Questionnaire 2

1. Provide an example of where the bear classification model might work poorly in production, due to structural or style differences in the training data.

   There are many cases that the bear classification model could fail, especially if these cases were not represented in the training data:

   - The bear is partially obstructed
   - Nighttime images are passed into the model
   - Low-res images are passed into the model
   - The bear is far away from the camera
   - The bear training dataset is highly biased toward one type of feature (i.e., color)

2. Where do text models currently have a major deficiency?

   Text models can generate appropriate text (like replies or imitating author style); however, text models still struggle with generating *correct* responses. Given factual information (such as a knowledge base), it is still hard to generate responses that utilizes this information to generate factually correct responses, though this text can seem very compelling. This can be very dangerous, because people unfamiliar with the subject may not be able to evaluate the factual accuracy of the generated text.

3. What are possible negative societal implications of text generation models?

   The ability for text generation models to generate context-aware, highly compelling responses can be used at a massive scale to spread disinformation and encourage conflict.

   Models also reinforce bias (like gender or racial bias) in training data and can create a vicious cycle of biased outputs.

4. In situations where a model might make mistakes, and those mistakes could be harmful, what is a good alternative to automating a process?

   The prediction could then be reviewed by a human professional, who can determine whether the prediction was accurate or not. For example, a machine learning model for identifying strokes in CT scans can alert high priority cases for expedited review, while other cases can be sent to radiologist for regular review.

   Other models can also augment a medical professional's abilities, reducing risk but improving efficiency of the workflow. For example, deep learning models can provide useful measurements for radiologists or pathologists.

5. What kind of tabular data is deep learning particularly good at?

   Deep learning is good at analyzing tabular data that includes natural language, or high cardinality categorical items (containing larger number of discrete choices like zip code).

6. What's a key downside of directly using a deep learning model for recommendation systems?

Current (as of 2020) machine learning approaches for recommendation systems will only tell what products a user might like, and may not be recommendations that would be helpful to the user. For example, if a user if familiar with other books from the same author, it isn't helpful to recommend those products even though the user bought the author's book.

7. What are the steps of the Drivetrain Approach?
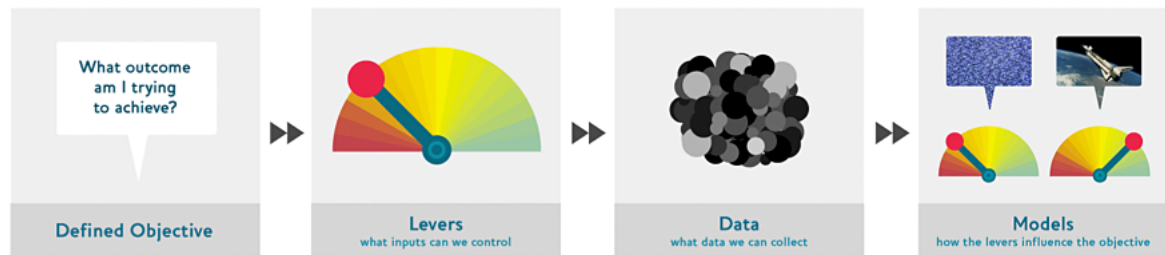
8. How do the steps of the Drivetrain Approach map to a recommendation system?

The **objective** of a recommendation engine is to drive additional sales by surprising and delighting the customer with recommendations of items they would not have purchased without the recommendation. The **lever** is the ranking of the recommendations. New **data** must be collected to generate recommendations that will cause new sales . This will require conducting many randomized experiments in order to collect data about a wide range of recommendations for a wide range of customers. This is a step that few organizations take; but without it, you don't have the information you need to actually optimize recommendations based on your true objective (more sales!)

9. Create an image recognition model using data you curate, and deploy it on the web. Placeholder

10. What is `DataLoaders`?

The `DataLoaders` class is the class that passes the data to the fastai model. It is essentially a class that stores the required `DataLoader` objects (usually for train and validation sets).

11. What four things do we need to tell fastai to create `DataLoaders`?

- what kinds of data we are working with
- how to get the list of items
- how to label these items
- how to create the validation set

12. What does the splitter parameter to `DataBlock` do?

The splitter parameter is used as a way for fastai to split up the dataset into multiple subsets (usually train and validation sets). For example, to randomly split the data, you can use fastai's predefined `RandomSplitter` class, providing it with the proportion of the data used for validation.

13. How do we ensure a random split always gives the same validation set?

It turns out that it is impossible to create truly random numbers. Instead, computers use a process known as a pseudo-random generator. However, this process can be controlled using a random seed. By setting a seed value, the pseudo-random generator will generate the same "random" numbers in a fixed manner which will be the same for every run as long as the seed is the same. By using a random seed, we can generate a random split that always gives the same validation set.

14. What letters are often used to signify the independent and dependent variables?

x is independent. y is dependent.

