CS760: Machine Learning Exam Review

Jack Truskowski

3/31/2018

1 Topics

1. Decision Tree Learning
2. Instance-based Learning, K-Nearest Neighbor
3. ML Methodology
4. Linear and Logistic Regression
5. Bayesian Network Learning
6. Neural Networks
7. Deep Neural Networks
8. Learning Theory
9. Support Vector Machines

2 Decision Tree Learning

2.1 Goals

- DT representation
- Standard approach
- Occam's razor
- Entropy / IG
- Types of DT Splits
- Test sets / unbiases accuracy estimates
- Overfitting
- Pruning
- Tuning (validation) sets

- Regression trees
- m-of-splits
- Lookahead

2.2 Notes

- Splits on nominal features have one branch per value
- Splits on continuous features use a threshold
- Candidate Splits on continuous features
  - sorts the values
  - split thresholds in intervals between different classes

- The simplest tree with accurate classification will be the best on unseen data
- **Occams razor**: Simpler models are better
- IG Limitation: biased towards tests with many outcomes
- Avoiding overfitting:
  1. Early stopping: stop if further splitting not justified by statistical test (ID3)
  2. Post pruning: grow a large tree, prune back some nodes, more robust
- Pruning: grow a complete tree, remove the nodes that most improves tuning-set accuracy until further pruning is harmful
- Regression Trees: CART does least squares regression
- Lookahead
  1. myopia: an important feature seems to not be informative until use in conjunction with other features
  2. Replaces the InfoGain step with an EvaluateSplit step
  3. Choose the best info gain that would result from a 2-level subtree

2.3 Relevant Equations

$$H(Y) = -\sum_{y \in \text{values}(Y)} P(y) \log_2 P(y)$$

Entropy:

$$H(Y \mid X) = \sum_{x \in \text{values}(X)} P(X = x)\, H(Y \mid X = x)$$

where

$$H(Y \mid X = x) = -\sum_{y \in \text{values}(Y)} P(Y = y \mid X = x)\, \log_2 P(Y = y \mid X = x)$$

$$\text{InfoGain}(D, S) = H_D(Y) - H_D(Y \mid S)$$

D indicates that we're calculating probabilities using the specific sample *D*

Information Gain:

$$= \sum_{L \in \text{leaves}} \sum_{i \in L} (y_i - \hat{y}_i)^2$$

Least Squares Regression in CART

3 Instance-Based Learning

3.1 Goals

1. k-NN classification
2. k-NN regression
3. edited nearest neighbor
4. k-d trees for nearest neighbor identification
5. locally weighted regression
6. inductive bias

3.2 Notes

1. Determining similarity/distance

(a) Hamming distance: count number of features for which 2 instances differ (discrete only)

(b) Euclidean distance: $d(x^{(i)}, x^{(j)}) = \sqrt{\sum_f (x_f^{(i)} - x_f^{(j)})^2}$

(c) Manhattan distance: $d(x^{(i)}, x^{(j)}) = \sum_f |x_f^{(i)} - x_f^{(j)}|$

(d) If a mix of continuous/discrete features, refer to equations

2. Normalization

- Determine mean and stddev for feature x_i

$$\mu_i = \frac{1}{|D|} \sum_{d=1}^{|D|} x_i^{(d)} \qquad \sigma_i = \sqrt{\frac{1}{|D|} \sum_{d=1}^{|D|} \left(x_i^{(d)} - \mu_i\right)^2}$$

- Standard each feature

$$\hat{x}_i^{(d)} = \frac{x_i^{(d)} - \mu_i}{\sigma_i}$$

$$\hat{y} \leftarrow \frac{1}{k} \sum_{i=1}^{k} y^{(i)}$$

3. k-NN Regression

4. Speeding up k-NN

- Don't retain every training instance
- Use smart data structure to look up nearest neighbors (ie k-d tree)

5. Edited instance-based learning
Incremental deletion, start will all train inst in memory. If other instances provide correct classification for $(x^{(i)}, y^{(i)})$, delete it
Incremental growth, start with empty memory. If other instances don't correctly classify $(x^{(i)}, y^{(i)})$, add it to memory

6. k-d trees

(a) Similar to DT

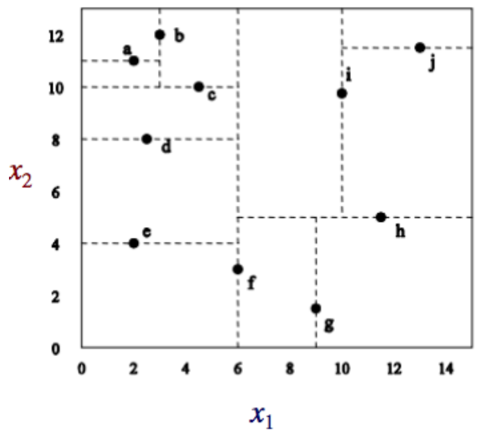
(b) Each node stores one instance

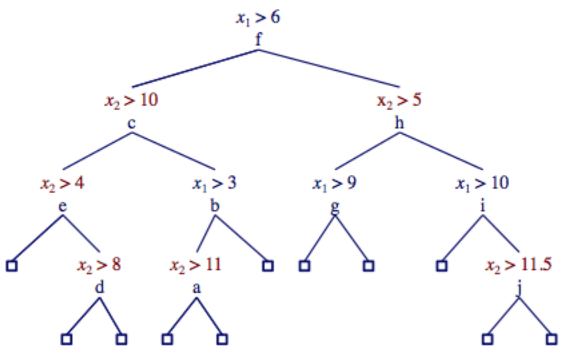
(c) Each node splits on median value of feature with highest variance

(d) Implemented using priority queue storing nodes considered and their lower bound on distance to query instance

(e) k-d trees are sensitive to irrelevant features, **locally weighted regression**

Example:





7. Locally weighted regression

   prediction/learning task

   - find the weights w_i for each $x^{(q)}$ by minimizing

   $$E(\mathbf{x}^{(q)}) = \sum_{i=1}^{k} \left(f(\mathbf{x}^{(i)}) - y^{(i)} \right)^2$$

   - this is done at prediction time, specifcally for $x^{(q)}$
   - can do this using gradient descent (to be covered soon)

8. Stengths of instance-based learning

   (a) simple to implement

   (b) adapts well to online training

   (c) robust to noisy training data with k ¿ 1

   (d) good in practice

9. Limits of instance-based learning

   (a) sensitive to range of feature values

   (b) potentially sensitive to irrelevant and correlated features

   (c) can be inefficient

   (d) no explicit model

3.3 Equations

$$d(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \sum_f \begin{cases} \left| x_f^{(i)} - x_f^{(j)} \right| & \text{if } f \text{ is continuous} \\ 1 - \delta\left(x_f^{(i)}, x_f^{(i)} \right) & \text{if } f \text{ is discrete} \end{cases}$$