**Machine Learning**

**Session 11**

1. Ensembles: Big Picture
   1. What is the source of error in supervised learning?
      1. We have seen under/over fitting.
      2. Another way to understand this: “Bias-Variance Tradeoff”.
   2. Bias: Tendency of a model to learn the same wrong thing.
      1. E.g., if trying to fit a linear model to a curve.
   3. Variance: Tendency of a model to learn random fluctuations independent of the underlying signal.
      1. E.g., Every time you fit a complicated model like decision tree, you can get quite a different tree.
   4. Ideal: Zero bias + zero variance
   5. We saw complexity control by cross-validation tries to find a good trade-off between under & over-fitting (bias & variance).
   6. Ensembles provide a way to further reduce variance.
   7. Key idea:
      1. Committee of experts collective opinion more reliable than individual expert.
      2. => Collection of models’ opinion more reliable than individual model.
   8. Intuition:
      1. If each expert’s errors are uncorrelated.
      2. Then their collective vote averages out their errors.
   9. Individual expert/model is high variance.
      1. Committee/ensemble is lower variance
2. Ensembles General Methods: Bagging
   1. Ensembles: useful if diverse & uncorrelated predictions
      1. Q: Ideas about how to generate models with diverse predictions?
   2. Bootstrap Aggregation (“Bagging”)
   3. Method:
      1. For N data, take K random samples size N with replacement.
      2. Train K models, one for each subsample
      3. Average or majority vote the predictions
   4. E.g., given models f1(x), f2(x), .. Etc  
        
      f\_bag(x) = (1/E) \* sum of(f\_e(x))
3. Ensembles: General Methods: Bagging: Bootstrapping  
     
