This notebook is an exercise in the Intro to Deep Learning course. You can reference the tutorial at this link.

Introduction

In this exercise you'll train a neural network on the *Fuel Economy* dataset and then explore the effect of the learning rate and batch size on SGD.

When you're ready, run this next cell to set everything up!

In [1]:

```
# Setup plotting
import matplotlib.pyplot as plt
from learntools.deep_learning_intro.dltools import animate_sgd
plt.style.use('seaborn-whitegrid')
# Set Matplotlib defaults
plt.rc('figure', autolayout=True)
plt.rc('axes', labelweight='bold', labelsize='large',
       titleweight='bold', titlesize=18, titlepad=10)
plt.rc('animation', html='html5')

# Setup feedback system
from learntools.core import binder
binder.bind(globals())
from learntools.deep_learning_intro.ex3 import *
```

In the *Fuel Economy* dataset your task is to predict the fuel economy of an automobile given features like its type of engine or the year it was made.

First load the dataset by running the cell below.

In [2]:

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import make_column_transformer, make_column_selector
from sklearn.model_selection import train_test_split

fuel = pd.read_csv('../input/dl-course-data/fuel.csv')

X = fuel.copy()
# Remove target
y = X.pop('FE')

preprocessor = make_column_transformer(
    (StandardScaler(),
     make_column_selector(dtype_include=np.number)),
    (OneHotEncoder(sparse=False),
     make_column_selector(dtype_include=object)),
)

X = preprocessor.fit_transform(X)
y = np.log(y) # log transform target instead of standardizing
```

```
input_shape = [X.shape[1]]
print("Input shape: {}".format(input_shape))
```

```
Input shape: [50]
```

Take a look at the data if you like. Our target in this case is the `'FE'` column and the remaining columns are the features.

In [3]:
```
# Uncomment to see original data
fuel.head()
# Uncomment to see processed features
pd.DataFrame(X[:10,:]).head()
```

Out[3]:

	0	1	2	3	4	5	6	7	8	9	...	40
0	0.913643	1.068005	0.524148	0.685653	-0.226455	0.391659	0.43492	0.463841	-0.447941	0.0	...	0.0
1	0.913643	1.068005	0.524148	0.685653	-0.226455	0.391659	0.43492	0.463841	-0.447941	0.0	...	0.0
2	0.530594	1.068005	0.524148	0.685653	-0.226455	0.391659	0.43492	0.463841	-0.447941	0.0	...	0.0
3	0.530594	1.068005	0.524148	0.685653	-0.226455	0.391659	0.43492	0.463841	-0.447941	0.0	...	0.0
4	1.296693	2.120794	0.524148	-1.458464	-0.226455	0.391659	0.43492	0.463841	-0.447941	0.0	...	0.0

5 rows × 50 columns

Run the next cell to define the network we'll use for this task.

In [4]:
```
from tensorflow import keras
from tensorflow.keras import layers

model = keras.Sequential([
    layers.Dense(128, activation='relu', input_shape=input_shape),
    layers.Dense(128, activation='relu'),
    layers.Dense(64, activation='relu'),
    layers.Dense(1),
])
```

```
2022-12-18 05:28:01.062056: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node zero
2022-12-18 05:28:01.155454: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node zero
2022-12-18 05:28:01.156256: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node zero
2022-12-18 05:28:01.158013: I tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical operations:  AVX2 AVX512F FMA
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
2022-12-18 05:28:01.158339: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node zero
2022-12-18 05:28:01.159068: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937]
```

```
successful NUMA node read from SysFS had negative value (-1), but there must be at least
one NUMA node, so returning NUMA node zero
2022-12-18 05:28:01.159720: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937]
successful NUMA node read from SysFS had negative value (-1), but there must be at least
one NUMA node, so returning NUMA node zero
2022-12-18 05:28:03.398214: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937]
successful NUMA node read from SysFS had negative value (-1), but there must be at least
one NUMA node, so returning NUMA node zero
2022-12-18 05:28:03.399177: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937]
successful NUMA node read from SysFS had negative value (-1), but there must be at least
one NUMA node, so returning NUMA node zero
2022-12-18 05:28:03.399860: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937]
successful NUMA node read from SysFS had negative value (-1), but there must be at least
one NUMA node, so returning NUMA node zero
2022-12-18 05:28:03.400465: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1510] Cre
ated device /job:localhost/replica:0/task:0/device:GPU:0 with 15401 MB memory:  -> devic
e: 0, name: Tesla P100-PCIE-16GB, pci bus id: 0000:00:04.0, compute capability: 6.0
```

1) Add Loss and Optimizer

Before training the network we need to define the loss and optimizer we'll use. Using the model's `compile` method, add the Adam optimizer and MAE loss.

In [5]:

```
# YOUR CODE HERE
model.compile(optimizer='adam', loss='mae')

# Check your answer
q_1.check()
```

Correct

In [6]:

```
# Lines below will give you a hint or solution code
#q_1.hint()
#q_1.solution()
```

2) Train Model

Once you've defined the model and compiled it with a loss and optimizer you're ready for training. Train the network for 200 epochs with a batch size of 128. The input data is `X` with target `y`.

In [7]:

```
# YOUR CODE HERE
history = model.fit(
    X, y,
    batch_size=128,
    epochs=200,
)

# Check your answer
q_2.check()
```

```
2022-12-18 05:28:03.975906: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:1
85] None of the MLIR Optimization Passes are enabled (registered 2)
Epoch 1/200
9/9 [==============================] - 1s 2ms/step - loss: 2.8497
Epoch 2/200
9/9 [==============================] - 0s 2ms/step - loss: 1.0535
Epoch 3/200
9/9 [==============================] - 0s 2ms/step - loss: 0.5483
Epoch 4/200
9/9 [==============================] - 0s 2ms/step - loss: 0.3417
Epoch 5/200
9/9 [==============================] - 0s 2ms/step - loss: 0.2094
Epoch 6/200
9/9 [==============================] - 0s 2ms/step - loss: 0.1623
Epoch 7/200
9/9 [==============================] - 0s 2ms/step - loss: 0.1323
Epoch 8/200
9/9 [==============================] - 0s 2ms/step - loss: 0.1027
Epoch 9/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0922
Epoch 10/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0856
Epoch 11/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0753
Epoch 12/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0714
Epoch 13/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0680
Epoch 14/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0663
Epoch 15/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0675
Epoch 16/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0610
Epoch 17/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0612
Epoch 18/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0598
Epoch 19/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0610
Epoch 20/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0586
Epoch 21/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0520
Epoch 22/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0494
Epoch 23/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0509
Epoch 24/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0470
Epoch 25/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0536
Epoch 26/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0521
Epoch 27/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0468
Epoch 28/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0464
Epoch 29/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0432
```

```
Epoch 30/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0441
Epoch 31/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0467
Epoch 32/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0491
Epoch 33/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0472
Epoch 34/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0420
Epoch 35/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0390
Epoch 36/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0399
Epoch 37/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0409
Epoch 38/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0414
Epoch 39/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0423
Epoch 40/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0400
Epoch 41/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0373
Epoch 42/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0387
Epoch 43/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0377
Epoch 44/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0364
Epoch 45/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0368
Epoch 46/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0376
Epoch 47/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0367
Epoch 48/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0376
Epoch 49/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0428
Epoch 50/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0453
Epoch 51/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0413
Epoch 52/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0349
Epoch 53/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0353
Epoch 54/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0366
Epoch 55/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0381
Epoch 56/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0382
Epoch 57/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0355
Epoch 58/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0332
Epoch 59/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0364
```

```
Epoch 60/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0344
Epoch 61/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0373
Epoch 62/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0359
Epoch 63/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0357
Epoch 64/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0344
Epoch 65/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0360
Epoch 66/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0369
Epoch 67/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0482
Epoch 68/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0489
Epoch 69/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0366
Epoch 70/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0342
Epoch 71/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0324
Epoch 72/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0328
Epoch 73/200
9/9 [==============================] - 0s 5ms/step - loss: 0.0321
Epoch 74/200
9/9 [==============================] - 0s 5ms/step - loss: 0.0336
Epoch 75/200
9/9 [==============================] - 0s 4ms/step - loss: 0.0332
Epoch 76/200
9/9 [==============================] - 0s 4ms/step - loss: 0.0324
Epoch 77/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0291
Epoch 78/200
9/9 [==============================] - 0s 3ms/step - loss: 0.0338
Epoch 79/200
9/9 [==============================] - 0s 3ms/step - loss: 0.0357
Epoch 80/200
9/9 [==============================] - 0s 3ms/step - loss: 0.0317
Epoch 81/200
9/9 [==============================] - 0s 3ms/step - loss: 0.0315
Epoch 82/200
9/9 [==============================] - 0s 3ms/step - loss: 0.0290
Epoch 83/200
9/9 [==============================] - 0s 3ms/step - loss: 0.0313
Epoch 84/200
9/9 [==============================] - 0s 3ms/step - loss: 0.0291
Epoch 85/200
9/9 [==============================] - 0s 3ms/step - loss: 0.0328
Epoch 86/200
9/9 [==============================] - 0s 3ms/step - loss: 0.0343
Epoch 87/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0302
Epoch 88/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0310
Epoch 89/200
9/9 [==============================] - 0s 3ms/step - loss: 0.0293
```

```
Epoch 90/200
9/9 [==============================] - 0s 3ms/step - loss: 0.0283
Epoch 91/200
9/9 [==============================] - 0s 3ms/step - loss: 0.0306
Epoch 92/200
9/9 [==============================] - 0s 3ms/step - loss: 0.0355
Epoch 93/200
9/9 [==============================] - 0s 4ms/step - loss: 0.0460
Epoch 94/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0369
Epoch 95/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0308
Epoch 96/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0292
Epoch 97/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0320
Epoch 98/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0338
Epoch 99/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0376
Epoch 100/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0302
Epoch 101/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0318
Epoch 102/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0329
Epoch 103/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0304
Epoch 104/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0374
Epoch 105/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0417
Epoch 106/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0456
Epoch 107/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0453
Epoch 108/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0372
Epoch 109/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0352
Epoch 110/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0377
Epoch 111/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0380
Epoch 112/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0362
Epoch 113/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0375
Epoch 114/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0349
Epoch 115/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0326
Epoch 116/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0324
Epoch 117/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0312
Epoch 118/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0340
Epoch 119/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0338
```

```
Epoch 120/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0308
Epoch 121/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0306
Epoch 122/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0357
Epoch 123/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0381
Epoch 124/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0323
Epoch 125/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0388
Epoch 126/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0320
Epoch 127/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0371
Epoch 128/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0304
Epoch 129/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0296
Epoch 130/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0304
Epoch 131/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0318
Epoch 132/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0410
Epoch 133/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0370
Epoch 134/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0307
Epoch 135/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0290
Epoch 136/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0286
Epoch 137/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0411
Epoch 138/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0356
Epoch 139/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0292
Epoch 140/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0300
Epoch 141/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0344
Epoch 142/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0296
Epoch 143/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0299
Epoch 144/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0288
Epoch 145/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0287
Epoch 146/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0279
Epoch 147/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0322
Epoch 148/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0336
Epoch 149/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0352
```

```
Epoch 150/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0286
Epoch 151/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0332
Epoch 152/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0347
Epoch 153/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0375
Epoch 154/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0336
Epoch 155/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0323
Epoch 156/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0428
Epoch 157/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0384
Epoch 158/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0388
Epoch 159/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0395
Epoch 160/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0379
Epoch 161/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0348
Epoch 162/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0304
Epoch 163/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0272
Epoch 164/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0264
Epoch 165/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0304
Epoch 166/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0281
Epoch 167/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0295
Epoch 168/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0295
Epoch 169/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0288
Epoch 170/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0313
Epoch 171/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0313
Epoch 172/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0281
Epoch 173/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0315
Epoch 174/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0322
Epoch 175/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0303
Epoch 176/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0302
Epoch 177/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0313
Epoch 178/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0299
Epoch 179/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0334
```

