**Training a linear model in batches**

In this exercise, we will train a linear regression model in batches, starting where we left off in the previous exercise. We will do this by stepping through the dataset in batches and updating the model's variables, intercept and slope, after each step. This approach will allow us to train with datasets that are otherwise too large to hold in memory.

Note that the loss function,loss\_function(intercept, slope, targets, features), has been defined for you. Additionally, keras has been imported for you and numpyis available as np. The trainable variables should be entered into var\_list in the order in which they appear as loss function arguments.

**Instructions**

**100 XP**

* Use the .Adam() optimizer.
* Load in the data from 'kc\_house\_data.csv' in batches with a chunksize of 100.
* Extract the price column from batch, convert it to a numpy array of type 32-bit float, and assign it to price\_batch.
* Complete the loss function, fill in the list of trainable variables, and perform minimization.

# Initialize adam optimizer

opt = keras.optimizers.Adam()

# Load data in batches

for batch in pd.read\_csv('kc\_house\_data.csv', chunksize=100):

size\_batch = np.array(batch['sqft\_lot'], np.float32)

# Extract the price values for the current batch

price\_batch = np.array(batch['price'], np.float32)

# Complete the loss, fill in the variable list, and minimize

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

# Print trained parameters

print(intercept.numpy(), slope.numpy())

Great work! Batch training will be very useful when you train neural networks, which we will do next.