**Train a linear model**

In this exercise, we will pick up where the previous exercise ended. The intercept and slope, intercept and slope, have been defined and initialized. Additionally, a function has been defined, loss\_function(intercept, slope), which computes the loss using the data and model variables.

You will now define an optimization operation as opt. You will then train a univariate linear model by minimizing the loss to find the optimal values of intercept and slope. Note that the opt operation will try to move closer to the optimum with each step, but will require many steps to find it. Thus, you must repeatedly execute the operation.

**Instructions**

**100 XP**

* Initialize an Adam optimizer as opt with a learning rate of 0.5.
* Apply the .minimize() method to the optimizer.
* Pass loss\_function() with the appropriate arguments as a lambda function to .minimize().
* Supply the list of variables that need to be updated to var\_list.

# Initialize an adam optimizer

opt = keras.optimizers.Adam(0.5)

for j in range(100):

# Apply minimize, pass the loss function, and supply the variables

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

# Print every 10th value of the loss

if j % 10 == 0:

print(loss\_function(intercept, slope).numpy())

# Plot data and regression line

plot\_results(intercept, slope)

Excellent! Notice that we printed loss\_function(intercept, slope) every 10th execution for 100 executions. Each time, the loss got closer to the minimum as the optimizer moved the slope and intercept parameters closer to their optimal values.