```
Epoch 180/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0326
Epoch 181/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0276
Epoch 182/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0267
Epoch 183/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0251
Epoch 184/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0263
Epoch 185/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0288
Epoch 186/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0387
Epoch 187/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0330
Epoch 188/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0301
Epoch 189/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0287
Epoch 190/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0311
Epoch 191/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0308
Epoch 192/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0326
Epoch 193/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0281
Epoch 194/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0279
Epoch 195/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0278
Epoch 196/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0281
Epoch 197/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0279
Epoch 198/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0267
Epoch 199/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0260
Epoch 200/200
9/9 [==============================] - 0s 2ms/step - loss: 0.0259
```

Correct

In [8]:

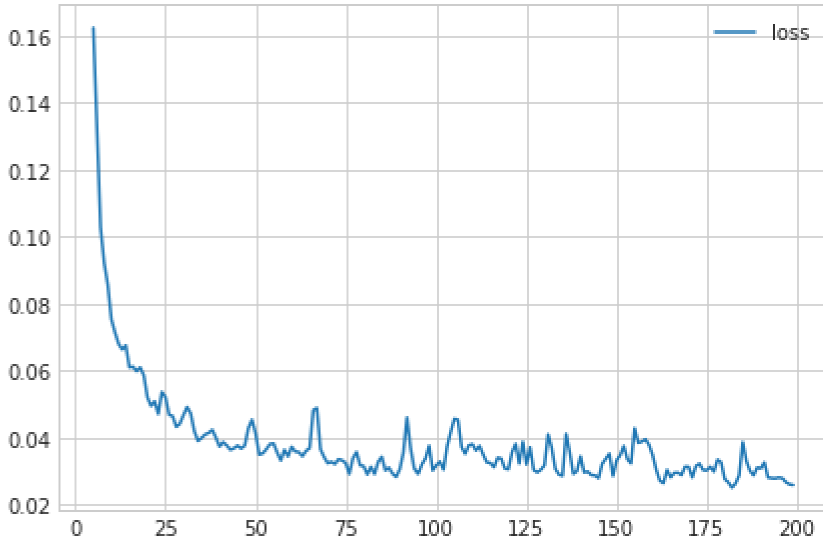
```
# Lines below will give you a hint or solution code
#q_2.hint()
#q_2.solution()
```

The last step is to look at the loss curves and evaluate the training. Run the cell below to get a plot of the training loss.

In [9]:

```
import pandas as pd

history_df = pd.DataFrame(history.history)
# Start the plot at epoch 5. You can change this to get a different view.
history_df.loc[5:, ['loss']].plot();
```



3) Evaluate Training

If you trained the model longer, would you expect the loss to decrease further?

In [10]:
```
# View the solution (Run this cell to receive credit!)
q_3.check()
```

Correct:

This depends on how the loss has evolved during training: if the learning curves have levelled off, there won't usually be any advantage to training for additional epochs. Conversely, if the loss appears to still be decreasing, then training for longer could be advantageous.

With the learning rate and the batch size, you have some control over:

- How long it takes to train a model
- How noisy the learning curves are
- How small the loss becomes

To get a better understanding of these two parameters, we'll look at the linear model, our ppsimplest neural network. Having only a single weight and a bias, it's easier to see what effect a change of parameter has.

The next cell will generate an animation like the one in the tutorial. Change the values for `learning_rate`, `batch_size`, and `num_examples` (how many data points) and then run the cell. (It may take a moment or two.) Try the following combinations, or try some of your own:

learning_rate	batch_size	num_examples
0.05	32	256
0.05	2	256

learning_rate	batch_size	num_examples
0.05	128	256
0.02	32	256
0.2	32	256
1.0	32	256
0.9	4096	8192
0.99	4096	8192

In [11]:

```
# YOUR CODE HERE: Experiment with different values for the learning rate, batch size, a
learning_rate = 0.05
batch_size = 32
num_examples = 256

animate_sgd(
    learning_rate=learning_rate,
    batch_size=batch_size,
    num_examples=num_examples,
    # You can also change these, if you like
    steps=50, # total training steps (batches seen)
    true_w=3.0, # the slope of the data
    true_b=2.0, # the bias of the data
)
```

Out[11]:

0:00 / 0:04

4) Learning Rate and Batch Size

What effect did changing these parameters have? After you've thought about it, run the cell below for some discussion.

In [12]:

```
# View the solution (Run this cell to receive credit!)
q_4.check()
```

Correct:

You probably saw that smaller batch sizes gave noisier weight updates and loss curves. This is because each batch is a small *sample* of data and smaller samples tend to give noisier estimates. Smaller batches can have an "averaging" effect though which can be beneficial.

Smaller learning rates make the updates smaller and the training takes longer to converge. Large learning rates can speed up training, but don't "settle in" to a minimum as well. When the learning rate is too large, the training can fail completely. (Try setting the learning rate to a large value like 0.99 to see this.)

Keep Going

Learn how to **improve your model's performance** by tuning capacity or adding an early stopping callback.

Have questions or comments? Visit the course discussion forum to chat with other learners.