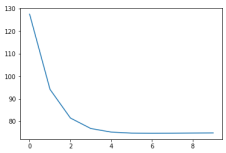
Welcome to Jupyter!

In [3]:
```
#Sequential Learning Question 2

%matplotlib inline
import numpy as np
import math as m
import matplotlib.pyplot as plt

def train(X,T,W,eta):
    phi = []
    wnew = W
    for i in range(int(X.size*0.66)):
        phi = np.array([1,m.exp(-10.0*(X[i]-1.0)**2),m.exp(-10.0*(X[i]+1.0)**2)])
        e = (T[i] - phi[0]*wnew[0] - phi[1]*wnew[1] -phi[2]*wnew[2])
        wnew = wnew + eta*e*phi
    return wnew

def compSSE(X,W,T):
    i = (int(X.size*0.66))+1
    e =0
    for i in range(X.size):
        phi = np.array([1,m.exp(-10.0*(X[i]-1.0)**2),m.exp(-10.0*(X[i]+1.0)**2)])
        e += (T[i] - phi[0]*W[0] - phi[1]*W[1] -phi[2]*W[2])**2
    return e

X = np.linspace(-1.0,1,21)

xout = np.linspace(-0.95,0.95,21)

T = np.array([5.12,4.97,4.92,4.83,4.90,5.06,5.29,5.34,5.36,5.76,5.99,6.30,6.66,6.70,7.49,7.92,8.48,9.09,9.7,10.3,10.98])

truearray = np.array([5,4.92,4.88,4.88,4.92,5,5.12,5.28,5.48,5.72,6,6.32,6.68,7.08,7.52,8,8.52,9.08,9.68,10.32,11])

tmean = np.mean(truearray)

W = np.zeros(3)

SSE = []
for i in range(10):
    W = train(X,T,W,0.4)
    SSE.append(compSSE(X,W,T))

print(W)

Y = []
plt.plot(range(10),SSE)
"""for i in range(xout.size):
    phi = (np.array([1,m.exp(-10.0*(xout[i]-1.0)**2),m.exp(-10.0*(xout[i]+1.0)**2)]))
    Yi = phi.dot(W)
    Y.append(Yi)

plt.plot(xout,Y)
plt.plot(X,truearray)
"""
plt.show()
```

```
[ 6.23521639e+00  5.67140433e-03 -1.99026668e+00]
```

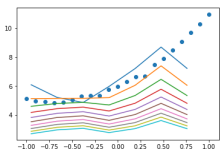


This repo contains an introduction to [Jupyter](https://jupyter.org) and [IPython](https://ipython.org).

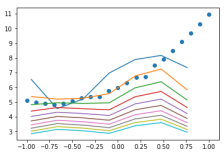
Outline of some basics:

* [Notebook Basics](../examples/Notebook/Notebook%20Basics.ipynb)
* [IPython - beyond plain python](../examples/IPython%20Kernel/Beyond%20Plain%20Python.ipynb)
* [Markdown Cells](../examples/Notebook/Working%20With%20Markdown%20Cells.ipynb)
* [Rich Display System](../examples/IPython%20Kernel/Rich%20Output.ipynb)
* [Custom Display Logic](../examples/IPython%20Kernel/Custom%20Display%20Logic.ipynb)
* [Running a Secure Public Notebook Server](../examples/Notebook/Running%20the%20Notebook%20Server.ipynb#Securing-the-notebook-server)
* [How Jupyter works](../examples/Notebook/Multiple%20Languages%2C%20Frontends.ipynb) to run code in different languages.

