Севастопольский государственный университет
Институт информационных технологий
Дополнительная профессиональная программа профессиональной переподготовки «Глубокие нейросети в компьютерном зрении»

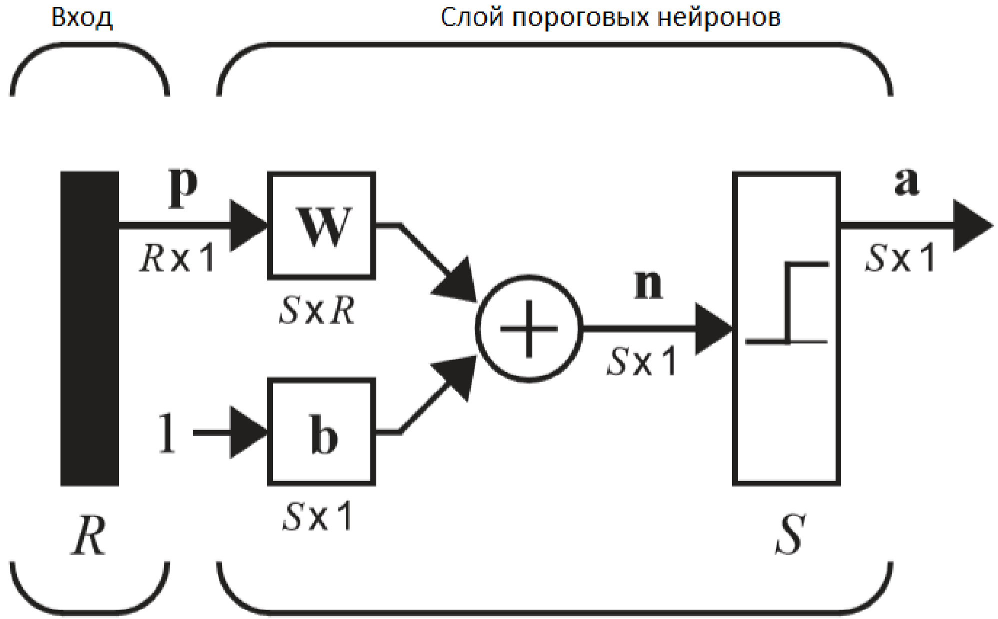
Основы нейронных сетей

Лекция 2
Персептрон: архитектура и правило обучения.

Бондарев Владимир Николаевич

Персептрон: архитектура и математическое описание

Общая архитектура персептрона (однослойного)



Выход сети :

$$\mathbf{a} = \mathbf{hardlim}(\mathbf{Wp} + \mathbf{b})$$

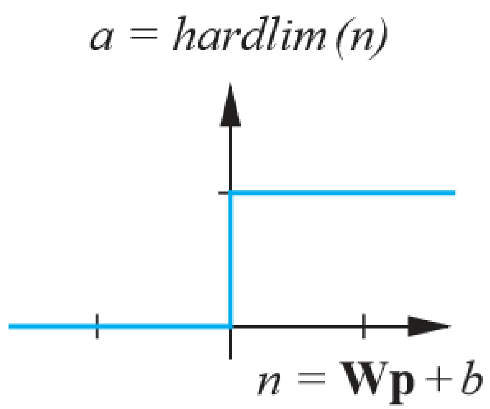
Персептрон: архитектура и математическое описание

Матрица весов персептрона

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & \dots & w_{1,R} \\ w_{2,1} & w_{2,2} & \dots & w_{2,R} \\ \vdots & \vdots & & \vdots \\ w_{S,1} & w_{S,2} & \dots & w_{S,R} \end{bmatrix}. \qquad {}_i\mathbf{w} = \begin{bmatrix} w_{i,1} \\ w_{i,2} \\ \vdots \\ w_{i,R} \end{bmatrix}. \qquad \mathbf{W} = \begin{bmatrix} {}_1\mathbf{w}^T \\ {}_2\mathbf{w}^T \\ \vdots \\ {}_S\mathbf{w}^T \end{bmatrix}.$$

Тогда выход отдельного нейрона сети будет равен:

