Feature Selection

Why?

* Useful to interpret data and for insight
  + “Of the 1000 features, these 10 matter…”
* Curse of dimensionality
  + As we add more features, the amount of data we need grows exponentially. Should help us reduce the data and make the learning problem easier.

How hard is the problem? N features, find m features where m <= n.

* Exponential. Would have to try all subsets, which would be m choose n (n over m). 2^n.
* Np hard

Given that we have this hard problem, how do we approach it?

* Filtering
  + Input set of features -> SEARCH -> outputs fewer features to learning algorithm
  + Pros
    - Speed
    - Can do some sort of approximation and make search arbitrarily fast
  + Cons
    - Ignores the learning problem
      * Could disregard feature due to it by itself, but maybe it in tandem with another feature would increase performance
  + Analogous to decision trees
    - Can use information gain to choose features
    - Can use inductive bias of decision trees to pick features then use inductive bias of learner to combat decision tree bias
  + Criteria for choosing features
    - Information gain (depends on labels)
    - Variance, entropy (doesn’t depend on labels)
    - “useful” features
    - Non-redundant (independent) features
      * Get rid of linearly dependent features (x2 = x1 + x3, get rid of x2).
    - Needs to make learning problem easier (need less data) while also not making it harder (giving it enough features to succeed).
* Wrapping
  + Input set of features -> SEARCH and asks learning algorithm how it’s doing
  + Search for features is *wrapped* around learning algorithm
  + Pros
    - Takes into account model bias – worried about learning problem
  + Cons
    - Slow – learning algorithm may takes thousands of iterations before even moving to next iteration.
  + How to run?
    - Can use randomized optimization
    - forward search
      * start with a feature and work down (1, 2, 3,…), then run permutations, adding a feature each time and summing the performance
        + first iteration, add figure 1
        + second iteration, add figure 3
        + third iteration, add figure 5
      * similar to hill climbing
      * similar to drafting players for a team
    - backward search
      * which one can you eliminate first? Get rid of the worst performer and keep going
        + first iteration, get rid of figure 4
        + second iteration, get rid of figure 2
        + third iteration, get rid of figure 5
      * similar to cutting people off a team for tryouts
* Relevance
  + feature is strongly relevant if removing it degrades B.O.C. (Bayes optimal classifier)
    - B.O.C. – takes weighted average of all the hypotheses (best you could do on average). Captures what the optimal thing might do. Is a measure of the information of variables.
  + feature is weakly relevant if:
    - not strongly relevant
    - there exists a subset of features S such that adding the feature to the subset S it would improve the B.O.C.
  + can be usefully and irrelevant
  + Relevant if it makes the B.O.C better or worse.
  + Relevance has to do with any added information.
* Usefulness
  + Measures effect on a **particular** predictor
  + Usefulness is about minimizing error given some model or learning algorithm
* **Kmeans and EM are a feature transformation algorithm and you’ve taken a bunch of features and converted them into something simple like a label. Whether or not that label is useful depends on whether it helps you do some classification or regression problem later.**

What did we learn?

* Feature selection
* Filtering vs. wrapping
  + Wrapping is slow but useful (solves the problem that matters)
  + Filtering is simpler and faster but ignores bias and possibly misses the point
* Relevance vs usefulness
  + Relevant things give information about class or regress
  + Useful helps do the learning given some specific algorithm
  + Relevance is usefulness with regards to the Bayes classifier
* Strong and weak relevance
  + “indispensable” data