In [18]:
```
%matplotlib inline
import numpy as np
import math
import matplotlib.pyplot as plt

def train(X1,xout1,T1,l):
    phi = []
    phix = []

    for i in range(7):
        phi1 = 1
        phi2 = math.exp(-10*(X1[i]-0.5)**2)
        phi3 = math.exp(-10*(X1[i]+0.5)**2)
        phi.append([phi1,phi2,phi3])

    for i in range(7):
        phi1 = 1
        phi2 = math.exp(-10*(xout[i]-0.5)**2)
        phi3 = math.exp(-10*(xout[i]+0.5)**2)
        phix.append([phi1,phi2,phi3])


    phi = np.array(phi)
    phix = np.array(phix)

    phixT = phix.T
    phiT = phi.T


    phiphiT = phiT.dot(phi)
    phixphixT = phixT.dot(phix)

    lbda = l

    reg = np.identity(3)
    reg = lbda*reg

    u = phiphiT + reg

    v = np.linalg.inv(u)

    b = phi.dot(v)

    b = b.T

    w = b.dot(T1)

    y = phi.dot(w)

    return y,y



X = np.linspace(-1.0,1,21)

xout = np.linspace(-0.95,0.95,21)

T = np.array([5.12,4.97,4.92,4.83,4.90,5.06,5.29,5.34,5.36,5.76,5.99,6.30,6.66,6.70,7.49,7.92,8.48,9.09,9.7,10.3,10.98])

truearray = np.array([5,4.92,4.88,4.88,4.92,5,5.12,5.28,5.48,5.72,6,6.32,6.68,7.08,7.52,8,8.52,9.08,9.68,10.32,11])

tmean = np.mean(truearray)


X1 = []
xout1 = []
X2 = []
xout2 = []
X3 = []
xout3 = []
T1 = []
T2 = []
T3 = []

i = 0

while i<21:
    X1.append(X[i])
    xout1.append(xout[i])
    T1.append(T[i])
    X2.append(X[i+1])
    xout2.append(xout[i+1])
    T2.append(T[i+1])
    X3.append(X[i+2])
    xout3.append(xout[i+2])
    T3.append(T[i+2])
    i += 3

X1 = np.array(X1)
T1 = np.array(T1)
X2 = np.array(X2)
T2 = np.array(T2)
X3 = np.array(X3)
T3 = np.array(T2)
xout1 = np.array(xout1)
xout2 = np.array(xout2)
xout3 = np.array(xout3)
yr = []
xd = []
xr = []

xr = np.array([])
xd = np.array([])
yr = np.array([])

X1 = np.append(X1,X2)
X1 = np.append(X1,X3)
xd = np.append(xd,X1)

Ylist = []

for i in range(10):
    Y1,yr1 = train(X1,xout1,T1,i)
    Ylist.append(Y1)
    Y2,yr2 = train(X2,xout2,T2,i)
    Y3,yr3 = train(X3,xout3,T3,i)
    #Y1 = np.append(Y1,Y2)
    #Y1 = np.append(Y1,Y3)
    #yr = np.append(yr,Y1)
    #xr = np.append(xr,xd)

sc = plt.scatter(X,T)
for i in range(10):
    plt.plot(xout1,Ylist[i])

plt.show()
```



In [19]:
```
%matplotlib inline
import numpy as np
import math
import matplotlib.pyplot as plt

def train(X1,xout1,T1,l):
    phi = []
    phix = []

    for i in range(7):
        phi1 = 1
        phi2 = math.exp(-10*(X1[i]-0.5)**2)
        phi3 = math.exp(-10*(X1[i]+0.5)**2)
        phi.append([phi1,phi2,phi3])

    for i in range(7):
        phi1 = 1
        phi2 = math.exp(-10*(xout[i]-0.5)**2)
        phi3 = math.exp(-10*(xout[i]+0.5)**2)
        phix.append([phi1,phi2,phi3])


    phi = np.array(phi)
    phix = np.array(phix)

    phixT = phix.T
    phiT = phi.T


    phiphiT = phiT.dot(phi)
    phixphixT = phixT.dot(phix)

    lbda = l

    reg = np.identity(3)
    reg = lbda*reg

    u = phiphiT + reg

    v = np.linalg.inv(u)

    b = phi.dot(v)

    b = b.T

    w = b.dot(T1)

    y = phi.dot(w)

    return y,y



X = np.linspace(-1.0,1,21)

xout = np.linspace(-0.95,0.95,21)

T = np.array([5.12,4.97,4.92,4.83,4.90,5.06,5.29,5.34,5.36,5.76,5.99,6.30,6.66,6.70,7.49,7.92,8.48,9.09,9.7,10.3,10.98])

truearray = np.array([5,4.92,4.88,4.88,4.92,5,5.12,5.28,5.48,5.72,6,6.32,6.68,7.08,7.52,8,8.52,9.08,9.68,10.32,11])

tmean = np.mean(truearray)


X1 = []
xout1 = []
X2 = []
xout2 = []
X3 = []
xout3 = []
T1 = []
T2 = []
T3 = []

i = 0

while i<21:
    X1.append(X[i])
    xout1.append(xout[i])
    T1.append(T[i])
    X2.append(X[i+1])
    xout2.append(xout[i+1])
    T2.append(T[i+1])
    X3.append(X[i+2])
    xout3.append(xout[i+2])
    T3.append(T[i+2])
    i += 3

X1 = np.array(X1)
T1 = np.array(T1)
X2 = np.array(X2)
T2 = np.array(T2)
X3 = np.array(X3)
T3 = np.array(T2)
xout1 = np.array(xout1)
xout2 = np.array(xout2)
xout3 = np.array(xout3)
yr = []
xd = []
xr = []

xr = np.array([])
xd = np.array([])
yr = np.array([])

X1 = np.append(X1,X2)
X1 = np.append(X1,X3)
xd = np.append(xd,X1)

Ylist = []

for i in range(10):
    Y1,yr1 = train(X1,xout1,T1,i)
    Y2,yr2 = train(X2,xout2,T2,i)
    Y3,yr3 = train(X3,xout3,T3,i)
    Ylist.append(Y2)
    #Y1 = np.append(Y1,Y2)
    #Y1 = np.append(Y1,Y3)
    #yr = np.append(yr,Y1)
    #xr = np.append(xr,xd)

sc = plt.scatter(X,T)
for i in range(10):
    plt.plot(xout1,Ylist[i])

plt.show()
```