15. What's the difference between the `crop`, `pad`, and `squish Resize()` approaches? When might you choose one over the others?
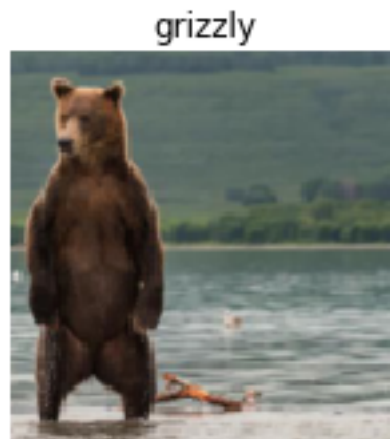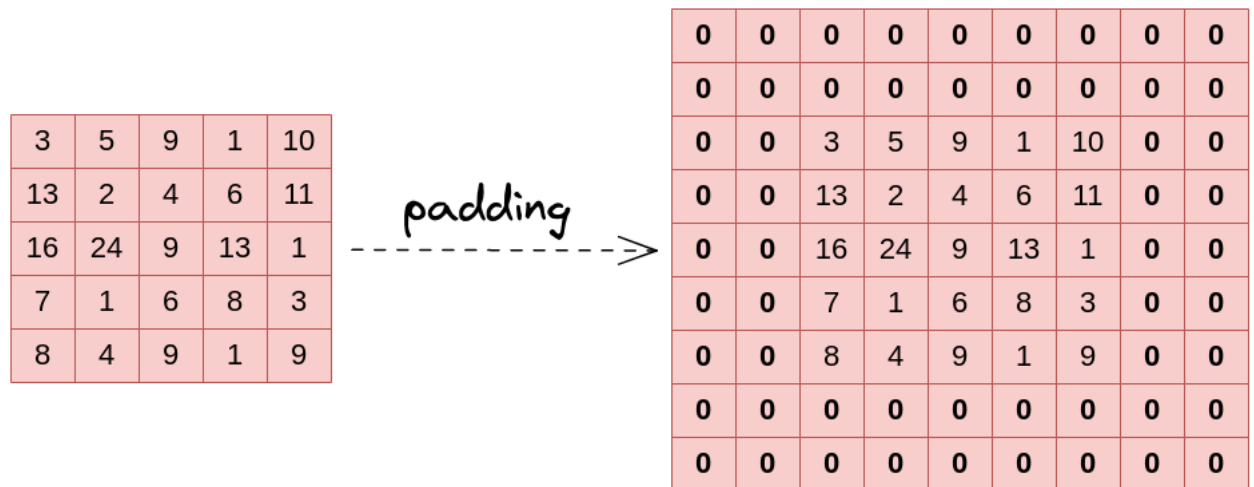

crop is the default `Resize()` method, and it *crops* the images to fit the a square shape of the size requested, using the full width or height.

In this case, this can result in losing some important details. For example, if we were trying to recognize the breed of dog or cat, we may end up cropping out a key part of the body or the face necessary to distinguish between similar breeds.
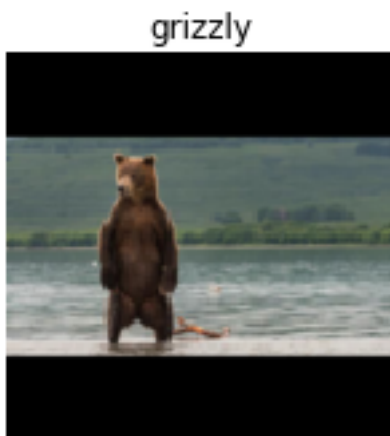
`pad` is an alternate `Resize()` method, which pads the matrix of the image's pixels with zeros (which shows as black when viewing the images).

3	5	9	1	10
13	2	4	6	11
16	24	9	13	1
7	1	6	8	3
8	4	9	1	9

padding $\longrightarrow$

0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	3	5	9	1	10	0	0
0	0	13	2	4	6	11	0	0
0	0	16	24	9	13	1	0	0
0	0	7	1	6	8	3	0	0
0	0	8	4	9	1	9	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

For convolutional networks, the center pixels are much more relevant than corner pixels, which are employed less often and are not as involved when applying a convolutional operation to a matrix. This causes the edges of the **input** picture to lose relevant information. Padding attempts to solve this issue.

In this case, padding images results in a lot of empty space, which results in a lower effective resolution for the part of the image we actually use.

grizzly
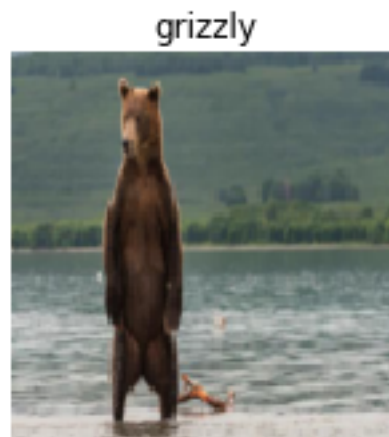
`squish` is another alternative `Resize()` method, which can either squish or stretch the image. This can cause the image to take on an unrealistic shape, leading to a model that learns that things look different to how they actually are, which we would expect to result in lower accuracy.

