CS256 – Midterm Exam Study Guide

By: Zayd Hammoudeh

Chapter #04 – Classification: Basic Concepts, Decision Trees, and Model Evaluation

Classification	Training Set	Model	Test Set	Example Classification Techniques
Task of assigning objects to one of several predefined categories.	A collection of records. Each **record** contains a set of attributes one of which is the **class**.	A function from the value of record attributes to the class attribute.	A collection of records used to determine the accuracy of the classification model.	1. Neural Networks 2. Decision Tree 3. Rule Based Classifier 4. Memory Based Reasoning 5. Support Vector Machines 6. Naïve Bayes and Bayesian Belief Networks

Induction

Using a training set to generate a model.

Deduction

Process of applying a model to a training set.

Decision Tree Induction

- **Greedy Strategy**
- **Key Decision #1:** Attribute to expand next
- **Key Decision #2:** When to stop expanding

Hunt's Decision Tree Induction Algorithm:

- Let D_t be the set of training records that reach a node t.

1. If D_t contains records that **all belong to the same class y_t**, then t is a leaf node with class value y_t.
2. If D_t is an **empty set**, then t is a leaf node with default value y_d.
3. If D_t contains **records that belong to more than one class and there are no attributes left**, then t is a leaf node with default value is a leaf node with default value y_d.
4. If D_t contains **records that belong to more than one class**, then use an attribute test to split the data into smaller subsets. Recursively apply the same procedure above.

Attribute Types

- **Binary** – Attribute with exactly two possible values.
- **Nominal** – Two or more class values with no intrinsic Order
- **Ordinal** – Two or more class values that can be ordered or ranked
- **Continuous** – Quantitative attribute that can be measured along a continuum.

Splitting Nominal and Ordinal Attributes

- **Binary** – Divides attribute values into two subsets. **This requires the additional step of finding optimal partitioning.**
- **Multi-way** – Use as many partitions as distinct values.

Splitting Based on Continuous Attributes

- **Discretization** – Form an ordinal categorical attribute.
  - **Static** – Discretize once at the beginning
  - **Dynamic** – Ranges can be found by equal interval bucketing, equal frequency bucketing, or clustering.
- **Binary Decision** (A < v or A >v) – Consider all possible splits and find the best cut.
  - **Binary Decision Procedure:** Go between each training set record value and calculate the GINI index if the splitting point was at that value. **Select the splitting point with the lowest $\text{GINI}_{\text{SPLIT}}$ value.**
    - **Computationally inefficient** $O(n)$ – where n is the number of records.

Homogeneity/Low Impurity – Extent to which nodes in the decision tree have the same class value/distribution.

Nodes with high levels of homogeneity (i.e. low levels of impurity) are preferred.

Impurity Measures

For all of these metrics, a lower score is generally preferable.

GINI Index

$$GINI(t) = 1 - \sum_{j=1}^{n_c} \left(p(j|t)\right)^2$$

- t – Node in the decision tree
- j – Class value
- n_c – Number of class values
- $p(j|t)$ – Probability (i.e. relative frequency) of class value j in node t

Minimum Value: 0 when:

$$\exists j(p(j|t) = 1)$$

Maximum Value: $1 - \frac{1}{n_c}$ when:

$$\forall j\left(p(j|t) = \frac{1}{n_c}\right)$$

$\text{GINI}_{\text{SPLIT}}$

$$GINI_{SPLIT} = \sum_{i=1}^{k} \frac{n_i}{n} \cdot GINI(i)$$

- i – Child node
- n – Number of records in parent node. Note:

$$n = \sum_{i=1}^{k} n_i$$

- n_i – Number of child nodes (i.e. attribute partitions)
- $GINI(i)$ – GINI index value of node i.

Minimum Value: 0 when:

$$\forall i(GINI(i) = 0)$$

Maximum Value: $1 - \frac{1}{n_c}$ when:

$$\forall i\left(GINI(i) = 1 - \frac{1}{n_c}\right)$$

Entropy

$$Entropy(t) = -\sum_{j=1}^{n_c} p(j|t) \cdot log_2(p(j|t))$$

- t – Node in the decision tree
- j – Class value
- n_c – Number of class values
- $p(j|t)$ – Probability (i.e. relative frequency) of class value j in node t

Minimum Value: 0 when:

$$\exists j(p(j|t) = 1)$$

Maximum Value: $\log_2(n_c)$ when:

$$\forall j\left(p(j|t) = \frac{1}{n_c}\right)$$

Information Gain

$$GAIN_{SPLIT}(t) = Entropy(p) - \left(\sum_{i=1}^{k} \frac{n_i}{n} \cdot Entropy(i)\right)$$

- p – Parent node in the decision tree
- i – Child node in the decision tree
- k – Number of child nodes
- n_i – Number of records in child node i
- n – Number of records in parent node p

$$n = \sum_{i=1}^{k} n_i$$

Key Note: A higher $GAIN_{SPLIT}$ is preferable unlike with the other metrics where a lower value was better.

Disadvantage of Information Gain: Tends to prefer splits that result in a large number of partitions, each being small but pure (i.e. overfitting)

Normalizing for Split Size

$$GainRATIO_{Split} = \frac{Gain_{SPLIT}(t)}{SplitINFO}$$

$$SplitINFO = -\sum_{i=1}^{k} \frac{n_i}{n} \cdot log_2\left(\frac{n_i}{n}\right)$$

$Split_{INFO}$ penalizes a large split by reducing the gain.

Classification Error

$$Error(t) = 1 - max_j\left(p(j|t)\right)$$

- t – Node in the decision tree
- j – Class value
- $p(j|t)$ – Probability (i.e. relative frequency) of class value j in node t

Minimum Value: 0 when:

$$\exists j\left(p(j|t) = 1\right)$$

Maximum Value: $1 - \frac{1}{n_c}$ when:

$$\forall j\left(p(j|t) = \frac{1}{n_c}\right)$$

Stopping Criteria for Decision Tree Induction

Three Stopping Criteria for Decision Tree Induction

- **When all records in a node have the same class value**
- **When all records in a node have similar attribute values**.
- **Early Termination**

Underfitting – When a model is too simple, both training and test errors are large.

Overfitting – When a model becomes too complex (e.g. too large a tree), the test error begins to increase even though the training error decreases.

- **Result:** Training error is **NOT** representative for generalization error.

Causes of Overfitting

- **Noise**
- **Insufficient training records** (including non-representative training set)

Resubstitution Error

Error on the **training** set.

Single Leaf Node Error: $e(t)$

Total Resubstitution Error: $e(T)$

$$e(T) = \sum e(t)$$

Generalization Error

Error on the **testing** data.

Single Leaf Node Error: $e'(t)$

Total Generalization Error: $e'(T)$

$$e'(T) = \sum e'(t)$$

Generalization Error Estimation

Optimistic Estimation

Training error is equal to the testing error.

$$\sum e(t) = \sum e'(t)$$

Pessimistic Estimation

Assign a penalty term to ea.

$$e'(t) = e(t) + 0.5$$

Total Pessimistic Error

$$e'^{(T)} = \sum\left(e(t)\right) + N \cdot 0.5$$

N – Number of leaf nodes.

Reduced Error Pruning

Use a validation set to estimate the generalization error.

Occam's Razor

Given two models with similar generalization errors, one should prefer the simpler model over the more complex model.

This is because more complex model has a greater chance of fitting accidentally by errors in the data.

Pre-pruning (Early Stopping Rule)

- **Stop the induction algorithm before it becomes a full tree.**
- **Typical Stopping Rules:**
  - All remaining records have the same class value
  - All attribute values are the same.
- **More restrictive conditions:**
  - Number of instances is below a user-specified threshold.
  - Expanding the current node does not improve impurity measures (e.g. GINI Index, Information Gain)
  - Class distribution of instances are independent of available features.

Post-pruning (Early Stopping Rule)

- **Grow the decision tree to its entirety**.
- Trim nodes in the tree in a **bottom-up fashion.**
- Only **trim nodes if by trimming the estimate of the generalization error improves**.
- New leaf node's **class label is determined from the majority class of instances in the merged node**.

Minimum Description Length	

Miscellaneous

Decision Tree Algorithm		
Advantages • **Inexpensive to construct** • **Extremely fast at classifying unknown records.** • **Easy to interpret for small sized trees.** • **Accuracy is comparable to other classification techniques for many simple datasets.** (Since everything comes right from the data)		