You can also get this tutorial and run it on your laptop:

```
git clone https://github.com/ipython/ipython-in-depth
```

Install IPython and Jupyter:

with conda (https://www.anaconda.com/download):

```
conda install ipython jupyter
```

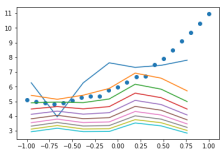
with pip:

```
# first, always upgrade pip!
pip install --upgrade pip
pip install --upgrade ipython jupyter
```

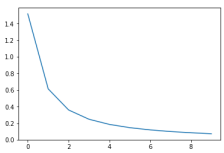
Start the notebook in the tutorial directory:

```
cd ipython-in-depth
jupyter notebook
```

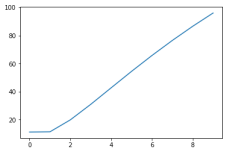
In [20]:
```
%matplotlib inline
import numpy as np
import math
import matplotlib.pyplot as plt

def train(X1,xout1,T1,l):
    phi = []
    phix = []

    for i in range(7):
        phi1 = 1
        phi2 = math.exp(-10*(X1[i]-0.5)**2)
        phi3 = math.exp(-10*(X1[i]+0.5)**2)
        phi.append([phi1,phi2,phi3])

    for i in range(7):
        phi1 = 1
        phi2 = math.exp(-10*(xout[i]-0.5)**2)
        phi3 = math.exp(-10*(xout[i]+0.5)**2)
        phix.append([phi1,phi2,phi3])


    phi = np.array(phi)
    phix = np.array(phix)

    phixT = phix.T
    phiT = phi.T


    phiphiT = phiT.dot(phi)
    phixphixT = phixT.dot(phix)

    lbda = l

    reg = np.identity(3)
    reg = lbda*reg

    u = phiphiT + reg

    v = np.linalg.inv(u)

    b = phi.dot(v)

    b = b.T

    w = b.dot(T1)

    y = phi.dot(w)

    return y,y



X = np.linspace(-1.0,1,21)

xout = np.linspace(-0.95,0.95,21)

T = np.array([5.12,4.97,4.92,4.83,4.90,5.06,5.29,5.34,5.36,5.76,5.99,6.30,6.66,6.70,7.49,7.92,8.48,9.09,9.7,10.3,10.98])

truearray = np.array([5,4.92,4.88,4.88,4.92,5,5.12,5.28,5.48,5.72,6,6.32,6.68,7.08,7.52,8,8.52,9.08,9.68,10.32,11])

tmean = np.mean(truearray)


X1 = []
xout1 = []
X2 = []
xout2 = []
X3 = []
xout3 = []
T1 = []
T2 = []
T3 = []

i = 0

while i<21:
    X1.append(X[i])
    xout1.append(xout[i])
    T1.append(T[i])
    X2.append(X[i+1])
    xout2.append(xout[i+1])
    T2.append(T[i+1])
    X3.append(X[i+2])
    xout3.append(xout[i+2])
    T3.append(T[i+2])
    i += 3

X1 = np.array(X1)
T1 = np.array(T1)
X2 = np.array(X2)
T2 = np.array(T2)
X3 = np.array(X3)
T3 = np.array(T2)
xout1 = np.array(xout1)
xout2 = np.array(xout2)
xout3 = np.array(xout3)
yr = []
xd = []
xr = []

xr = np.array([])
xd = np.array([])
yr = np.array([])

X1 = np.append(X1,X2)
X1 = np.append(X1,X3)
xd = np.append(xd,X1)

Ylist = []

for i in range(10):
    Y1,yr1 = train(X1,xout1,T1,i)
    Y2,yr2 = train(X2,xout2,T2,i)
    Y3,yr3 = train(X3,xout3,T3,i)
    Ylist.append(Y3)
    #Y1 = np.append(Y1,Y2)
    #Y1 = np.append(Y1,Y3)
    #yr = np.append(yr,Y1)
    #xr = np.append(xr,xd)

sc = plt.scatter(X,T)
for i in range(10):
    plt.plot(xout1,Ylist[i])


    plt.show()
```