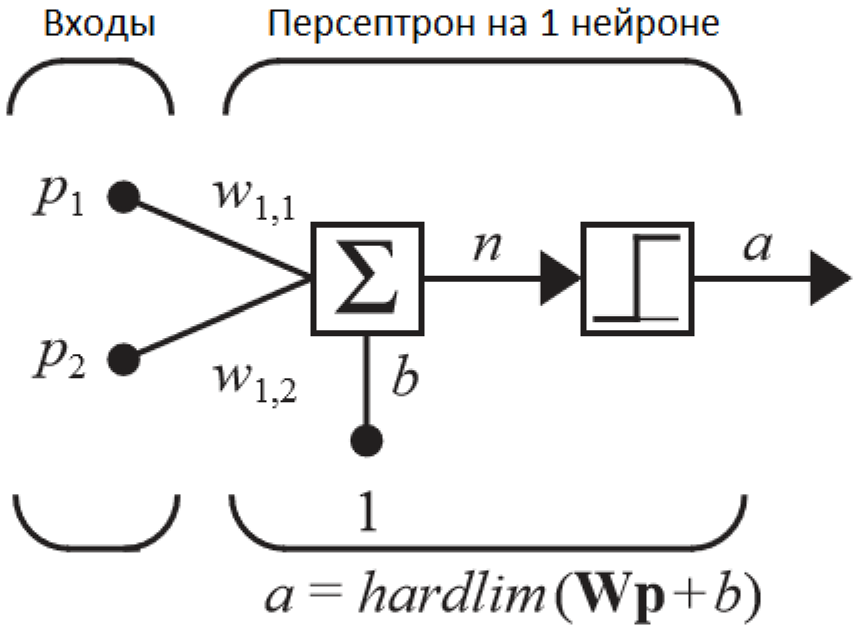
$$a_i = hardlim(n_i) = hardlim({}_i\mathbf{w}^T\mathbf{p} + b_i).$$



Если скалярное произведение i-ой строки матрицы ${}_i\mathbf{w}^T$ на входной вектор **p** ,будет больше или равно $-b_i$, то выход будет равен 1, иначе - 0.
Таким образом, каждый нейрон сети делит пространство входных сигналов на две области. Персептрон, состоящий из S нейронов, может классифицировать входные образы на 2^S классов.
Исследуем границы между эти областями.

Простой персептрон: граница решения

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



Выход нейрона:

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

Нейрон разделяет входное пространство на 2 области.

Граница решения между областями определится из условия:

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

Рассмотрим конкретный пример. Пусть

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

В.Бондарев

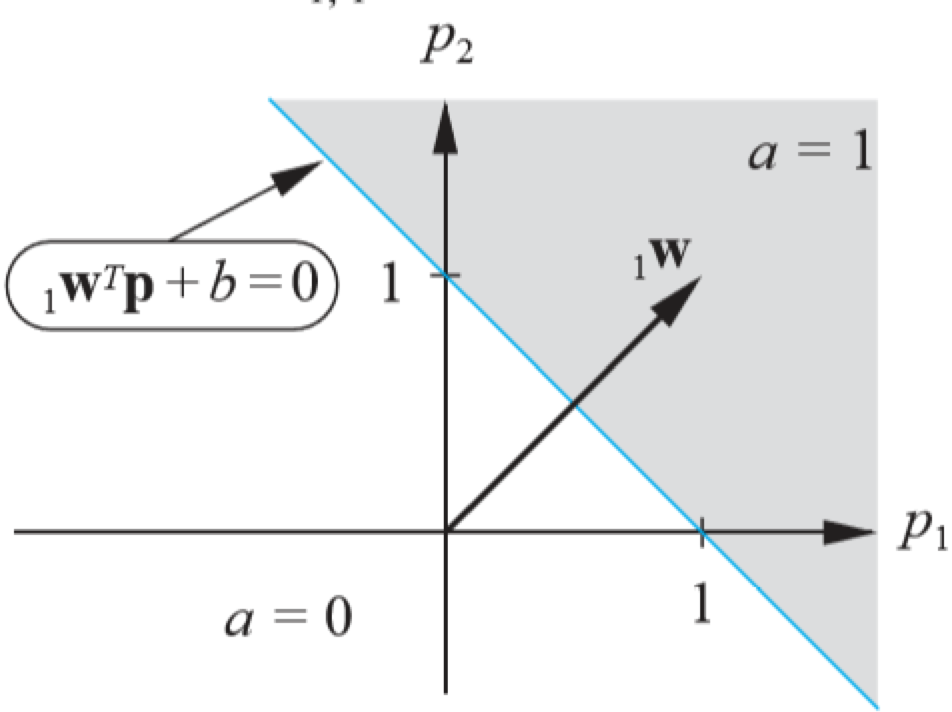
Простой перцептрон: граница решения

Тогда граница решения между областями определится из условия:

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

Это уравнение задаёт границу решения в виде линии. Чтобы изобразить линию, найдем точки её пересечения с осями (p_1, p_2):

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



С одной стороны границы выход равен 1, а с другой - равен 0. Например для точки

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

выход равен 1, т.к.

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

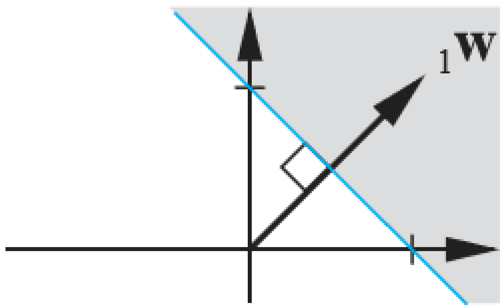
В.Бондарев

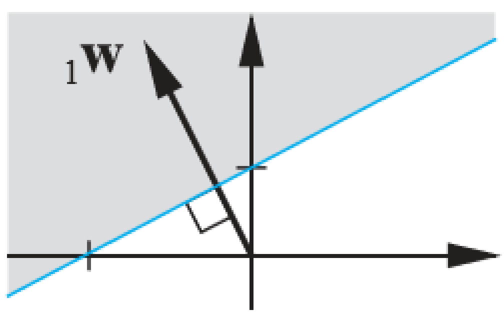
Граница решения: простой персептрон из одного нейрона

Мы также можем **построить границу графически**.
Отметим, что ***граница всегда ортогональна вектору*** $_1w$, как изображено на рис.

Действительно, граница решения определяется условием:

$$_1\mathbf{w}^T\mathbf{p} + b = 0 . \qquad (1)$$





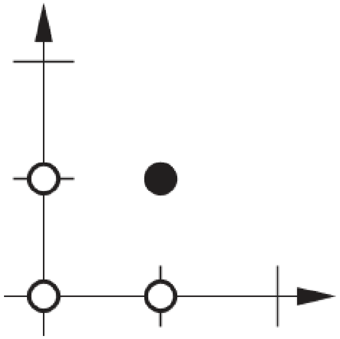
Для всех точек, принадлежащих границе, скалярное произведение входного вектора **p** на вектор весов имеет одно и то же значение. Это означает, что эти входные векторы **p** имеют одно и то же значение проекции на вектор весов , т.о., *их концы должны лежать на линии ортогональной вектору весов*.

Отметим, что ***вектор весов всегда направлен в строну области , где выход нейрона равен 1.***

После того как определено направление вектора весов, **значение смещения** определяется путем выбора точки на границе и решения уравнения (1)

Реализация функции AND на основе простого персептрона

1. Зададим последовательность входных и выходных сигналов логического элемента AND в соответствии с рисунком

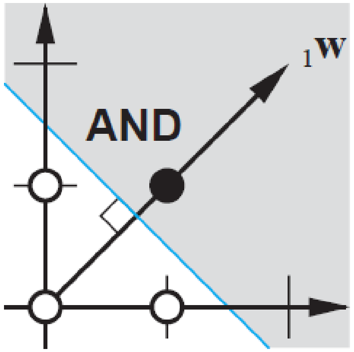


$$\left\{\mathbf{p}_1 = \begin{bmatrix}0\\0\end{bmatrix}, t_1 = 0\right\}\left\{\mathbf{p}_2 = \begin{bmatrix}0\\1\end{bmatrix}, t_2 = 0\right\}$$

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

2. Выберем возможную границу решения.
3. Построим вектор весов, ортогональный границе.
   Вектор может иметь любую длину, например:

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



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

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

5. Протестируем решение. Например для $\mathbf{p}_2$

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

В.Бондарев

Правило обучения простого 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 выполнения процедуры обучения персептрона на его вход подается каждый входной вектор **p** обучающего множества, вычисляется реакция *a*, которая сравнивается с требуемым выходом *t* и вычисляется ошибка *e*:

$$e = t - a.$$

В зависимости от значения ошибки корректируются веса персептрона :

$$e = 1, \qquad {}_1\mathbf{w}^{new} = {}_1\mathbf{w}^{old} + \mathbf{p}.$$

$$e = -1, \qquad {}_1\mathbf{w}^{new} = {}_1\mathbf{w}^{old} - \mathbf{p}.$$

$$e = 0, \qquad {}_1\mathbf{w}^{new} = {}_1\mathbf{w}^{old}.$$

Приведенные три правила можно записать в виде одного выражения:

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

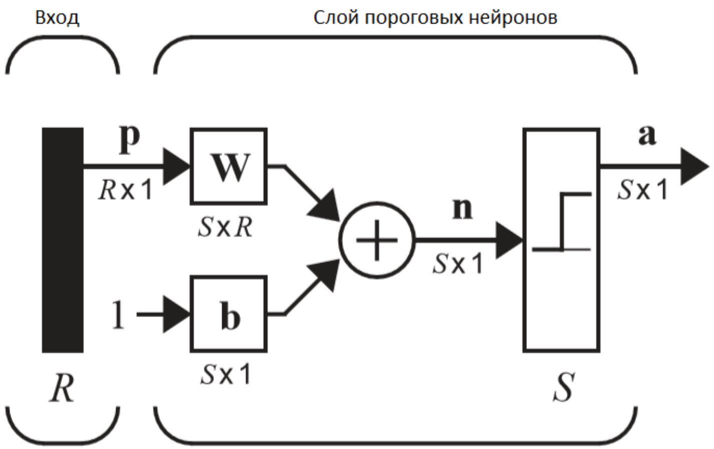
Для обновления смещений (~ весу, для которого p=1) используется выражение:

$$b^{new} = b^{old} + e.$$

В.Бондарев

Правило обучения   персептрона

Обобщим правила обучения простого однонейронного персептрона на случай мультинейронного персептрона.



Обновление весов *i*-го нейрона (*i*-я строка W):

$$ {}_i\mathbf{w}^{new} = {}_i\mathbf{w}^{old} + e_i\mathbf{p}\,. $$

Обновление смещений *i*-го нейрона:

$$ b_i^{new} = b_i^{old} + e_i\,. $$

Правило обучения мультинейронного персептрона в матричной записи:

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

Вариация правила →

$$ \boxed{\begin{aligned} \mathbf{W}^{new} &= \mathbf{W}^{old} + \alpha\mathbf{e}\mathbf{p}^T \\ \mathbf{b}^{new} &= \mathbf{b}^{old} + \alpha\mathbf{e} \end{aligned}} $$

Обычно **начальные значения весов и смещений** инициализируют небольшими случайными значениями.
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) .

Работа с векторами и матрицами в Scilab

```
--> xvec=[1 2 3]
xvec  =
   1.  2.   3.

--> yvec=[1;2;3]
 yvec  =
   1.
   2.
   3.

--> x=1:2:10
 x  =
   1.   3.   5.   7.
9.

--> x(2:4)
 ans  =
   3.   5.   7.
```

Работа с векторами и матрицами в Scilab

```
--> xvec=[1 2 3]
xvec  =
   1.  2.   3.

--> yvec=[1;2;3]
 yvec  =
   1.
   2.
   3.

--> x=1:2:10
 x  =
   1.   3.   5.   7.
9.

--> x(2:4)
 ans  =
   3.   5.   7.
```

```
--> m=[1 2 3;4 5 6]
 m  =
   1.   2.   3.
   4.   5.   6.

--> size(m)
 ans  =
   2.   3.

--> m(2,:)
 ans  =
   4.   5.   6.

--> m(:,3)
 ans  =
   3.
   6.
```

