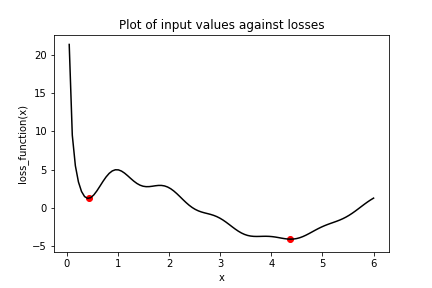
**The dangers of local minima**

Consider the plot of the following loss function, loss\_function(), which contains a global minimum, marked by the dot on the right, and several local minima, including the one marked by the dot on the left.



In this exercise, you will try to find the global minimum of loss\_function()using keras.optimizers.SGD(). You will do this twice, each time with a different initial value of the input to loss\_function(). First, you will use x\_1, which is a variable with an initial value of 6.0. Second, you will use x\_2, which is a variable with an initial value of 0.3. Note that loss\_function() has been defined and is available.

**Instructions**

**100 XP**

* Set opt to use the stochastic gradient descent optimizer (SGD) with a learning rate of 0.01.
* Perform minimization using the loss function, loss\_function(), and the variable with an initial value of 6.0, x\_1.
* Perform minimization using the loss function, loss\_function(), and the variable with an initial value of 0.3, x\_2.
* Print x\_1 and x\_2 as numpy arrays and check whether the values differ. These are the minima that the algorithm identified.

###### Hint

* Use the keras.optimizers.SGD(learning\_rate=r) operation, where r is the learning rate.
* Pass x\_1 as the only argument to the loss function and also to the variables list.
* Remember to pass loss\_function(x\_2) to the minimize operation.
* Apply .numpy() to x\_1 and x\_2 and pass them to the print()function.

# Initialize x\_1 and x\_2

x\_1 = Variable(6.0,float32)

x\_2 = Variable(0.3,float32)

# Define the optimization operation

opt = keras.optimizers.SGD(learning\_rate=0.01)

for j in range(100):

# Perform minimization using the loss function and x\_1

opt.minimize(lambda: loss\_function(x\_1), var\_list=[x\_1])

# Perform minimization using the loss function and x\_2

opt.minimize(lambda: loss\_function(x\_2), var\_list=[x\_2])

# Print x\_1 and x\_2 as numpy arrays

print(x\_1.numpy(), x\_2.numpy())

Great work! Notice that we used the same optimizer and loss function, but two different initial values. When we started at 6.0 with x\_1, we found the global minimum at 4.38, marked by the dot on the right. When we started at 0.3, we stopped around 0.42 with x\_2, the local minimum marked by a dot on the far left.