In [26]:
```
# variance with lambda
import numpy as np
import math
import matplotlib.pyplot as plt

def train(X1,xout1,T1,l):
    phi = []
    phix = []

    for i in range(7):
        phi1 = 1
        phi2 = math.exp(-10*(X1[i]-0.5)**2)
        phi3 = math.exp(-10*(X1[i]+0.5)**2)
        phi.append([phi1,phi2,phi3])

    for i in range(7):
        phi1 = 1
        phi2 = math.exp(-10*(xout[i]-0.5)**2)
        phi3 = math.exp(-10*(xout[i]+0.5)**2)
        phix.append([phi1,phi2,phi3])


    phi = np.array(phi)
    phix = np.array(phix)

    phixT = phix.T
    phiT = phi.T


    phiphiT = phiT.dot(phi)
    phixphixT = phixT.dot(phix)

    lbda = l

    reg = np.identity(3)
    reg = lbda*reg

    u = phiphiT + reg

    v = np.linalg.inv(u)

    b = phi.dot(v)

    b = b.T

    w = b.dot(T1)

    y = phi.dot(w)

    return y,y



X = np.linspace(-1.0,1,21)

xout = np.linspace(-0.95,0.95,21)

T = np.array([5.12,4.97,4.92,4.83,4.90,5.06,5.29,5.34,5.36,5.76,5.99,6.30,6.66,6.70,7.49,7.92,8.48,9.09,9.7,10.3,10.98])

truearray = np.array([5,4.92,4.88,4.88,4.92,5,5.12,5.28,5.48,5.72,6,6.32,6.68,7.08,7.52,8,8.52,9.08,9.68,10.32,11])

tmean = np.mean(truearray)


X1 = []
xout1 = []
X2 = []
xout2 = []
X3 = []
xout3 = []
T1 = []
T2 = []
T3 = []

i = 0

while i<21:
    X1.append(X[i])
    xout1.append(xout[i])
    T1.append(T[i])
    X2.append(X[i+1])
    xout2.append(xout[i+1])
    T2.append(T[i+1])
    X3.append(X[i+2])
    xout3.append(xout[i+2])
    T3.append(T[i+2])
    i += 3

X1 = np.array(X1)
T1 = np.array(T1)
X2 = np.array(X2)
T2 = np.array(T2)
X3 = np.array(X3)
T3 = np.array(T2)
xout1 = np.array(xout1)
xout2 = np.array(xout2)
xout3 = np.array(xout3)
yr = []
xd = []
xr = []

xr = np.array([])
xd = np.array([])
yr = np.array([])

X1 = np.append(X1,X2)
X1 = np.append(X1,X3)
xd = np.append(xd,X1)

Ylist = []
var = []

for i in range(10):
    Y1,yr1 = train(X1,xout1,T1,i)
    var.append(np.var(Y1))
    Y2,yr2 = train(X2,xout2,T2,i)
    Y3,yr3 = train(X3,xout3,T3,i)
    Ylist.append(Y3)
    #Y1 = np.append(Y1,Y2)
    #Y1 = np.append(Y1,Y3)
    #yr = np.append(yr,Y1)
    #xr = np.append(xr,xd)

plt.plot(range(10),var)

plt.show()
```