Работа с векторами и матрицами в Scilab

```
--> xvec=[1 2 3]
xvec  =
   1.  2.   3.

--> yvec=[1;2;3]
 yvec  =
   1.
   2.
   3.

--> x=1:2:10
 x  =
   1.   3.   5.   7.
9.

--> x(2:4)
 ans  =
   3.   5.   7.
```

```
--> m=[1 2 3;4 5 6]
 m  =
   1.   2.   3.
   4.   5.   6.

--> size(m)
 ans  =
   2.   3.

--> m(2,:)
 ans  =
   4.   5.   6.

--> m(:,3)
 ans  =
   3.
   6.
```

```
--> x=rand(3,4)
 x  =

   0.025871    0.2413538   0.2893728   0.3454984
   0.5174468   0.5064435   0.0887932   0.7064868
   0.3916873   0.4236102   0.6212882   0.5211472

--> x=rand(3,4,'normal')
 x  =

  -1.3189197  -0.1043591  -1.5404673   0.0075659
   0.9307226   0.2973099  -0.3966362   1.0422456
  -0.8575198   0.5308516   0.5163255   2.6705108

--> size(x,1)
 ans  =
   3.

--> size(x,2)
 ans  =
   4.
```

Управляющие конструкции Scilab
(if, for)

```
--> x=5
 x  =
   5.
--> if x>=0
         y=x
     else
         y=-x
     end
 y  =
   5.
```

```
--> x=5;

--> if x>=0
         y=x;
     else
         y=-x;
     end

--> y=5
 y=
   5.
```

```
--> x=20:-2:10
x  =
   20.   18.   16.   14.   12.   10.

--> s=0;
--> for i=1:1:length(x)
          s=s+x(i);
     end
--> s
 s  =
   90.

-->  sum(x)
 ans  =

   90.
```

Реализация правила обучения персептрона

size(T) = S x Q
size(P) = R x Q
size(w) = S x R

```
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 повторения эпох
epoch = 0; // счетчик эпох

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 для 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 стало = числу примеров
            iter = %f; // сбрасываем флаг цикла эпох
        end
    end
    epoch = epoch + 1; // счетчик эпох
    disp('Epoch: ' + string(epoch));
end

endfunction
```

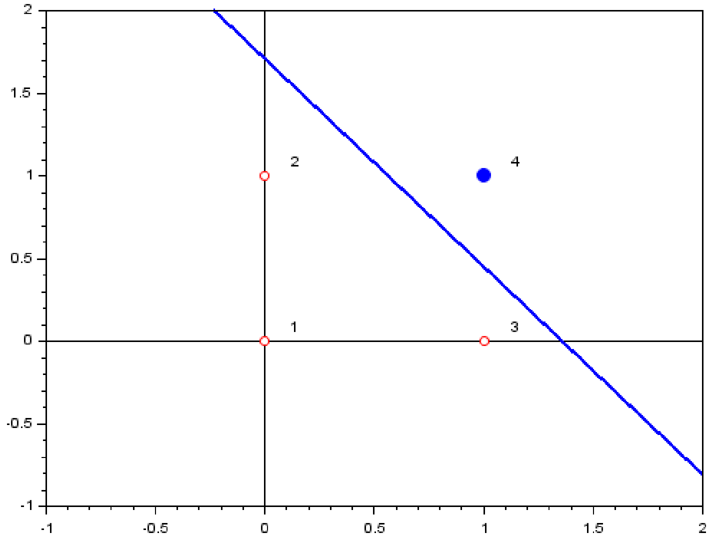
Тестирование функции ann_PERCEPTRON(P,T) на примере AND

```
// пример обучающей выборки для лог. эл-та 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
// тестирование ЛЭ AND
// вызов функции моделирования персептрона:
// y = ann_hardlim_activ(w*P+repmat(b,1,size(P,2));
y = ann_PERCEPTRON_run(P,w,b)
 y  =
   0.   0.   0.   1.
```

Функция обучения+ визуалиазия обучения

[w,b] = ann_PERCEPTRON_visualize(P,T, delay);



Ограничения персептрона

Сетевая функция простого персептрона играет роль дискриминантной функции. В общем случае эта функция описывает уравнение гиперплоскости:

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

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**.

Классификация в двумерном пространстве