In [1]:

```
from fastai import *
from fastai.vision.all import *
```

In [2]:

```
path = Path();path
```

Out[2]:

```
Path('.')
```

In [3]:

```
path.ls()
```

Out[3]:

```
(#9) [Path('.ipynb_checkpoints'),Path('label.txt'),Path('label_rename.
txt'),Path('PDD271_sample.ipynb'),Path('Sample'),Path('Sample.zip'),Pa
th('test_list.txt'),Path('train_list.txt'),Path('validate_list.txt')]
```

In [4]:

```
label_file = (path/'label.txt')
```

In [5]:

```
with open(label_file) as file:
    lines = file.readlines()
coding_dict = {int(x[-4:-1]):x[:-5] for x in lines}
coding_dict
```

Out[5]:

```
{100: 'Pumpkin virus disease',
 101: 'Pumpkin powdery mildew',
 102: 'Pumpkin downy mildew',
 103: 'Leaf Mustard Healthy',
 104: 'Sunflower bacterial leaf spot',
 105: 'Sunflower Black Spot',
 106: 'Chinese cabbage gray spot',
 107: 'Chinese cabbage anthracnose',
 108: 'Chinese cabbage blight',
 109: 'Chinese cabbage virus disease',
 110: 'White spot disease of Chinese cabbage',
 111: 'Bacterial brown spot disease of Chinese cabbage',
 112: 'Chinese cabbage bacterial black spot',
 113: 'Magnesium deficiency in Chinese cabbage',
 114: 'Chinese cabbage poisoning',
 115: 'Chinese cabbage downy mildew',
 116: 'Chinese cabbage black spot',
 117: 'Chinese cabbage black rot',
 118: 'Soybean leaf blight',
 119: 'Soybean sunburn',
```

In [6]:

```
def directory_files(dirct: Path) -> list:
    Lf = []
    if dirct.is_dir():
        for d in dirct.iterdir():
            Lf.extend(directory_files(d))
    else:
        Lf.append(str(dirct))
    return Lf
```

In [7]:
```
fname = directory_files(path/'Sample')
```

In [10]:
```
labels = [coding_dict[int(Path(x).parent.name)] for x in fname]; labels
```

```
 'Wheat yellow leaf blight',
 'Wheat yellow leaf blight',
 'Hawthorn leaf spot',
 'Hawthorn leaf spot',
 'Hawthorn leaf spot',
 'Hawthorn leaf spot',
 'Hawthorn leaf spot',
 'Hawthorn leaf spot',
 'Hawthorn leaf spot',
 'Hawthorn leaf spot',
 'Hawthorn leaf spot',
 'Hawthorn leaf spot',
 'Hawthorn rust',
 'Hawthorn rust',
 'Hawthorn rust',
 'Hawthorn rust',
 'Hawthorn rust',
 'Hawthorn rust',
 'Hawthorn rust',
 ...]
```

In [11]:
```
len(fname)==len(labels)
```

Out[11]:
```
True
```

In [12]:

```
df = pd.DataFrame()
df['fname'] = fname
df['labels']= labels
df
```

Out[12]:

	fname	labels
0	Sample/100/DSC05819.JPG	Pumpkin virus disease
1	Sample/100/DSC05982.JPG	Pumpkin virus disease
2	Sample/100/DSC06018.JPG	Pumpkin virus disease
3	Sample/100/DSC06024.JPG	Pumpkin virus disease
4	Sample/100/DSC06040.JPG	Pumpkin virus disease
...	...	...
10160	Sample/370/DSC01268.JPG	Cucumber black spot
10161	Sample/370/DSC01489.JPG	Cucumber black spot
10162	Sample/370/DSC01512.JPG	Cucumber black spot
10163	Sample/370/DSC01536.JPG	Cucumber black spot
10164	Sample/370/DSC01686.JPG	Cucumber black spot

10165 rows × 2 columns

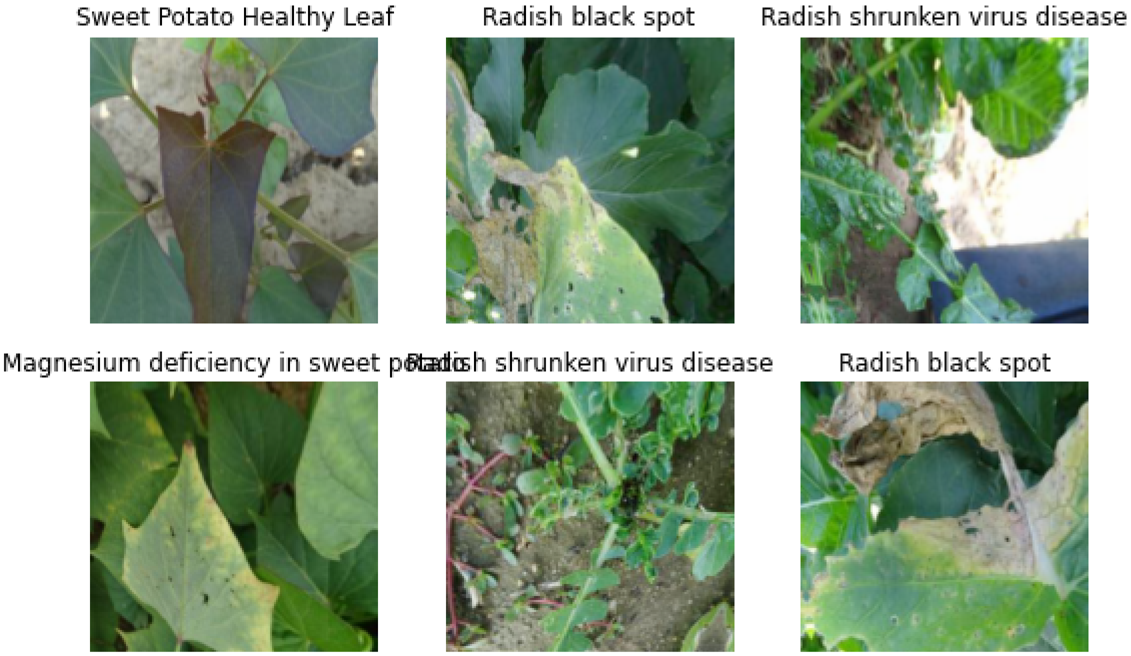
Create a datablock

In [16]:

```
dls = ImageDataLoaders.from_df(df,
                               path,
                               item_tfms=Resize(460),
                               batch_tfms=aug_transforms(size=224),
                               bs=16,
                              )
```

In [17]:

```
dls.show_batch()
```



Sweet Potato Healthy Leaf

Radish black spot

Radish shrunken virus disease

Magnesium deficiency in sweet potato

Radish shrunken virus disease

Radish black spot

In [19]:

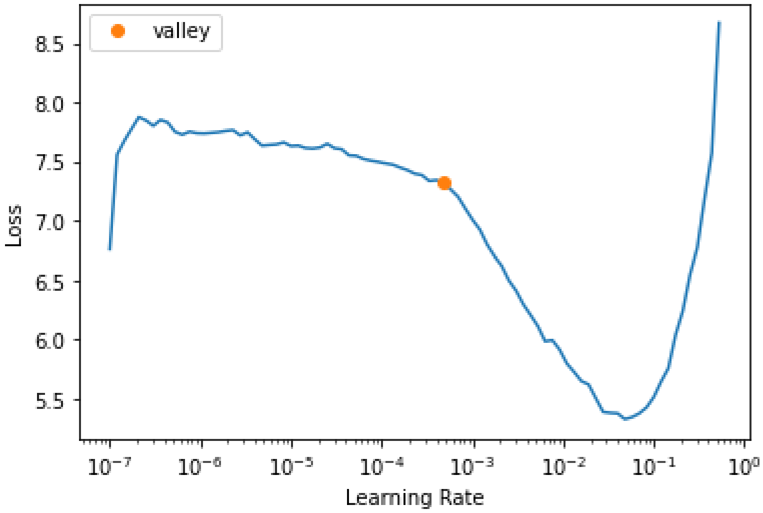
```
learn = cnn_learner(dls, resnet50, metrics=accuracy)
```

```
/home/akmal/anaconda3/envs/fastai/lib/python3.9/site-packages/torch/n
n/functional.py:718: UserWarning: Named tensors and all their associat
ed APIs are an experimental feature and subject to change. Please do n
ot use them for anything important until they are released as stable.
(Triggered internally at  /opt/conda/conda-bld/pytorch_1623448255797/w
ork/c10/core/TensorImpl.h:1156.)
  return torch.max_pool2d(input, kernel_size, stride, padding, dilatio
n, ceil_mode)
```

In [20]:
```
learn.lr_find()
```

```
/cache/fun/fastai/fastai/fastai/callback/schedule.py:270: UserWarning:
color is redundantly defined by the 'color' keyword argument and the f
mt string "ro" (-> color='r'). The keyword argument will take preceden
ce.
  ax.plot(val, idx, 'ro', label=nm, c=color)
```

Out[20]:
```
SuggestedLRs(valley=0.0004786300996784121)
```

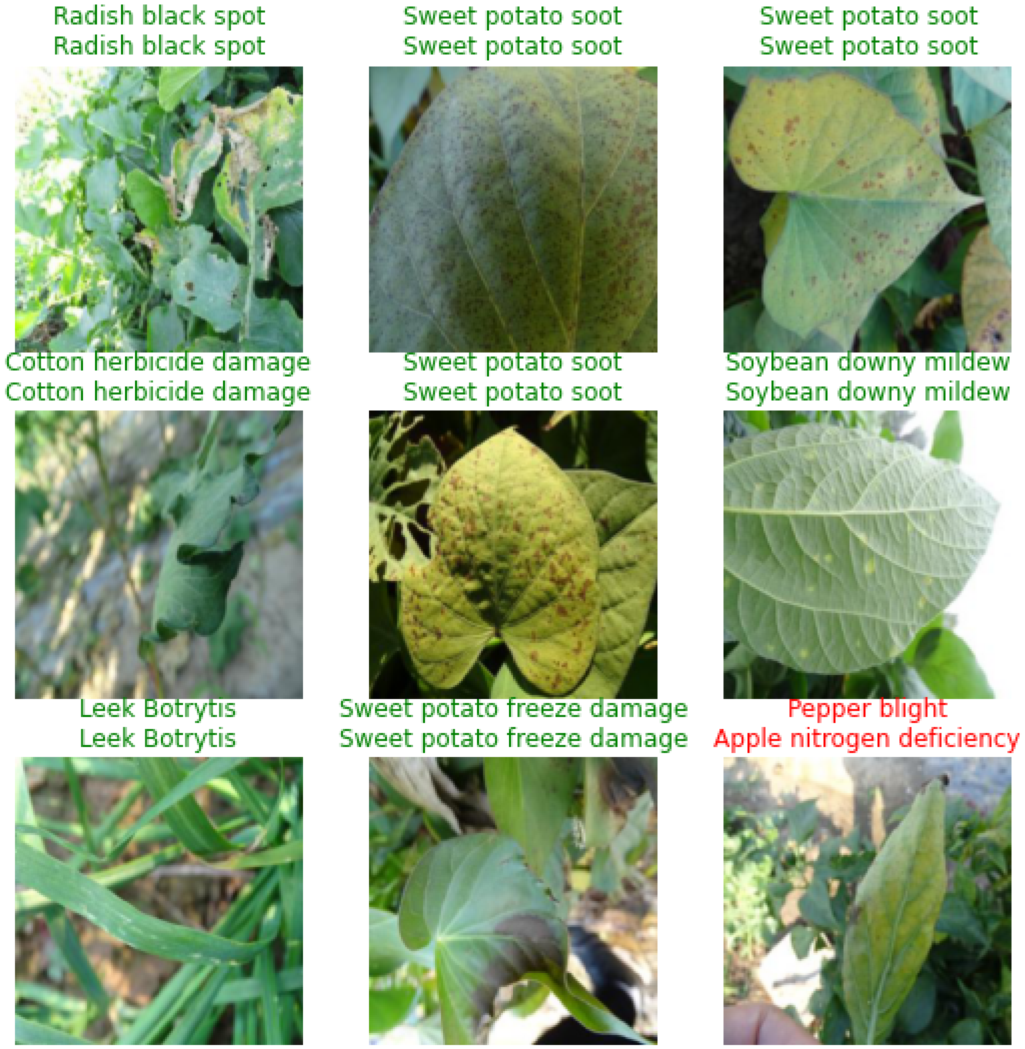


In [21]:
```
learn.fine_tune(5)
```

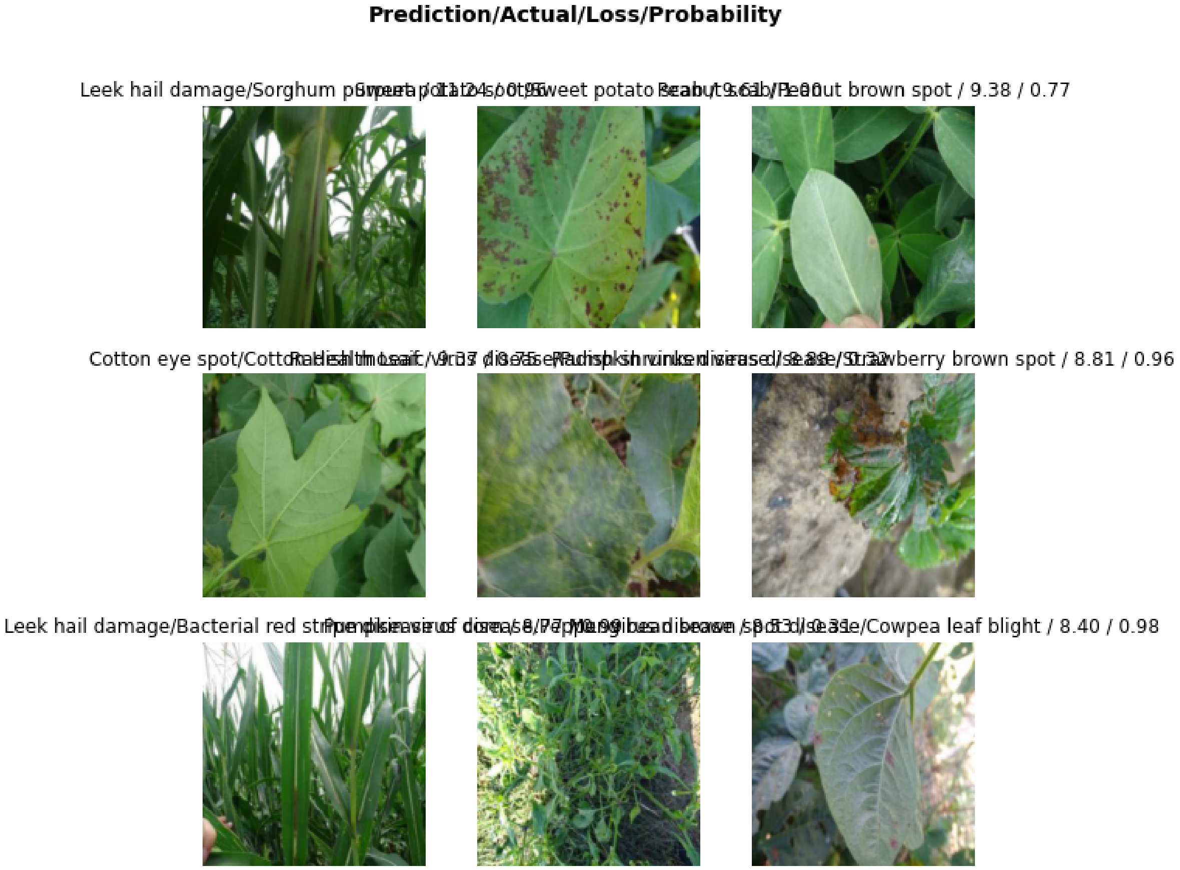
epoch	train_loss	valid_loss	accuracy	time
0	1.825670	1.469818	0.725529	00:31

epoch	train_loss	valid_loss	accuracy	time
0	1.321742	1.122570	0.764388	00:38
1	1.103199	1.015752	0.773733	00:38
2	0.744163	0.698069	0.841613	00:38
3	0.406617	0.544702	0.870143	00:38
4	0.234577	0.525607	0.875062	00:38

```
In [36]: learn.show_results()
```



In [34]:
```
interp = Interpretation.from_learner(learn)
interp.plot_top_losses(9)
```


Prediction/Actual/Loss/Probability
Leek hail damage/Sorghum pSweet potato sot/Sweet potato scab/Peanut brown spot / 9.38 / 0.77
Cotton eye spot/CottoRadish mosaic virus disease/Pumpkin wrinkled virus disease/Strawberry brown spot / 8.81 / 0.96
Leek hail damage/Bacterial red stripe disease of corn/Pepper mung bean brown spot disease/Cowpea leaf blight / 8.40 / 0.98

In [35]:
```
interp.plot_top_losses??
```

In [ ]:
```

```