```
#hide
! [ -e /content ] && pip install -Uqq fastbook
import fastbook
fastbook.setup_book()
```

```
          |████████████████████████████████| 719 kB 14.7 MB/s
          |████████████████████████████████| 5.8 MB 61.4 MB/s
          |████████████████████████████████| 452 kB 77.6 MB/s
          |████████████████████████████████| 1.3 MB 57.2 MB/s
          |████████████████████████████████| 1.6 MB 60.4 MB/s
          |████████████████████████████████| 182 kB 70.5 MB/s
          |████████████████████████████████| 132 kB 78.5 MB/s
          |████████████████████████████████| 213 kB 13.2 MB/s
          |████████████████████████████████| 127 kB 61.1 MB/s
          |████████████████████████████████| 7.6 MB 55.6 MB/s
     Mounted at /content/gdrive
```

```
#hide
from fastbook import *
```

Your Deep Learning Journey

Deep Learning Is for Everyone

Neural Networks: A Brief History

Who We Are

How to Learn Deep Learning

Your Projects and Your Mindset

The Software: PyTorch, fastai, and Jupyter

Your First Model

Getting a GPU Deep Learning Server

Running Your First Notebook

```
# CLICK ME
from fastai.vision.all import *
path = untar_data(URLs.PETS)/'images'

def is_cat(x): return x[0].isupper()
dls = ImageDataLoaders.from_name_func(
    path, get_image_files(path), valid_pct=0.2, seed=42,
    label_func=is_cat, item_tfms=Resize(224))

learn = vision_learner(dls, resnet34, metrics=error_rate)
learn.fine_tune(1)
```

```
                                        100.00% [811712512/811706944 00:30<00:00]
/usr/local/lib/python3.8/dist-packages/torchvision/models/_utils.py:208: UserWarni
  warnings.warn(
/usr/local/lib/python3.8/dist-packages/torchvision/models/_utils.py:223: UserWarni
  warnings.warn(msg)
Downloading: "https://download.pytorch.org/models/resnet34-b627a593.pth" to /root/
```

Sidebar: This Book Was Written in Jupyter Notebooks

```
     0      0.171691      0.000000      0.000470   00:52
```

```
1+1
```

```
2
```

```
img = PILImage.create(image_cat())
img.to_thumb(192)
```

End sidebar

```
uploader = widgets.FileUpload()
uploader
```

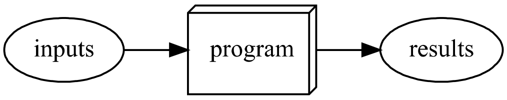
```
#hide
# For the book, we can't actually click an upload button, so we fake it
uploader = SimpleNamespace(data = ['images/chapter1_cat_example.jpg'])
```

```
# Breno: I Upload an image then it's giving this error on the variable img
img = PILImage.create(uploader.data[0])
is_cat,_,probs = learn.predict(img)
print(f"Is this a cat?: {is_cat}.")
print(f"Probability it's a cat: {probs[1].item():.6f}")
```

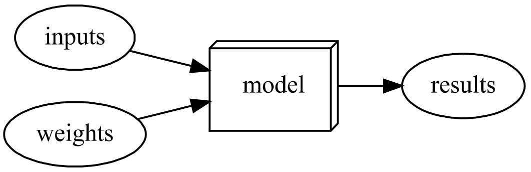
```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
<ipython-input-44-64542688999e> in <module>
      1 # Breno: I Upload an image then it's giving this error on the variable
img
----> 2 img = PILImage.create(uploader.data[0])
      3 is_cat,_,probs = learn.predict(img)
      4 print(f"Is this a cat?: {is_cat}.")
      5 print(f"Probability it's a cat: {probs[1].item():.6f}")

                              ⇕ 2 frames
/usr/local/lib/python3.8/dist-packages/PIL/Image.py in open(fp, mode)
   2841
   2842     if filename:
-> 2843         fp = builtins.open(filename, "rb")
   2844         exclusive_fp = True
   2845

FileNotFoundError: [Errno 2] No such file or directory:
'images/chapter1_cat_example.jpg'
```

What Is Machine Learning?