In [29]:
```
# bias square with lambda
import numpy as np
import math
import matplotlib.pyplot as plt

def train(X1,xout1,T1,l):
    phi = []
    phix = []

    for i in range(7):
        phi1 = 1
        phi2 = math.exp(-10*(X1[i]-0.5)**2)
        phi3 = math.exp(-10*(X1[i]+0.5)**2)
        phi.append([phi1,phi2,phi3])

    for i in range(7):
        phi1 = 1
        phi2 = math.exp(-10*(xout[i]-0.5)**2)
        phi3 = math.exp(-10*(xout[i]+0.5)**2)
        phix.append([phi1,phi2,phi3])


    phi = np.array(phi)
    phix = np.array(phix)

    phixT = phix.T
    phiT = phi.T


    phiphiT = phiT.dot(phi)
    phixphixT = phixT.dot(phix)

    lbda = l

    reg = np.identity(3)
    reg = lbda*reg

    u = phiphiT + reg

    v = np.linalg.inv(u)

    b = phi.dot(v)

    b = b.T

    w = b.dot(T1)

    y = phi.dot(w)

    return y,y



X = np.linspace(-1.0,1,21)

xout = np.linspace(-0.95,0.95,21)

T = np.array([5.12,4.97,4.92,4.83,4.90,5.06,5.29,5.34,5.36,5.76,5.99,6.30,6.66,6.70,7.49,7.92,8.48,9.09,9.7,10.3,10.98])

truearray = np.array([5,4.92,4.88,4.88,4.92,5,5.12,5.28,5.48,5.72,6,6.32,6.68,7.08,7.52,8,8.52,9.08,9.68,10.32,11])

tmean = np.mean(truearray)


X1 = []
xout1 = []
X2 = []
xout2 = []
X3 = []
xout3 = []
T1 = []
T2 = []
T3 = []

i = 0

while i<21:
    X1.append(X[i])
    xout1.append(xout[i])
    T1.append(T[i])
    X2.append(X[i+1])
    xout2.append(xout[i+1])
    T2.append(T[i+1])
    X3.append(X[i+2])
    xout3.append(xout[i+2])
    T3.append(T[i+2])
    i += 3

X1 = np.array(X1)
T1 = np.array(T1)
X2 = np.array(X2)
T2 = np.array(T2)
X3 = np.array(X3)
T3 = np.array(T2)
xout1 = np.array(xout1)
xout2 = np.array(xout2)
xout3 = np.array(xout3)
yr = []
xd = []
xr = []

xr = np.array([])
xd = np.array([])
yr = np.array([])

X1 = np.append(X1,X2)
X1 = np.append(X1,X3)
xd = np.append(xd,X1)

Ylist = []
bias2 = []

for i in range(10):
    Y1,yr1 = train(X1,xout1,T1,i)
    summ = 0
    for j in range(Y1.size):
        summ += (tmean - Y1[j])**2
    bias2.append(summ)
    Y2,yr2 = train(X2,xout2,T2,i)
    Y3,yr3 = train(X3,xout3,T3,i)
    Ylist.append(Y3)
    #Y1 = np.append(Y1,Y2)
    #Y1 = np.append(Y1,Y3)
    #yr = np.append(yr,Y1)
    #xr = np.append(xr,xd)

plt.plot(range(10),bias2)

plt.show()
```



