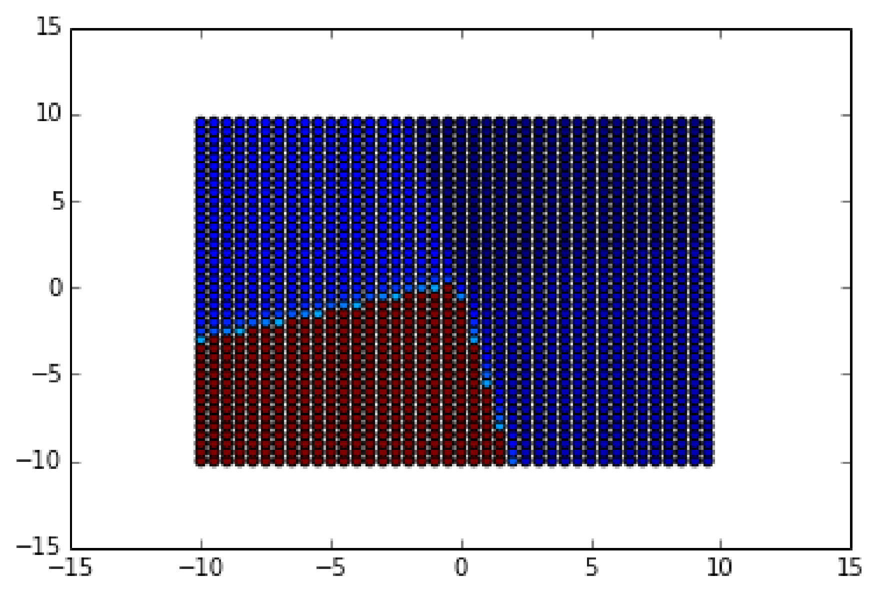
```
import sympy
import numpy
import itertools
```

```
v1 = sympy.Symbol(’v1’)
v2 = sympy.Symbol(’v2’)

def linear_function(v):
    if v >= 0.5:
        return 1
    elif v > -1.5:
        return v
    else:
        return 0

def activation_function(v):
    if v > 0:
        return 1
    elif v == 0:
        return 0
    else:
        return -1
```

```
#sigmoid case
v3 = 1/(1 + sympy.exp(-1*(5*v1 + v2 )))
v4 = 1/(1 + sympy.exp(-1*(v1 - 3*v2 )))
v5 = 1/(1 + sympy.exp(-1*(3*v3 - 1*v4 )))
v6 = 1/(1 + sympy.exp(-1*(4*v3 + 6*v4 )))
v7 = 1/(1 + sympy.exp(-1*(-2*v5 + v6 )))
v7.simplify()
sigmoid = v7
print v7
print ’’

f = sympy.lambdify((v1, v2), v7)


#build a grid
x = numpy.arange(-10, 10, 0.5)
y = x
xyf = []
for v in y:
    xyf.extend([[i,v] for i in x])


#include f(v1, v2)
for vs in xyf:
    vs.append(f(vs[0], vs[1]))
```

```
#apply linear function
for vs in xyf:
    vs[2] = linear_function(vs[2])
    #vs[2] = activation_function(vs[2])

#plot
scatter([a[0] for a in xyf], [a[1] for a in xyf], c=[a[2] for a in xyf])
```

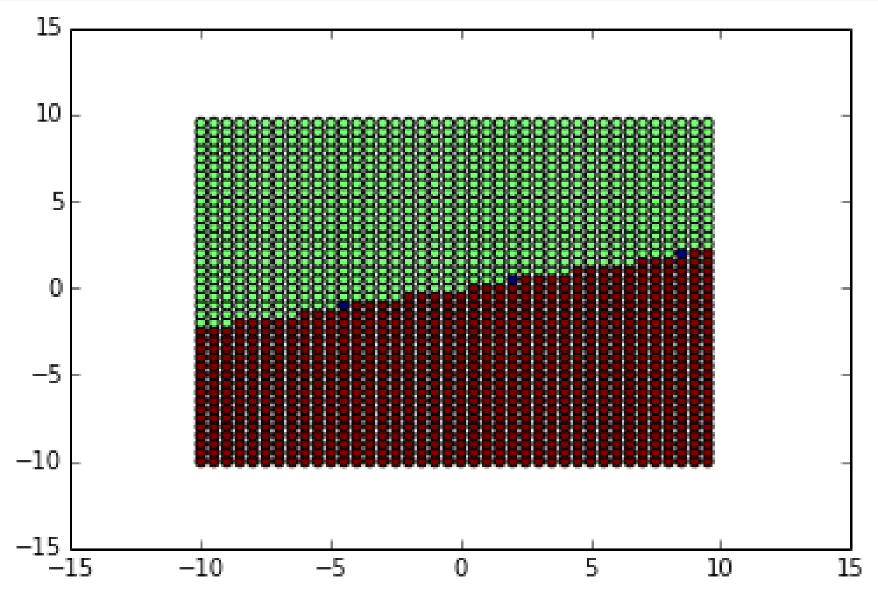
```
1/(exp(2/(exp(1/(exp(-v1 + 3*v2) + 1) - 3/(exp(-5*v1 - v2) + 1)) + 1) - 1/(exp(-6/(exp(-v1 + 3*

<matplotlib.collections.PathCollection at 0x80badd0>
```



```
#linear case
v3 = 5*v1 + v2
v4 = 2*v1 - 3*v2
v5 = 3*v3 - v4
v6 = 4*v3 + 6*v4
v7 = -2*v5 + v6
print v7


#Copy pasta
f = sympy.lambdify((v1, v2), v7)


#build a grid
x = numpy.arange(-10, 10, 0.5)
y = x
xyf = []
for v in y:
    xyf.extend([[i,v] for i in x])
```

```
#include f(v1, v2)
for vs in xyf:
    vs.append(f(vs[0], vs[1]))


#apply linear function
for vs in xyf:
    vs[2] = linear_function(vs[2])
    #vs[2] = activation_function(vs[2])

#plot
scatter([a[0] for a in xyf], [a[1] for a in xyf], c=[a[2] for a in xyf])
```

```
6*v1 - 26*v2

<matplotlib.collections.PathCollection at 0x7a03310>
```


15
10
5
0
−5
−10
−15
−15
−10
−5
0
5
10
15