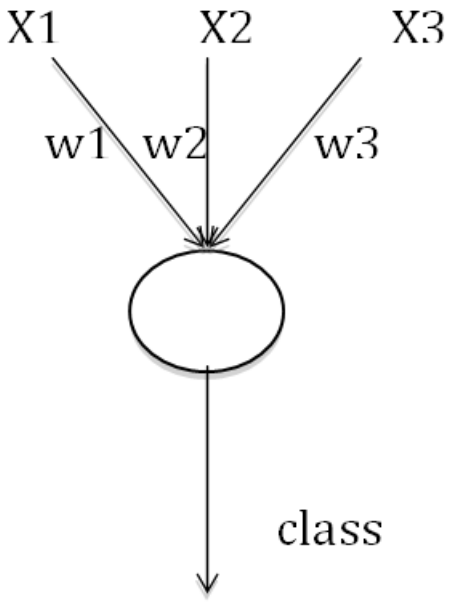
Perceptrons

Example: Learn an AND using the approximation of gradient descent:

	f_1	f_2	f_3	class
l=1	0	0	1	0
l=2	0	1	1	0
l=3	1	0	1	0
l=4	1	1	1	1



```
#include<stdio.h>
#include<math.h>

#define g(x) ((1.0/(1.0+exp(-x))))
#define gprime(x) ((g(x) * (1-g(x))))

main()
{
  float alpha = 0.01;
  int trainingexamples = 4;
  int features = 3;
  float f[4][3] = {{0, 0, 1}, {0, 1, 1}, {1, 0, 1}, {1, 1, 1}};
  float class[4] = {0,0,0,1};
  float w[3] = {1.1, -2.1, 0.3}; /* random values */

  int l, j, epoch;
  float weightedsum;

  for (epoch = 0; 1; ++epoch)
  {
    for (l = 0; l < trainingexamples; ++l)
    {
      weightedsum = 0.0;
      for (j = 0; j < features; ++j)
        weightedsum += w[j]*f[l][j];
      for (j = 0; j < features; ++j)
        w[j] -= alpha*(g(weightedsum) - class[l])*gprime(weightedsum)*f[l][j];
    }
    printf("epoch = %d, weights = ", epoch);
    for(j = 0; j < features; ++j)
      printf("%.2f", w[j]);
    printf(", outputs =");
    for(l = 0; l < trainingexamples; ++l)
    {
      weightedsum = 0.0;
      for(j = 0; j < features; ++j)
        weightedsum += w[j]*f[l][j];
      printf(" %.2f", g(weightedsum));
    }
    printf("\n");
  }
}
```

```
> gcc -lm learning.c
> ./a.out
epoch = 0, weights = 1.10 -2.10 0.30, outputs = 0.57 0.14 0.80 0.33
epoch = 1, weights = 1.10 -2.10 0.30, outputs = 0.57 0.14 0.80 0.33
epoch = 2, weights = 1.10 -2.10 0.30, outputs = 0.57 0.14 0.80 0.33
epoch = 3, weights = 1.10 -2.09 0.29, outputs = 0.57 0.14 0.80 0.33
epoch = 4, weights = 1.10 -2.09 0.29, outputs = 0.57 0.14 0.80 0.33
epoch = 5, weights = 1.10 -2.09 0.29, outputs = 0.57 0.14 0.80 0.33
...
epoch = 100, weights = 1.12 -1.97 0.16, outputs = 0.54 0.14 0.78 0.33
...
epoch = 1000, weights = 1.15 -0.80 -0.84, outputs = 0.30 0.16 0.58 0.38
...
epoch = 10000, weights = 2.56 2.55 -3.96, outputs = 0.02 0.20 0.20 0.76
...
epoch = 100000, weights = 5.47 5.47 -8.30, outputs = 0.00 0.06 0.06 0.93
```