MIS2502: Data Analytics

Classification using Decision Trees

Jing Gong

gong@temple.edu

http://community.mis.temple.edu/gong
What is classification?

- Determining to what group a data element belongs
  - Or “attributes” of that “entity”

- Examples
  - Determining whether a customer should be given a loan
  - Flagging a credit card transaction as a fraudulent charge
  - Categorizing a news story as finance, entertainment, or sports
How classification works

1. Split the data set into training and validation subsets
2. Choose a discrete outcome variable
3. Find model that predicts the outcome as a function of the other attributes
4. Apply the model to the validation set to check accuracy
5. Apply the final model to future cases
## Decision Tree Learning

### Training Set

<table>
<thead>
<tr>
<th>Trans. ID</th>
<th>Charge Amount</th>
<th>Avg. Charge 6 months</th>
<th>Item</th>
<th>Same state as billing</th>
<th>Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$800</td>
<td>$100</td>
<td>Electronics</td>
<td>No</td>
<td>Fraudulent</td>
</tr>
<tr>
<td>2</td>
<td>$60</td>
<td>$100</td>
<td>Gas</td>
<td>Yes</td>
<td>Legitimate</td>
</tr>
<tr>
<td>3</td>
<td>$1</td>
<td>$50</td>
<td>Gas</td>
<td>No</td>
<td>Fraudulent</td>
</tr>
<tr>
<td>4</td>
<td>$200</td>
<td>$100</td>
<td>Restaurant</td>
<td>Yes</td>
<td>Legitimate</td>
</tr>
<tr>
<td>5</td>
<td>$50</td>
<td>$40</td>
<td>Gas</td>
<td>No</td>
<td>Legitimate</td>
</tr>
<tr>
<td>6</td>
<td>$80</td>
<td>$80</td>
<td>Groceries</td>
<td>Yes</td>
<td>Legitimate</td>
</tr>
<tr>
<td>7</td>
<td>$140</td>
<td>$100</td>
<td>Retail</td>
<td>No</td>
<td>Legitimate</td>
</tr>
<tr>
<td>8</td>
<td>$140</td>
<td>$100</td>
<td>Retail</td>
<td>No</td>
<td>Fraudulent</td>
</tr>
</tbody>
</table>

### Validation Set

<table>
<thead>
<tr>
<th>Trans. ID</th>
<th>Charge Amount</th>
<th>Avg. Charge 6 months</th>
<th>Item</th>
<th>Same state as billing</th>
<th>Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>$100</td>
<td>$200</td>
<td>Electronics</td>
<td>Yes</td>
<td>?</td>
</tr>
<tr>
<td>102</td>
<td>$200</td>
<td>$100</td>
<td>Groceries</td>
<td>No</td>
<td>?</td>
</tr>
<tr>
<td>103</td>
<td>$1</td>
<td>$100</td>
<td>Gas</td>
<td>Yes</td>
<td>?</td>
</tr>
<tr>
<td>104</td>
<td>$30</td>
<td>$25</td>
<td>Restaurant</td>
<td>Yes</td>
<td>?</td>
</tr>
</tbody>
</table>
Goals

The trained model should assign new cases to the right category

It won’t be 100% accurate, but should be as close as possible

The model’s rules can be applied to new records as they come along

An automated, reliable way to predict the outcome
Classification Method: The Decision Tree

- A model to predict membership of cases or values of a dependent variable based on one or more predictor variables (Tan, Steinback, and Kumar 2004)
**Example: Credit Card Default**

<table>
<thead>
<tr>
<th>TID</th>
<th>Income</th>
<th>Debt</th>
<th>Owns/Rents</th>
<th>Outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>25k</td>
<td>35%</td>
<td>Owns</td>
<td>Default</td>
</tr>
<tr>
<td>2</td>
<td>35k</td>
<td>40%</td>
<td>Rent</td>
<td>Default</td>
</tr>
<tr>
<td>3</td>
<td>33k</td>
<td>15%</td>
<td>Owns</td>
<td>No default</td>
</tr>
<tr>
<td>4</td>
<td>28k</td>
<td>19%</td>
<td>Rents</td>
<td>Default</td>
</tr>
<tr>
<td>5</td>
<td>55k</td>
<td>30%</td>
<td>Owns</td>
<td>No default</td>
</tr>
<tr>
<td>6</td>
<td>48k</td>
<td>35%</td>
<td>Rent</td>
<td>Default</td>
</tr>
<tr>
<td>7</td>
<td>65k</td>
<td>17%</td>
<td>Owns</td>
<td>No default</td>
</tr>
<tr>
<td>8</td>
<td>85k</td>
<td>10%</td>
<td>Rents</td>
<td>No default</td>
</tr>
</tbody>
</table>

**Training Data**

We create the tree from a set of training data.

