```
from numpy import *
from matplotlib.pyplot import *
import scipy.stats
def log_likelihood(x, y, w, sigma):
    #Return the log-likelihood of the data (x, y) given weights w
    #(up to a constant)
    #
    #y ~ N(dot(w, x), sigma)

    return -sum((y-dot(w, x))**2/(2*sigma**2), 1)

def log_prior(w, sigma_w):
    #Return the log-prior probability for w, with
    #w_i ~ N(0, sqrt(1/(2*lam))^2)
    #
    return sum((-.5*log(2*pi*sigma_w**2)-(w**2/(2*sigma_w**2))), 1)

random.seed(0)
w = array([3, -1.5, 2, 1])


N = 20
sigma = 2.2
x_raw = 5*(random.random((N))-.5)
x = vstack((     ones_like(x_raw),
                 x_raw,
                 x_raw**2,
                 x_raw**3,
               ))

y = dot(w, x) + scipy.stats.norm.rvs(scale= sigma,size=N)
scatter(x_raw,y)

w_range = linspace(-5, 5, 40)
w0, w1, w2, w3 = [e.flatten() for e in meshgrid(w_range, w_range, w_range, w_range)]
all_w = vstack((w0, w1, w2, w3)).T

#Compute log P(w|data) (up to a constant)
logPw = log_likelihood(x, y, all_w, sigma)+log_prior(all_w, 12)
Pw = (exp(logPw)/(sum(exp(logPw)))).reshape((logPw.shape[0],1))

all_pred_y = dot(all_w, x)
prob_weighted_all_pred_y = all_pred_y * tile(Pw, (1,all_pred_y.shape[1]))

pred_y = sum(prob_weighted_all_pred_y,0)
```