Which resizing method to use therefore depends on the underlying problem and dataset. For example, if the features in the dataset images take up the entire image and cropping may result in loss of information, then padding or squishing may be more suitable. Another better method is `RandomResizedCrop`, in which we crop on a randomly selected region of the image; during every epoch, the model will see a different part of the image and will learn accordingly.

16. What is data augmentation? Why is it needed?

    Data augmentation refers to creating random variations of our input data, such that they appear different, but not so different it changes the meaning of the data. Examples include flipping, rotation, perspective warping, brightness changes, etc.

    Data augmentation is useful for the model to better understand the basic concept of what an object is and how the objects of interest are represented in images. Therefore, data augmentation allows machine learning models to *generalize.* This is especially important when it can be slow and expensive to label data, or when there is little data.

17. What is the difference between `item_tfms` and `batch_tfms`?

    `item_tfms` are transformations applied to a single data sample `x` on the CPU. `Resize()` is a common transform because the mini-batch of input images to a CNN must have the same dimensions. Assuming the images are RGB with 3 channels, then `Resize()` as `item_tfms` will make sure the images have the same width and height.

    `batch_tfms` are applied to batched data samples (aka individual samples that have been collated into a mini-batch) on the GPU. They are faster and more efficient than `item_tfms`. A good example of these are the ones provided by `aug_transforms()`. Inside are several batch-level augmentations that help many models.

18. What is a confusion matrix?

    A class confusion matrix is a representation of the predictions made vs. the correct models. The rows of the matrix represent the actual labels while the columns represent the predictions. Therefore, the number of images in the diagonal elements represent the number of correctly classified images, while the off-diagonal elements are incorrectly classified images. Class confusion matrices provide useful information about how well the model is doing and which classes the model might *confusing.*

19. What does `export` save?

    `export` saves both the architecture, as well as the trained parameters of the neural network architecture. It also saves how the `DataLoaders` are defined.

20. What is it called when we use a model for getting predictions, instead of training?

    Inference.

21. What are IPython widgets?

    IPython widgets are JavaScript and Python combined functionalities that let us build and interact with GUI components directly in a Jupyter notebook. An example of this would be an upload button, which can be created with the Python function `widgets.FileUpload()`.

22. When might you want to use CPU for deployment? When might GPU be better?

GPUs are best for doing identical work in parallel. If you will be analyzing single pieces of data at a time (like a single image or a single sentence), then CPUs may be more cost-effective instead, especially with more market competition for CPU servers vs. GPU servers. GPUs could be used if you collect user responses into a batch at a time, and perform inference on the batch. This may require the user to wait for model predictions. Additionally, there are many other complexities when it comes to GPU inference, like memory management and queuing of the batches.

23. What are the downsides of deploying your app to a server, instead of to a client (or edge) device such as a phone or PC?

The application will require a network connection, and there will be extra network latency time when submitting input and returning results. Additionally, sending private data to a network server can lead to security concerns.

Conversely, deploying a model to a server makes it easier to iterate and roll out new versions of a model. This is because you, as a developer, have full control over the server environment and only need to do it once rather than having to make sure that all the endpoints (phones, PCs) upgrade their version individually.

24. What are three examples of problems that could occur when rolling out a bear warning system in practice?

The model I trained will likely perform poorly when:

1. Handling nighttime images
2. Dealing with low-res images (ex: some smartphone images)
3. The model returns prediction too slowly to be useful

25. What is "out-of-domain data"?

Out-of-domain data is data that is fundamentally different in some aspect compared to the model's training data. For example, an object detector that was trained exclusively with outside daytime photos is given a photo taken at night.

26. What is "domain shift"?

This is when the type of data changes gradually over time. For example, an insurance company is using a deep learning model as part of their pricing algorithm, but over time their customers will be different, with the original training data not being representative of current data, and the deep learning model being applied on effectively out-of-domain data.

27. What are the three steps in the deployment process?

1. Manual process – The model is run in parallel and not directly driving any actions, with humans still checking the model outputs.

2. Limited scope deployment – The model's scope is limited and carefully surprised. For example, doing a geographically and time-constrained trial of model deployment that is carefully supervised.

3. Gradual expansion – The model scope is gradually increased, while good reporting systems are implemented in order to check for any significant changes to the actions taken compared to the manual process (i.e. the models should perform similarly to the humans, unless it is already anticipated to be better).

Further Research

1. Consider how the Drivetrain Approach maps to a project or problem you're interested in.

2. When might it be best to avoid certain types of data augmentation?

   Some common data augmentation techniques for convolutional neural networks (CNN) are:

   1. Cropping

   2. Flipping

   3. Scaling

   4. Brightness

   5. Color Augmentation

   6. Saturation

   7. Contrast

   8. Translation

   9. Rotation

3. For a project you're interested in applying deep learning to, consider the thought experiment "What would happen if it went really, really well?"

4. Start a blog, and write your first blog post. For instance, write about what you think deep learning might be useful for in a domain you're interested in.