Each unique combination of predictors is associated with an outcome.

This set was “rigged” so that every combination is accounted for and has an outcome.
Example: Credit Card Default

Predictors

<table>
<thead>
<tr>
<th>TID</th>
<th>Income</th>
<th>Debt</th>
<th>Owns/Rents</th>
<th>Outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>25k</td>
<td>35%</td>
<td>Owns</td>
<td>Default</td>
</tr>
<tr>
<td>2</td>
<td>35k</td>
<td>40%</td>
<td>Rent</td>
<td>Default</td>
</tr>
<tr>
<td>3</td>
<td>33k</td>
<td>15%</td>
<td>Owns</td>
<td>No default</td>
</tr>
<tr>
<td>4</td>
<td>28k</td>
<td>19%</td>
<td>Rents</td>
<td>Default</td>
</tr>
<tr>
<td>5</td>
<td>55k</td>
<td>30%</td>
<td>Owns</td>
<td>No default</td>
</tr>
<tr>
<td>6</td>
<td>48k</td>
<td>35%</td>
<td>Rent</td>
<td>Default</td>
</tr>
<tr>
<td>7</td>
<td>65k</td>
<td>17%</td>
<td>Owns</td>
<td>No default</td>
</tr>
<tr>
<td>8</td>
<td>85k</td>
<td>10%</td>
<td>Rents</td>
<td>No default</td>
</tr>
</tbody>
</table>

Classification

Root node

Credit Approval

Child node

Income <40k

- Debt > 20%
  - Owns house
    - Default
  - Rents
    - Default

- Debt < 20%
  - Owns house
    - No Default
  - Rents
    - Default

Income >40k

- Debt > 20%
  - Owns house
    - No Default
  - Rents
    - No Default

- Debt < 20%
  - Owns house
    - No Default
  - Rents
    - No Default

Leaf node

Training Data
Same Data, Different Tree

### Training Data

<table>
<thead>
<tr>
<th>TID</th>
<th>Income</th>
<th>Debt</th>
<th>Owns/Rents</th>
<th>Outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>25k</td>
<td>35%</td>
<td>Owns</td>
<td>Default</td>
</tr>
<tr>
<td>2</td>
<td>35k</td>
<td>40%</td>
<td>Rent</td>
<td>Default</td>
</tr>
<tr>
<td>3</td>
<td>33k</td>
<td>15%</td>
<td>Owns</td>
<td>No default</td>
</tr>
<tr>
<td>4</td>
<td>28k</td>
<td>19%</td>
<td>Rents</td>
<td>Default</td>
</tr>
<tr>
<td>5</td>
<td>55k</td>
<td>30%</td>
<td>Owns</td>
<td>No default</td>
</tr>
<tr>
<td>6</td>
<td>48k</td>
<td>35%</td>
<td>Rent</td>
<td>Default</td>
</tr>
<tr>
<td>7</td>
<td>65k</td>
<td>17%</td>
<td>Owns</td>
<td>No default</td>
</tr>
<tr>
<td>8</td>
<td>85k</td>
<td>10%</td>
<td>Rents</td>
<td>No default</td>
</tr>
</tbody>
</table>

We just changed the order of the predictors.

We just changed the order of the predictors.
**Apply to new (validation) data**

### Validation Data

<table>
<thead>
<tr>
<th>TID</th>
<th>Income</th>
<th>Debt</th>
<th>Owns/Rents</th>
<th>Decision (Predicted)</th>
<th>Decision (Actual)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>80k</td>
<td>35%</td>
<td>Rent</td>
<td>Default</td>
<td>No Default</td>
</tr>
<tr>
<td>2</td>
<td>20k</td>
<td>40%</td>
<td>Owns</td>
<td>Default</td>
<td>Default</td>
</tr>
<tr>
<td>3</td>
<td>15k</td>
<td>15%</td>
<td>Owns</td>
<td>No Default</td>
<td>No Default</td>
</tr>
<tr>
<td>4</td>
<td>50k</td>
<td>19%</td>
<td>Rents</td>
<td>No Default</td>
<td>Default</td>
</tr>
<tr>
<td>5</td>
<td>35k</td>
<td>30%</td>
<td>Owns</td>
<td>Default</td>
<td>No Default</td>
</tr>
</tbody>
</table>

**How well did the decision tree do in predicting the outcome?**

When it’s “good enough,” we’ve got our model for future decisions.
In a real situation...