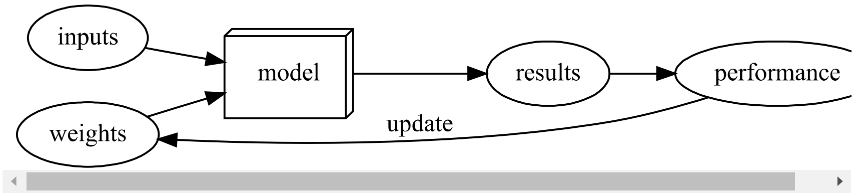
```
gv('''program[shape=box3d width=1 height=0.7]
inputs->program->results''')
```



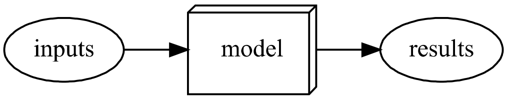
```
gv('''model[shape=box3d width=1 height=0.7]
inputs->model->results; weights->model''')
```



```
gv('''ordering=in
model[shape=box3d width=1 height=0.7]
inputs->model->results; weights->model; results->performance
performance->weights[constraint=false label=update]''')
```



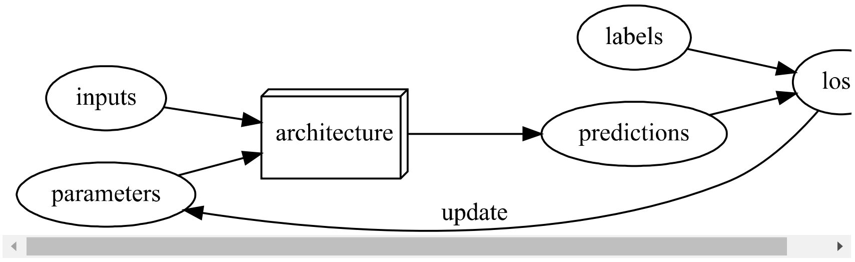
```
gv('''model[shape=box3d width=1 height=0.7]
inputs->model->results''')
```



What Is a Neural Network?

A Bit of Deep Learning Jargon

```
gv('''ordering=in
model[shape=box3d width=1 height=0.7 label=architecture]
inputs->model->predictions; parameters->model; labels->loss; predictions->loss
loss->parameters[constraint=false label=update]''')
```



Limitations Inherent To Machine Learning

From this picture we can now see some fundamental things about training a deep learning model:

- A model cannot be created without data.
- A model can only learn to operate on the patterns seen in the input data used to train it.
- This learning approach only creates *predictions*, not recommended *actions*.
- It's not enough to just have examples of input data; we need *labels* for that data too (e.g., pictures of dogs and cats aren't enough to train a model; we need a label for each one, saying which ones are dogs, and which are cats).

Generally speaking, we've seen that most organizations that say they don't have enough data, actually mean they don't have enough *labeled* data. If any organization is interested in doing something in practice with a model, then presumably they have some inputs they plan to run their model against. And presumably they've been doing that some other way for a while (e.g., manually, or with some heuristic program), so they have data from those processes! For instance, a radiology practice will almost certainly have an archive of medical scans (since they need to be able to check how their patients are progressing over time), but those scans may not have structured labels containing a list of diagnoses or interventions (since radiologists generally create free-text natural language reports, not structured data). We'll be discussing labeling approaches a lot in this book, because it's such an important issue in practice.

Since these kinds of machine learning models can only make *predictions* (i.e., attempt to replicate labels), this can result in a significant gap between organizational goals and model capabilities. For instance, in this book you'll learn how to create a *recommendation system* that can predict what products a user might purchase. This is often used in e-commerce, such as to customize products shown on a home page by showing the highest-ranked items. But such a model is generally created by looking at a user and their buying history (*inputs*) and what they went on to buy or look at (*labels*), which means that the model is likely to tell you about products the user already has or already knows about, rather than new products that they are most likely to be interested in hearing about. That's very different to what, say, an expert at your local bookseller might do, where they ask questions to figure out your taste, and then tell you about authors or series that you've never heard of before.