In [30]:
```
#kernel trick
%matplotlib inline
import numpy as np
import math as m
import matplotlib.pyplot as plt

def kernelf(x1,x2,ss):
    y = m.exp((-(x1-x2)**2.0)/ss)
    return y



Xi = [-1,-0.7,-0.4,-0.1,0.2,0.5,0.8]
Vi = [-0.9,-0.6,-0.3,0,0.3,0.6,0.9]
testi = [-0.8,-0.5,-0.2,0.1,0.4,0.7,1]
traino = [5.12,4.83,5.29,5.76,6.66,6.70,8.48,10.30]
Vo = [4.97,4.9,5.34,5.99,6.7,8.48,10.30]
testo = [4.92,5.06,5.36,6.30,7.49,9.09,10.98]

Xi =np.array(Xi)
Vi =np.array(Vi)
testi =np.array(testi)
traino =np.array(traino)
Vo =np.array(Vo)
testo =np.array(testo)
ss = 0.1

Yi =[]
for j in range(Vi.size):
    M = []
    for i in range(Xi.size):
        mo = kernelf(Xi[i],Vi[j],ss)
        M.append(mo)
    print(M)
    mm = np.sum(M)
    M = M/mm
    print(M)
    Y = M.dot(testo)
    Yi.append(Y)

Yi = np.array(Yi)

dif = Vo-Yi

SSE = dif.dot(dif)
print(SSE)

O = []
Yii = []

SS = np.array([0.2,0.3,0.4,0.5])
sse  = []
for k in range(SS.size):
    Yii = []
    for j in range(Vi.size):
        M = []
        for i in range(Xi.size):
            mo = kernelf(Xi[i],Vi[j],SS[k])
            M.append(mo)
        mm = np.sum(M)
        M = M/mm
        Y = M.dot(testo)
        Yii.append(Y)
    O = np.array(Yii)

    dif = Vo-O

    SSE = dif.dot(dif)

    sse.append(SSE)

print('sse array')
print(sse)


Yi =[]
for j in range(Vi.size):
    M = []
    for i in range(Xi.size):
        mo = kernelf(Xi[i],Vi[j],SS[0])
        M.append(mo)
    mm = np.sum(M)
    M = M/mm
    Y = M.dot(testo)
    Yi.append(Y)

print('training out')
print(testo)
print('predicted out')
print(Yi)

Yi = np.array(Yi)

dif = Vo-Yi

SSE = dif.dot(dif)
print('minimum sse')
print(SSE)
```

```
[0.9048374180359596, 0.6703200460356391, 0.0820849986238988, 0.0016615572731739324, 5.559513241650137e-06, 3.0748798795866172e-09, 2.8111852987890247e-13]
[5.45441070e-01 4.04072683e-01 4.94812975e-02 1.00159604e-03
 3.35130576e-06 1.85355484e-09 1.69459826e-13]
[0.20189651799465536, 0.9048374180359596, 0.6703200460356394, 0.0820849986238988, 0.0016615572731739324, 5.559513241650137e-06, 3.0748798795866172e-09]
[1.08499493e-01 4.86260991e-01 3.60231002e-01 4.41126018e-02
 8.92923380e-04 2.98769079e-06 1.65244508e-09]
[0.0074465830709244344, 0.20189651799465547, 0.9048374180359595, 0.6703200460356393, 0.0820849986238988, 0.0016615572731739324, 5.559513241650137e-06]
[3.98585435e-03 1.08067030e-01 4.84322826e-01 3.58795174e-01
 4.39367755e-02 8.89364319e-04 2.97578229e-06]
[4.5399929762484854e-05, 0.0074465830709244344, 0.20189651799465536, 0.9048374180359595, 0.6703200460356392, 0.0820849986238988, 0.0016615572731739324]
[2.43002256e-05 3.98576935e-03 1.08064725e-01 4.84312498e-01
 3.58787523e-01 4.39358386e-02 8.89345354e-04]
[4.5753387694457955e-08, 4.5399929762484854e-05, 0.0074465830709244344, 0.20189651799465536, 0.9048374180359596, 0.6703200460356392, 0.0820849986238988]
[2.45112116e-08 2.43218555e-05 3.98931714e-03 1.08160915e-01
 8.84743591e-01 3.59106884e-01 4.39749464e-02]
[7.6218651945128636-12, 4.5753387694458286-08, 4.5399929762484854e-05, 0.0074465830709244344, 0.20189651799465547, 0.9048374180359596, 0.6703200460356391]
[4.27103877e-12 2.56386708e-08 2.54406048e-05 4.17281652e-03
 1.13136068e-01 5.07040677e-01 3.75624972e-01]
[2.0987910487930646e-16, 7.6218651945128636-12, 4.5753387694457955e-08, 4.5399929762484854e-05, 0.0074465830709244344, 0.20189651799465536, 0.9048374180359596]
[1.08363143e-16 6.84050223e-12 4.10629344e-08 4.07457115e-05
 6.68318932e-03 1.81198899e-01 8.12077125e-01]
4.122188854805864
sse array
[3.883489419720259, 3.941810093404, 4.214118431821516, 4.642343457295543]
training out
[ 4.92  5.06  5.36  6.3   7.49  9.09 10.98]
predicted out
[5.064420120346568, 5.313773826327173, 5.880789507396874, 6.8401349106827505, 8.095778127421047, 9.332897527385729, 10.187964066653803]
minimum sse
3.883489419720259
```

In [ ]: