def get\_mae(max\_leaf\_nodes, train\_X, val\_X, train\_y, val\_y):

model = DecisionTreeRegressor(max\_leaf\_nodes=max\_leaf\_nodes, random\_state=0)

model.fit(train\_X, train\_y)

preds\_val = model.predict(val\_X)

mae = mean\_absolute\_error(val\_y, preds\_val)

return(mae)

**### My option**

candidate\_max\_leaf\_nodes = [5, 25, 50, 100, 250, 500]

# Write loop to find the ideal tree size from candidate\_max\_leaf\_nodes

min\_error=val\_mae # previously defined MAE

for leaf\_size in candidate\_max\_leaf\_nodes:

error=get\_mae(leaf\_size, train\_X, val\_X, train\_y, val\_y)

if error <= min\_error:

min\_error= error

min\_error\_leaf\_nodes= leaf\_size

# Store the best value of max\_leaf\_nodes (it will be either 5, 25, 50, 100, 250 or 500)

best\_tree\_size = min\_error\_leaf\_nodes

**### Short Solution**

# **Here is a short solution with a dict comprehension.**

# The lesson gives an example of how to do this with an explicit loop.

scores **=** {leaf\_size: get\_mae(leaf\_size, train\_X, val\_X, train\_y, val\_y) **for** leaf\_size **in** candidate\_max\_leaf\_nodes}

best\_tree\_size **=** min(scores, key**=**scores.get)