How Our Image Recognizer Works

What Our Image Recognizer Learned

Image Recognizers Can Tackle Non-Image Tasks

Jargon Recap

Deep Learning Is Not Just for Image Classification

```
path = untar_data(URLs.CAMVID_TINY)
dls = SegmentationDataLoaders.from_label_func(
    path, bs=8, fnames = get_image_files(path/"images"),
    label_func = lambda o: path/'labels'/f'{o.stem}_P{o.suffix}',
    codes = np.loadtxt(path/'codes.txt', dtype=str)
)

learn = unet_learner(dls, resnet34)
learn.fine_tune(8)
```

```
                                        100.18% [2318336/2314212 00:00<00:00]
/usr/local/lib/python3.8/dist-packages/torchvision/models/_utils.py:208: UserWarni
  warnings.warn(
/usr/local/lib/python3.8/dist-packages/torchvision/models/_utils.py:223: UserWarni
  warnings.warn(msg)
epoch  train_loss  valid_loss  time
    0    2.716805    2.421561  00:01
epoch  train_loss  valid_loss  time
    0    1.809804    1.611556  00:01
    1    1.580019    1.348666  00:01
    2    1.460032    1.321212  00:01
    3    1.331450    0.968976  00:01
    4    1.193744    0.885084  00:01
    5    1.072593    0.812920  00:01
    6    0.976319    0.799590  00:01
    7    0.901411    0.785505  00:01
```

```
learn.show_results(max_n=6, figsize=(7,8))
```

Target/Prediction

```
from fastai.text.all import *

dls = TextDataLoaders.from_folder(untar_data(URLs.IMDB), valid='test')
learn = text_classifier_learner(dls, AWD_LSTM, drop_mult=0.5, metrics=accuracy)
learn.fine_tune(4, 1e-2)
```

```
100.00% [144441344/144440600 00:04<00:00]
100.00% [105070592/105067061 00:03<00:00]
```

epoch	train_loss	valid_loss	accuracy	time
0	0.462068	0.393640	0.823960	03:21

epoch	train_loss	valid_loss	accuracy	time
0	0.316752	0.240082	0.905480	07:00
1	0.254804	0.219403	0.920320	07:01
2	0.182490	0.195798	0.928520	07:02
3	0.147036	0.185644	0.931680	07:00

If you hit a "CUDA out of memory error" after running this cell, click on the menu Kernel, then restart. Instead of executing the cell above, copy and paste the following code in it:

```
from fastai.text.all import *

dls = TextDataLoaders.from_folder(untar_data(URLs.IMDB), valid='test', bs=32)
learn = text_classifier_learner(dls, AWD_LSTM, drop_mult=0.5, metrics=accuracy)
learn.fine_tune(4, 1e-2)
```

This reduces the batch size to 32 (we will explain this later). If you keep hitting the same error, change 32 to 16.

```
learn.predict("I really liked that movie!")
```

```
('pos', tensor(1), tensor([0.0053, 0.9947]))
```

Sidebar: The Order Matters

End sidebar

```
from fastai.tabular.all import *
path = untar_data(URLs.ADULT_SAMPLE)

dls = TabularDataLoaders.from_csv(path/'adult.csv', path=path, y_names="salary",
    cat_names = ['workclass', 'education', 'marital-status', 'occupation',
                 'relationship', 'race'],
    cont_names = ['age', 'fnlwgt', 'education-num'],
    procs = [Categorify, FillMissing, Normalize])

learn = tabular_learner(dls, metrics=accuracy)
```

```
100.69% [974848/968212 00:00<00:00]
```

```
learn.fit_one_cycle(3)
```

epoch	train_loss	valid_loss	accuracy	time
0	0.382581	0.360513	0.830927	00:04
1	0.355281	0.351091	0.838606	00:04
2	0.360611	0.347945	0.837991	00:04

```
from fastai.collab import *
path = untar_data(URLs.ML_SAMPLE)
dls = CollabDataLoaders.from_csv(path/'ratings.csv')
learn = collab_learner(dls, y_range=(0.5,5.5))
learn.fine_tune(10)
```

```
110.72% [57344/51790 00:00<00:00]
```