- The tree induction software has to deal with instances where...

  - The same set of predictors resulting in different outcomes
  - Multiple paths result in the same outcome
  - Not every combination of predictors is in the training set
Tree induction algorithms take large sets of data and compute the tree. Similar cases may have different outcomes. So probability of an outcome is computed.

For instance, you may find:
When income > 40k, debt < 20%, and the customers rents no default occurs 80% of the time.
How the induction algorithm works

1. Start with a single node with all training data.
2. Are samples all of the same classification?
   - Yes: Go to next node.
   - No: Proceed to the next step.
3. Are there predictor(s) that will split the data?
   - Yes: Partition node into child nodes according to predictor(s).
   - No: Proceed to the next step.
4. Are there more nodes (i.e., new child nodes)?
   - Yes: Go to next node.
   - No: DONE!
Start with root node

- There are both “defaults” and “no defaults” in the set
- So we need to look for predictors to split the data
First split: on income

• Income (with 40k as the cutoff point) is a factor that produces the greatest “separation”
  • More income, less default
• But there are also a combination of defaults and no defaults within each income group
• So look for another split
Second split: on debt

- **Debt is a factor**
  - (Income < 40, Debt < 20, Owns = No default)
  - but
    - (Income < 40, Debt < 20, Rents = Default)

- **But there are also a combination of defaults and no defaults within some debt groups**

- **So look for another split**
• Owns/Rents is a factor
• For some cases it doesn’t matter, but for some it does
• So you group similar branches
• …And we stop because we’re out of predictors!
How often does the tree make a correct prediction?

- **Error rate**: Percent of misclassified records out of the total records
- **Correct classification rate**: Percent of correctly classified records out of the total records

The Confusion Matrix:

<table>
<thead>
<tr>
<th>Observed outcome:</th>
<th>Predicted outcome:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Default</td>
</tr>
<tr>
<td>Default</td>
<td>600</td>
</tr>
<tr>
<td>No default</td>
<td>150</td>
</tr>
</tbody>
</table>

- Error rate = $(150+100)/1500 = 16.7%$
- Correct classification rate = $(1-16.7%) = 83.3%$
Which variables are important predictors?

There are statistics that show which predictor variable produces the greatest “separation” in the outcome variable

- Intuition: When including the predictor, you should see substantial improvement in classification accuracy.

https://www.researchgate.net/figure/48204326_fig2_Figure-2-a-Toy-classification-dataset-b-Resulting-decision-tree
Can we keep splitting as long as it can?

• You may get better classification accuracy

• But... the tree will become too complex
  – Difficult to interpret

• Problem of “Overfitting”:
  – The tree may have poor predictive performance, as it can exaggerate minor fluctuations in the data.

“A good model must not only fit the training data well but also accurately classify records it has never seen.”
Finding the Right Tree

The trade-off is between **accuracy** and **complexity**.

“Prune” the tree

We want to reduce error rate, as well as reduce complexity
Accuracy vs. Complexity

- **X-val Relative Error**: relative error rate against a baseline tree with 1 node.
  - When there is 1 node, the relative error rate is 1
  - When there are 3 nodes, the relative error rate is 0.82

- **cp (complexity parameter)**: incremental reduction in error when adding an additional node

The larger the tree is, the smaller the error is (i.e. higher accuracy)
Outcome cases are not 100% certain
- There are probabilities attached to each outcome in a node
- So let’s code “Default” as 1 and “No Default” as 0

How many leaf nodes are there?
Reading the R Decision Tree

Outcome cases are not 100% certain

- There are probabilities attached to each outcome in a node
- So let’s code “Default” as 1 and “No Default” as 0

So what is the chance that:

- A renter making more than $40,000 and debt more than 20% of income will default?
- A home owner making less than $40,000 and debt more than 20% of income will default?
Outcome cases are not 100% certain

- There are probabilities attached to each outcome in a node
- So let’s code “Default” as 1 and “No Default” as 0

Describe the characteristics of the most likely and least likely groups to default.
Summary

• Structure of a decision tree
  – Predictor and outcome variables
  – Q: Why do decision trees typically have categorical outcome variables?

• Error rate and correct classification rate

• Pros and cons of a complex tree
  – Overfitting
  – Understand the tradeoff between accuracy and complexity

• Interpret a decision tree: determine the probability of an event happening based on predictor variable values

• Exercise: Decision trees in R