класса, которые должны быть линейно-разделимыми. Чтобы обучить этому персептрон, необходимо корректно сформировать матрицу целевых выходов персептрона **T**.

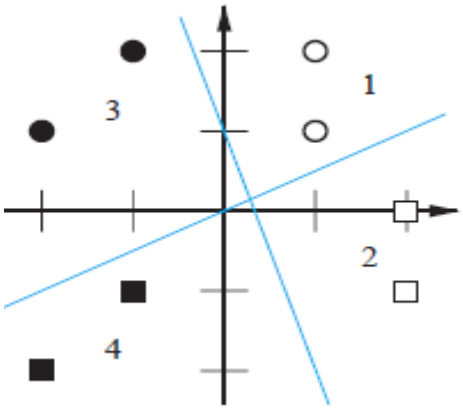


Рисунок 3.5

Рассмотрим пример. На рисунке 3.5 изображены «точки», представляющие входные классы (1, 2, 3, 4) и возможные границы решений персептрона. Необходимо сформировать целевые значения выходов двух нейронов персептрона. Для этого примем решение относительно того, с какой стороны границы на выходе нейрона будет 1, а с какой – 0. Пусть для границы, отделяющей зачерненные «точки» от не зачернённых, 1 будет со стороны зачерненных точек. Пусть для границы, отделяющей кружочки от квадратов, на выходе нейрона будет 1 со стороны, где изображены квадраты. Тогда векторы целевых значений выходов будут следующими:

класс 1: t = [0;0];
класс 2: t = [0;1];
класс 3: t = [1;0];
класс 4: t = [1;1].

Отсюда для заданной матрицы входных данных **P** (каждый столбец соответствует «точке» на рисунке)

P= [1 1 2 2 -1 -2 -1 -2;
1 2 -1 0 2 1 -1 -2]

матрица **T** целевых значений выходов будет равна:

T= [0 0 0 0 1 1 1 1;
0 0 1 1 0 0 1 1].

Теперь, используя значения матриц **P** и **T**, можно обучить персептрон, вызвав функцию обучения:

```
[w,b]=ann_PERCEPTRON(P,T)
```

Ниже приведен пример кода программы, обеспечивающей обучение персептрона для рассматриваемого примера, отображение диаграммы расположения точек и границ решений (рисунок 3.6):

```
clear //очистка рабочего пространства
K=4; //количество классов
N=2; //количество точек в классе
S=2; //число нейронов в слое

P=[ 1 1 2 2 -1 -2 -1 -2;
    1 2 -1 0  2  1 -1 -2];

T=[0 0 0 0 1 1 1 1;
   0 0 1 1 0 0 1 1];
```

```
// вызов функции обучения
[w,b] = ann_PERCEPTRON(P,T);

//построение диаграммы расположения входных точек
clf;
j=-3;
for i=1:K*N //цикл для всех точек данных из P
    // 3-й  аргумент plot2d ( , ,  j )  определяет отображаемый символ
   //  его допустимые значения -1,-2,…,-15
    plot2d(P(1,i),P(2,i),j);
    if modulo(i,2)==0 // если i четное
        j=j-1; // меняем j для отображения класса из P другим символом
    end
end
xgrid;
xtitle("Границы решений", "p1", "p2");
mtlb_hold('on')  // запрет очистки граф. окна

//отображение границ решений
p1 = min(P)-0.1:0.1:max(P)+0.1;
len=length(p1);
p2=zeros(S, len);
for i=1:S
    p2(i,:) = -w(i,1)/w(i,2).*p1 - b(i)./w(i,2); //уравнение прямой
    plot(p1,p2(i,:));
end

// тестирование персептрона
// вызов функции моделирования персептрона:
a = ann_PERCEPTRON_run(P,w,b)
a  =
  0.   0.   0.   0.   1.   1.   1.   1.
  0.   0.   1.   1.   0.   0.   1.   1.
```

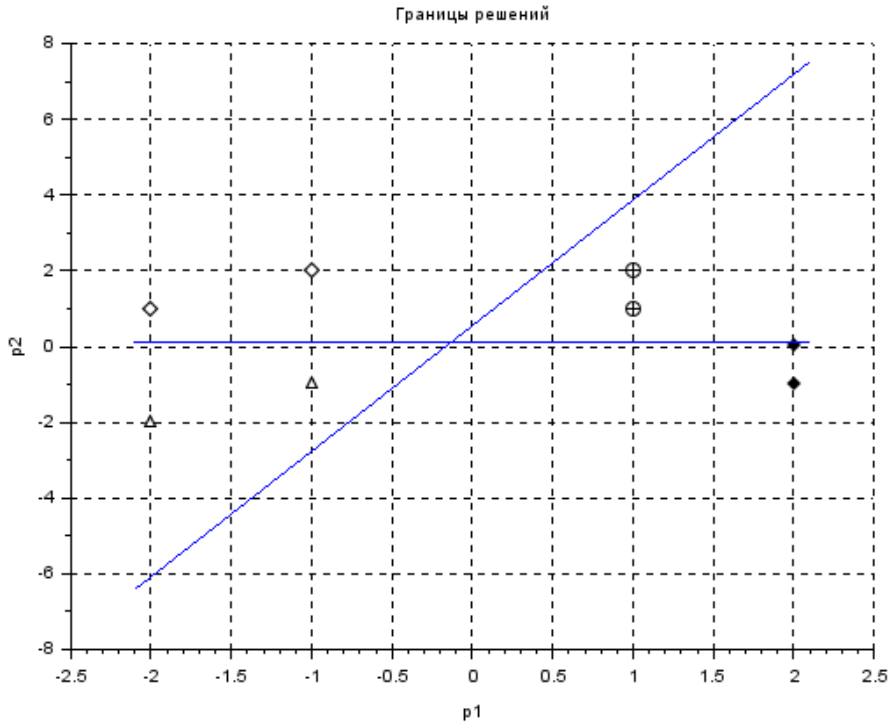


Рисунок 3.6 – Диаграмма расположения входных точек и границы решений

5. Содержание отчета

5.1. Цель работы.
5.2. Вариант задания.
5.3. Схема простого персептрона, решающего задачу классификации 2-х классов; графические построения границы решения и вычисления его весов и смещения, выполненные вручную; листинг программы с результатами тестирования простого персептрона.
5.4. Схема однослойного персептрона, решающего задачу классификации 4-х классов; пояснение выбора значений матрицы T; результаты обучения персептрона, правила обучения; диаграмма расположения классов с границами решений, соответствующими обученному персептрону; листинг программы с комментариями.
5.5. Выводы по результатам исследований.

6. Контрольные вопросы

6.1 Нарисуйте схему однослойного персептрона, объясните все обозначения, запишите формулу вычисления вектора выходных значений и объясните её.
6.2 Сколько линейно-разделимых классов способен распознать персептрон, содержащий S нейронов? Объясните почему?
6.3. Изобразите простой персептрон. Запишите уравнение границы решения.
6.4. Как построить границу решения простого персептрона, если известны его веса и смещения? Как определить область входного пространства, где выход персептрона будет равен 1?

6.5. Как можно графически построить границу решения персептрона. Приведите пример построения.
6.6. Как связана ориентация вектора весов с границей решения нейрона персептрона?
6.7. Запишите правило обучения персептрона в векторной и матричной форме?
6.8. Чему равны значения ошибки персептрона?
6.9. В каком направлении корректируются значения векторов весов нейронов персептрона?
6.10. Сформулируйте теорему сходимости персептрона.
6.11. Какой класс задач распознавания способен решать персептрон?
6.12. Как при известном расположении распознаваемых классов, корректно сформировать матрицу целевых значений выходов персептрона? Приведите пример.
6.13. Какие встроенные функции модуля Neuralnetwork 3.0 используются для обучения и моделирования персептрона? Приведите примеры их вызова.
6.14. Объясните алгоритм, реализуемый функцией ann_PERCEPTRON.

Приложение А. Код функции ann_PERCEPTRON(P, T)

```
function [w,b] = ann_PERCEPTRON(P, T)
//инициализация матрицы весов и вектора смещений
w = rand(size(T,1),size(P,1));
b = rand(size(T,1),1);
iter = %t; // присвоение флагу iter=True, контроль эпох
itercnt = 0; // счетчик итераций
while iter == %t  // цикл по эпохам, пока iter=True
no_err = 0          // счетчик безошибочных классификаций
   for cnt = 1:size(P,2) // цикл по всем входным примерам (одна эпоха)
      e = T(:,cnt) - ann_hardlim_activ(w*P(:,cnt)+b);  // вектор ошибки текущего
примера
      w = (w + e*P(:,cnt)'); // обучение матрицы весов
      b = b + e; // обучение вектора смещений
      if sum(e) == 0 then  // сумма текущих ошибок равна нулю
         no_err = no_err + 1;  // число безошибочных классификаций
      end
      if  no_err == size(P,2) then //число безошиб. классификаций = числу примеров
         iter = %f; // сбрасываем флаг цикла эпох
      end
   end
   itercnt = itercnt + 1; // счетчик эпох
   disp('Epoch: ' + string(itercnt));
end
endfunction
```

Список рекомендованной литературы

1. Бондарев В.Н. Искусственный интеллект: Учеб. пособие для студентов вузов / В. Н. Бондарев, Ф. Г. Аде. — Севастополь: Изд-во СевНТУ, 2002. — 613 с.
2. Ерин С.В. Scilab – примеры и задачи: практическое пособие / С.В. Ерин – М.: Лаборатория «Знания будущего», 2017. – 154 с.
3. Медведев, В.С. Нейронные сети. MATLAB 6 / В.С. Медведев, В.Г. Потемкин; под общ. ред. В.Г. Потемкина. — М.: ДИАЛОГ-МИФИ, 2002. — 496 с.
4. Хайкин С. Нейронные сети: Полный курс. Пер. С англ. / С. Хайкин. — М.: Изд. «Вильямс», 2006. — 1104 с.
5. Hagan M.T. Neural Network Design. The 2nd edition [Электронный ресурс] /M.T.Hagan, H.B.Demuth, M.H.Beale, O.D. Jesus. . — Frisco, Texas, 2014 . — 1012 p. Режим доступа: https://www.hagan.okstate.edu/NNDesign.pdf. —Последний доступ: 14.01.2019. —Название с экрана.