epoch	train_loss	valid_loss	time
0	1.519207	1.413368	00:00

epoch	train_loss	valid_loss	time
0	1.388155	1.359782	00:00
1	1.294653	1.193021	00:00
2	1.049291	0.891161	00:00
3	0.825404	0.737101	00:00
4	0.701211	0.699669	00:00
5	0.646747	0.688687	00:00
6	0.618522	0.683482	00:00
7	0.600537	0.680893	00:00
8	0.609770	0.680110	00:00
9	0.594119	0.679964	00:00

```
learn.show_results()
```

	userId	movieId	rating	rating_pred
0	38.0	95.0	3.5	4.402937
1	53.0	49.0	5.0	4.268655
2	41.0	49.0	4.0	4.011588
3	24.0	1.0	4.0	3.606720
4	72.0	40.0	3.5	2.999226
5	62.0	44.0	3.0	3.019116
6	14.0	18.0	3.0	4.017102
7	27.0	68.0	1.5	3.370472
8	16.0	63.0	4.0	4.526047

Sidebar: Datasets: Food for Models

End sidebar

Validation Sets and Test Sets

Use Judgment in Defining Test Sets

A *Choose Your Own Adventure* moment

Questionnaire

It can be hard to know in pages and pages of prose what the key things are that you really need to focus on and remember. So, we've prepared a list of questions and suggested steps to complete at the end of each chapter. All the answers are in the text of the chapter, so if you're not sure about anything here, reread that part of the text and make sure you understand it. Answers to all these questions are also available on the book's website. You can also visit the forums if you get stuck to get help from other folks studying this material.

For more questions, including detailed answers and links to the video timeline, have a look at Radek Osmulski's aiquizzes.

1. Do you need these for deep learning?
   - Lots of math T / F
   - Lots of data T / F
   - Lots of expensive computers T / F
   - A PhD T / F
2. Name five areas where deep learning is now the best in the world.
3. What was the name of the first device that was based on the principle of the artificial neuron?
4. Based on the book of the same name, what are the requirements for parallel distributed processing (PDP)?
5. What were the two theoretical misunderstandings that held back the field of neural networks?
6. What is a GPU?
7. Open a notebook and execute a cell containing: `1+1`. What happens?
8. Follow through each cell of the stripped version of the notebook for this chapter. Before executing each cell, guess what will happen.
9. Complete the Jupyter Notebook online appendix.
10. Why is it hard to use a traditional computer program to recognize images in a photo?
11. What did Samuel mean by "weight assignment"?
12. What term do we normally use in deep learning for what Samuel called "weights"?
13. Draw a picture that summarizes Samuel's view of a machine learning model.
14. Why is it hard to understand why a deep learning model makes a particular prediction?
15. What is the name of the theorem that shows that a neural network can solve any mathematical problem to any level of accuracy?
16. What do you need in order to train a model?
17. How could a feedback loop impact the rollout of a predictive policing model?
18. Do we always have to use 224×224-pixel images with the cat recognition model?
19. What is the difference between classification and regression?
20. What is a validation set? What is a test set? Why do we need them?
21. What will fastai do if you don't provide a validation set?
22. Can we always use a random sample for a validation set? Why or why not?
23. What is overfitting? Provide an example.
24. What is a metric? How does it differ from "loss"?
25. How can pretrained models help?
26. What is the "head" of a model?
27. What kinds of features do the early layers of a CNN find? How about the later layers?
28. Are image models only useful for photos?
29. What is an "architecture"?
30. What is segmentation?
31. What is `y_range` used for? When do we need it?
32. What are "hyperparameters"?
33. What's the best way to avoid failures when using AI in an organization?

Further Research

Each chapter also has a "Further Research" section that poses questions that aren't fully answered in the text, or gives more advanced assignments. Answers to these questions aren't on the book's website; you'll need to do your own research!

1. Why is a GPU useful for deep learning? How is a CPU different, and why is it less effective for deep learning?
2. Try to think of three areas where feedback loops might impact the use of machine learning. See if you can find documented examples of that happening in practice.

● ×