     
   Original Dataset

|  |  |
| --- | --- |
| 1 | 6 |
| 2 | 7 |
| 3 | 8 |
| 4 | 9 |
| 5 | 10 |

Bootstrapping: A random sample with replacement

|  |  |
| --- | --- |
| 1 | 6 |
| 10 | 10 |
| 7 | 8 |
| 3 | 1 |
| 10 | 6 |

Unselected

|  |
| --- |
| 2 |
| 4 |
| 5 |
| 9 |

1. Bagging Illustration: A fig of a xy graph with data points clustered together and separated by a straight line. Ensemble methods allow to establish a non-linear line between the data points.
2. Ensembles General Methods: Bagging
   1. Bootstrap Aggregation (“Bagging”)
      1. Train K models on K subsamples, and average.
   2. Notes:
      1. Requires “unstable” (non-linear) base models: where each can be very different: E.g., decision trees (=> expert diversity)
      2. Each ensemble member likely worse than a full model
         1. But collectively better!
      3. Ensemble will have similar bias
         1. … but reduced variance
   3. Tradeoff:
      1. Small bags, worse models, more diversity
      2. Big bags, better models, less diversity
3. Ensembles: General Methods: Random Subspaces and Model Combination
   1. So far we introduced expert diversity by bagging.
   2. Bagging randomizes instances:
      1. Each expert trained on a random subset of instances
   3. Q: What else can we randomize?
   4. “Random Subspaces” randomizes attributes:
      1. Each expert trained on a random subset of attributes/dimensions
   5. Can also do so by any other kind of diversity.
      1. E.g., For methods that converge to a local minima only, start in different initial conditions.
   6. “Model Combination”: combine many models: decision tree, logistic regression, naïve bayes, KNN, etc.
4. Ensembles Model Specific: Random Forests Random Forest
   1. Grow a whole array of decision trees
      1. Average their prediction at test time
   2. Diversity: Each decision tree randomized with both instances (bagging) and dimensions (random subspaces)
   3. Pros:
      1. Generally great accuracy. State of the art for many problems.
   4. Cons:
      1. Decision trees usually interpretable, forests not.
      2. Can be expensive to train many trees (but very parallelizable)
5. Ensembles: What sized committee?
   1. Tradeoff:
      1. Larger committee => better aggregate decision
      2. Larger committee => slower to train & test
         1. May have real-time test constraints
   2. More diverse and uncorrelated
      1. Good performance with small committee
6. Case Study: Kinect
   1. Uses Decision Forest ensemble.
   2. Training the forest
      1. 1 million images
      2. Depth 20 trees, 2000 random features per tree, 300k images per tree
      3. Use distributed implementation: 1 day on 1000 core cluster.
7. Ensembles: General Methods: From Bagging to Stacking
   1. So far we said average/vote each model/expert: f\_ensemble(x) = (1/E) \* sum of(f\_e(x))
   2. Q: What could we do better? f\_ensemble(x) = (1/E) \* sum of(w\_e\*f\_e(x))
   3. What if some experts are better than others?
      1. Could we do better with a weighted vote?
   4. How to determine the per-expert weight?
      1. Run another “meta” learner!
8. Ensembles: General Methods: Stacking
   1. Base Learners
      1. Input: Raw data
      2. Output: Class
   2. Meta Learner
      1. Input: Vector of estimated classes from a bank of base learners
      2. Output: Class
   3. Overall, making a complex super-model
      1. Very complex model => easy to overfit
      2. So keep meta-learner simple: Use linear model
      3. Train meta-learner using cross-validation to reduce overfitting
         1. Important because individual classifiers may be overfit/unrealistically confident
9. Ensembles: Boosting: Motivation
   1. Ensemble methods take a (weighted) sum of model predictions: f\_ensemble(x) = (1/E) \* sum of(w\_e\*f\_e(x))
   2. Each model f\_e (x) is trained independently.
   3. Rely on Instance/Feature or other randomization to generate diversity among the experts
   4. Boosting: Seek to explicitly learn complementary models
   5. Ensembles: Boosting: Mechanism Boosting: Algorithm Sketch  
      Train the first model f\_i=1(x)
   6. Repeat i=2…T
      1. Note which data instances the ensemble so far f1(x)…f\_i-1(x) gets wrong
      2. Train next model fi(x), but focus on training examples currently predicted wrongly
   7. Basic models:
      1. Now forced to be different, unlike bagging
      2. Commonly they are decision stumps
      3. Needs to be able to accept instance weights
   8. Typically boosting builds complex model out of many simple models, instead of out of many complex models (bagging)
10. Ensembles: Boosting: Properties
    1. Ensembles: Weighted sum of model predictions
    2. Contrast: Bagging & Randomisation
       1. Trained independently with randomisation for diversity
    3. Contrast: Boosting
       1. Trained sequentially to explicily seek complementarity.
    4. Boosting Properties
       1. Often best performing ensemble type, but can sometimes overfit, since tuning w (whole ensemble makes a complex classifier)
       2. Can’t parallelize the whole thing like independently trained ensembles
    5. Neat (surprising?) fact: Even if each base learner only 51% accurate, the entire ensemble can be arbitrarily (100%) accurate
11. Aside: Boosted Cascades + Case Study: Face Detection
    1. Almost all embedded face detectors use boosting. Why?
       1. Recall: Boosting builds an ensemble in order: first models will be most useful, later models will “fine-tune”.
    2. Detection Issue: “Sliding window”, means very many classifications needed at test time. 1000x1000 pixel image => 1M-1B classifications.
    3. Variant: Boosted cascade.
       1. Builds an ensemble that also prefers to put cheaper classifiers first.
       2. Test time: Evaluate them in order: 1…. E
       3. If you are confident that its not a face
          1. Then terminate early
       4. Most squares are non-face and << full cost
12. Case Study: Face Detection
    1. Almost all embedded face detectors use boosting
       1. Recall: Boosting builds an ensemble in order.
    2. Variant: Boosted cascade.
       1. Builds an ensemble that also prefers to put cheaper classifiers first.
       2. Test time: Evaluate them in order: 1…. E
       3. If you are confident that its not a face
          1. Terminate early
       4. => Most squares don’t evaluate whole ensemble
       5. => Most squares use << full cost
13. Bagging versus Boosting:

|  |  |
| --- | --- |
| Bagging | Boosting |
| Built independently / parallel | Built sequentially |
| Resample data | Reweight data |
| Reduces variance  – Typically works with complex models | Reduces variance  – Typically also used to reduce bias by combining simple models |
| Parameters:  – How many members  – How to introduce diversity? | Parameters:  – Termination condition |
| f\_ensemble^bag(x) = (1/E) \* sum of(f\_e(x)) | f\_ensemble^boost(x) = (1/E) \* sum of(w\_e\*f\_e(x)) |

1. Ensembles: Summary
   1. Pros
      1. Improves accuracy, often a lot!
      2. Bagging & Subspace randomisation can make overfitting in the individual models less of an issue: (The overfits “average out”)
   2. Cons
      1. More models to train => More expensive
         1. (Maybe parallelizable, except boosting)
      2. Loss of interpretability
      3. Test time is more expensive (have to evaluate many models)
         1. Except boosted cascades
      4. Boosting is weak to label noise & outliers