17. Multi-class classification

* Many linear classification models don’t extend naturally to the multiclass case.
* A common technique is to reduce multiclass classication into several instances of binary classification problems.
* Two kind of “hacky” ways to reduce multi-class classification into binary classification:
  + the one-vs.-rest approach
  + the one-vs.-one approach

### One vs. Rest

* 1v{2,3}, 2v{1,3}, 3v{1,2}
* Learn a binary model for each class which tries to separate that class from all of the other classes.
* If you have k classes, it’ll train k binary classifiers, one for each class. (This makes our interpretation harder)
* **Trained on imbalanced datasets containing all examples. (Every time you try to the train on a binary classification, the rest of the classes become one majority and your own class becomes the minority)**
* Given a test point, get scores from all binary classifiers (e.g., raw scores for logistic regression).
* The classifier which has the highest score for this class “wins” and that’s going to be the prediction for this class.
* Since we have one binary classifier per class, for each class, we have coefficients per feature and an intercept.

### One Vs. One approach (most likely works a little bit better)

* Build a binary model for each pair of classes.
* 1v2, 1v3, 2v3
* Trains n×(n−1)/2 binary classifiers **(more time-consuming)**
* Trained on **relatively balanced** subsets (pro against one vs rest)

### One Vs. One prediction

* Apply all of the classifiers on the test example.
* Count how often each class was predicted.
* Predict the class with most votes.

Computer Vision

## **Transfer learning**

* In practice, very few people train an entire CNN from scratch because it requires a **large dataset, powerful computers, and a huge amount of human effort to train the model.**
* Instead, a common practice is to download a pre-trained model and fine tune it for your task.
* This is called **transfer learning**.
* Transfer learning is one of the most common techniques used in the context of computer vision and natural language processing.

## **Using pre-trained models as feature extractor**[**¶**](https://ubc-cs.github.io/cpsc330/lectures/17_intro_to_computer-vision.html#using-pre-trained-models-as-feature-extractor)

* Here we will use pre-trained models to extract features.
* We will pass our specific data through a pre-trained network to get a feature vector for each example in the data.
* You train a machine learning classifier such as logistic regression or random forest using these extracted feature vectors.