Decision Trees

Announcements

- No class on Wednesday Jan 29th - I’m out of town.
- Assignment still due on that Wed afternoon. - you have all information you need.
- Office hours: Wednesdays, 3-5PM (I’ll be there today, but not next week.)

Decision Trees

Algorithm 1 DecisionTreeTrain(*data*, *remaining features*)

```
guess ← most frequent answer in data                        // default answer for this data
if the labels in data are unambiguous then
    return Leaf(guess)                                      // base case: no need to split further
else if remaining features is empty then
    return Leaf(guess)                                      // base case: cannot split further
else                                                        // we need to query more features
    for all f ∈ remaining features do
        NO ← the subset of data on which f=no
        YES ← the subset of data on which f=yes
        score[f] ← # of majority vote answers in NO
                    + # of majority vote answers in YES
                                                // the accuracy we would get if we only queried on f
    end for
    f ← the feature with maximal score(f)
    NO ← the subset of data on which f=no
    YES ← the subset of data on which f=yes
    left ← DecisionTreeTrain(NO, remaining features \ {f})
    right ← DecisionTreeTrain(YES, remaining features \ {f})
    return Node(f, left, right)
end if
```

Decision Trees

Algorithm 2 DecisionTreeTest(*tree*, *test point*)

```
if tree is of the form Leaf(guess) then
    return guess
else if tree is of the form Node(f, left, right) then
    if f = no in test point then
        return DecisionTreeTest(left, test point)
    else
        return DecisionTreeTest(right, test point)
    end if
end if
```

Questions

- Is this greedy strategy good?

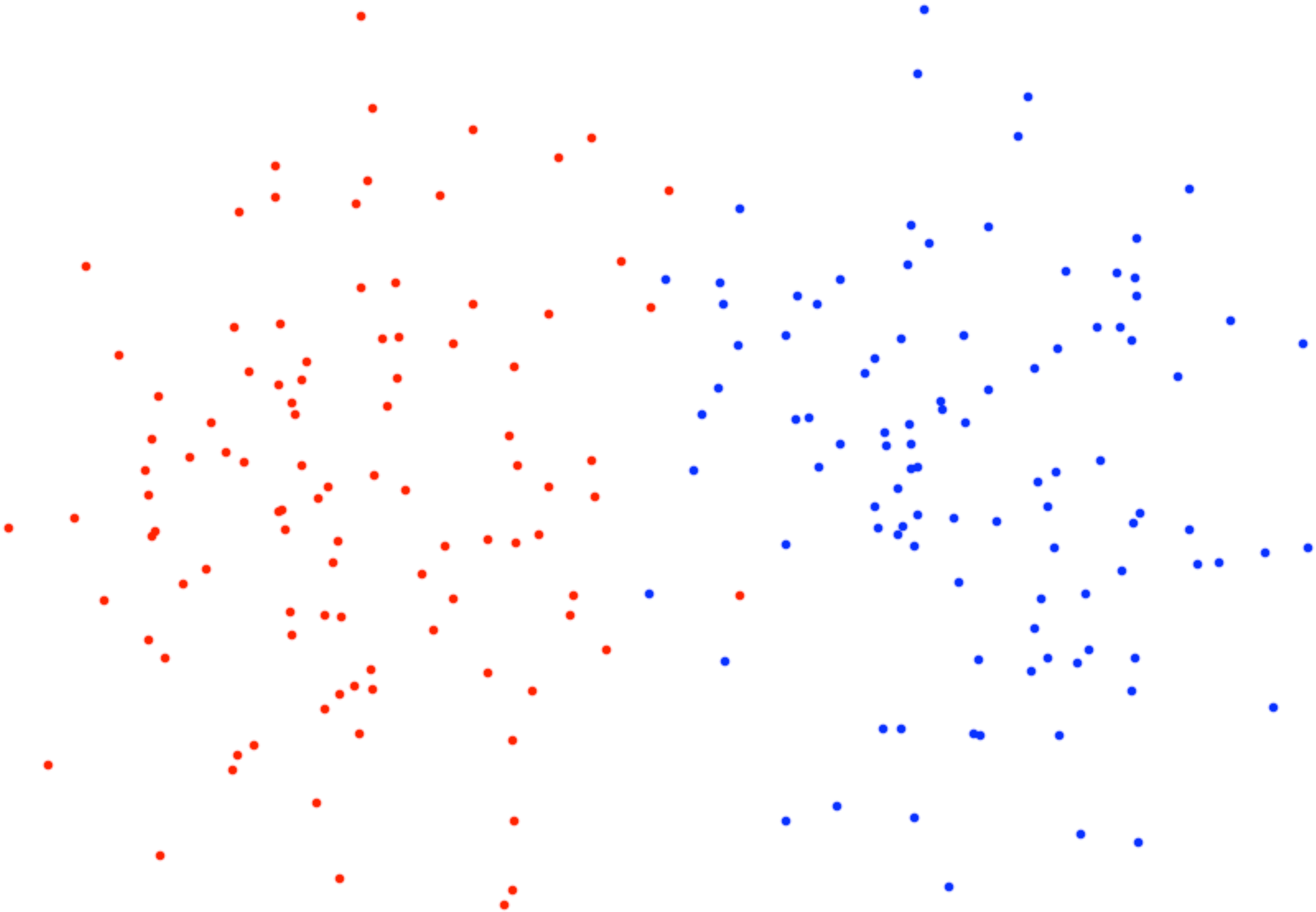
Questions

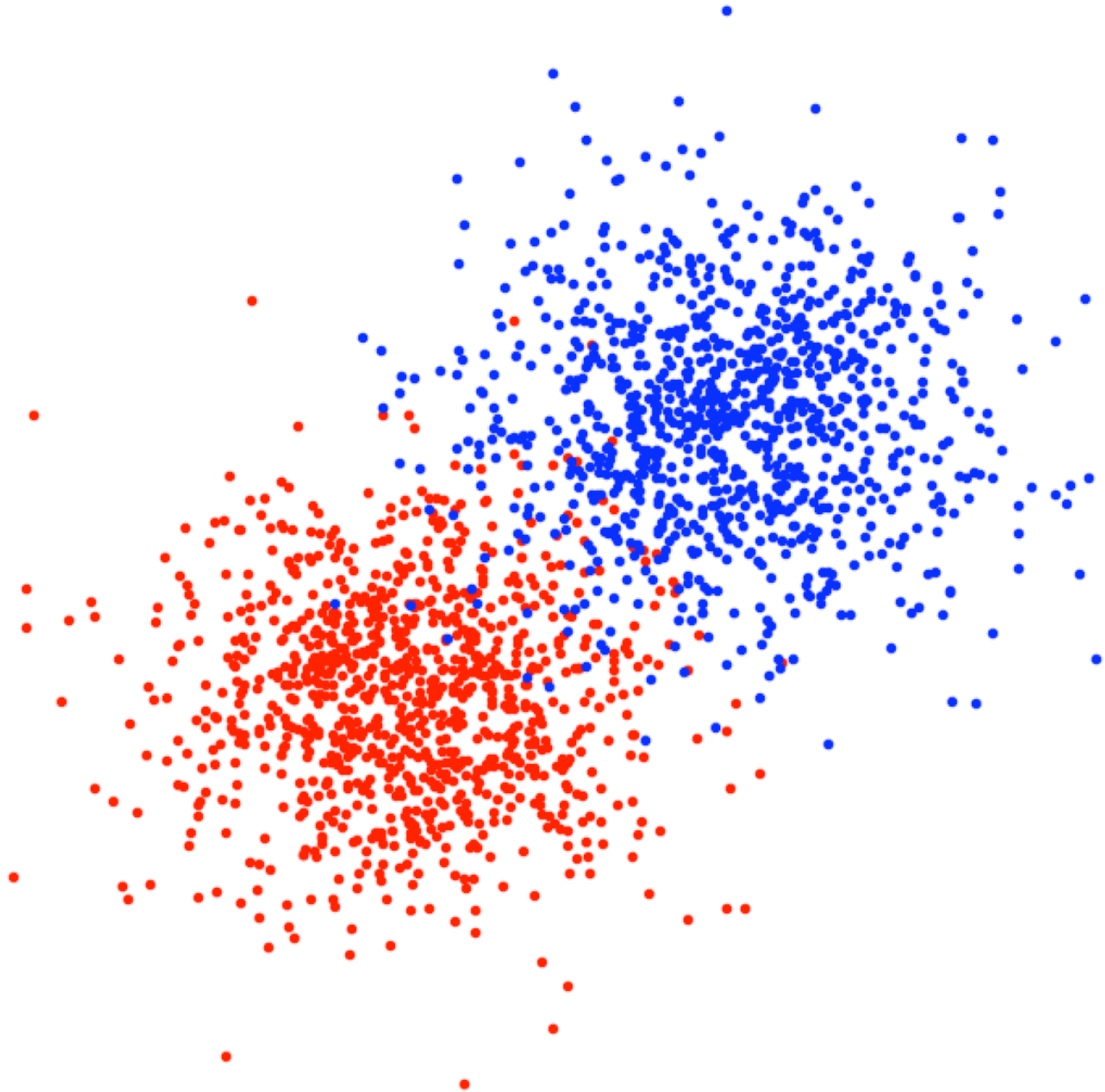
- Is this greedy strategy **always** good?

Extensions to Ponder

- What if labels are categorical but not binary?
- What if features are categorical but not binary?

- What if features are numeric?
- What if labels are numeric?





Induction Learning

As you’ve seen, there are several issues that we must take into account when formalizing the notion of learning.

- The performance of the learning algorithm should be measured on unseen “test” data.
- The way in which we measure performance should depend on the problem we are trying to solve.
- There should be a strong relationship between the data that our algorithm sees at training time and the data it sees at test time.

Setting up the learning problem

- Where does the data come from?
  - Data Generating Distribution
- How do we define that the model is good?
  - The loss function
- How do we combine those?

$$\epsilon \triangleq \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\ell(y, f(\boldsymbol{x}))\right] = \sum_{(x,y)} \mathcal{D}(\boldsymbol{x}, y)\ell(y, f(\boldsymbol{x}))$$

So, putting it all together, we get a formal definition of induction machine learning: **Given (i) a loss function ℓ and (ii) a sample D from some unknown distribution $\mathcal{D}$, you must compute a function f that has low expected error ϵ over $\mathcal{D}$ with respect to ℓ.**

So, putting it all together, we get a formal definition of induction machine learning: **Given (i) a loss function ℓ and (ii) a sample D from some unknown distribution $\mathcal{D}$, you must compute a function f that has low expected error ϵ over $\mathcal{D}$ with respect to ℓ.**

? Why might it be a bad idea to use zero/one loss to measure performance for a regression problem?

So, putting it all together, we get a formal definition of induction machine learning: **Given (i) a loss function ℓ and (ii) a sample D from some unknown distribution $\mathcal{D}$, you must compute a function f that has low expected error ϵ over $\mathcal{D}$ with respect to ℓ.**

? Verify by calculation that we can write our training error as $\mathbb{E}_{(x,y)\sim D}\left[\ell(y, f(\boldsymbol{x}))\right]$, by thinking of D as a distribution that places probability $1/N$ to each example in D and probability 0 on everything else.