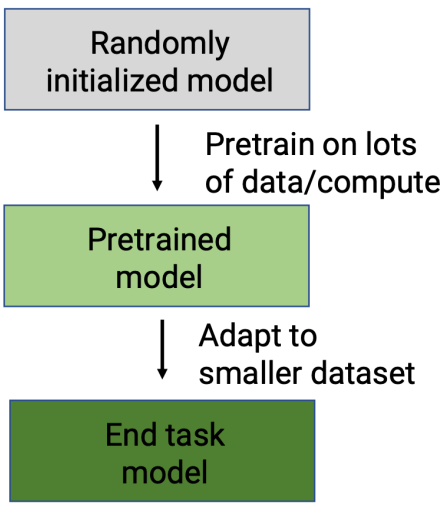
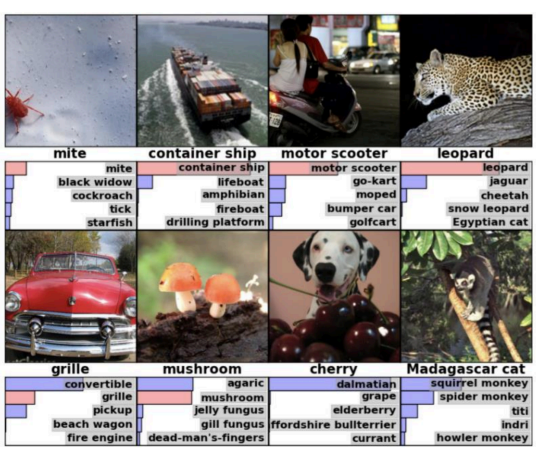
Pretraining

- Neural networks learn to extract features useful for some training task
  - The more data you have, the more successful this will be
- If your training task is very general, these features may also be useful for other tasks!
- Hence: Pretraining
  - First pre-train your model on one task with a lot of data
  - Then use model's features for a task with less data
  - Upends the conventional wisdom: You can use neural networks with small datasets now, if they were pretrained appropriately!



ImageNet Features



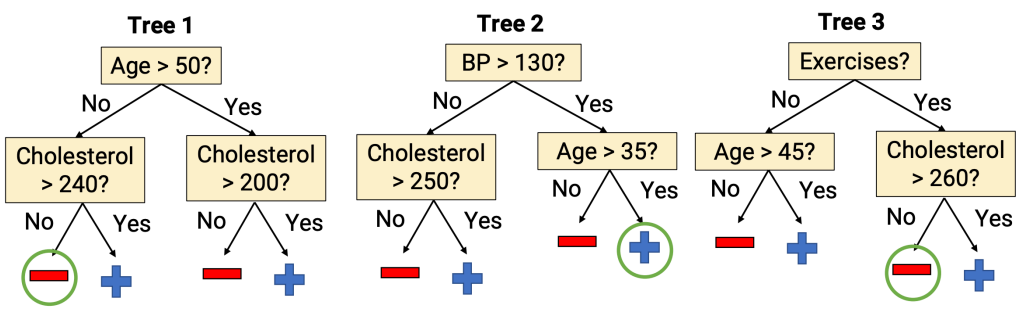
- ImageNet dataset: **14M** images, 1000-way classification
- Most applications don't have this much data
- **But the same features are still useful**
- Using "frozen" pretrained features
  - Get a (small) dataset for your task
  - Generate features from ImageNet-trained model on this data
  - Train linear classifier (or shallow neural network) using ImageNet features

Masked Language Modeling (MLM)
- MLM: Randomly mask some words, train model to predict what's missing
    - Doing this well requires understanding grammar, world knowledge, etc.
    - Get training data just by grabbing any text and randomly delete words
    - Thus: Crawl internet for text data
- Transformers are good fit due to scalability
    - Large matrix multiplications are highly optimized on GPUs/TPUs
    - Don't need lots of operations happening in series (like RNNs)
- Most famous example: BERT

Fine-tuning
- Initialize parameters with BERT
    - BERT was trained to expect every input to start with a special token called [CLS]
- Add parameters that take in the output at the [CLS] position and make prediction
- Keep training all parameters ("fine-tune") on the new task
- Point: BERT provides very good initialization for SGD

Decision Trees
- At each node, split on one feature
- Remember the best output at each leaf node
    - Classification: Majority class
    - Regression: Mean within node
- Given new example, find which leaf node it belongs to and predict the associated output

Ensembling
- Create an "ensemble" of multiple models (e.g., multiple trees)
- Make final prediction by averaging/majority vote



Bagging
- How do you learn different trees from the same dataset?
- Idea: Randomly resample the dataset!
    - Given dataset with n examples, sample a new dataset of n examples with replacement
    - Also known as "Bootstrapping"
    - In expectation, each new dataset contains 63% of the original dataset, with some examples duplicated
    - Learn a tree on each resampled dataset