Linear classifiers: Handling overfitting, categorical inputs, & multiple classes

STAT/CSE 416: Machine Learning
Emily Fox
University of Washington
April 24, 2018

Encoding categorical inputs
Categorical inputs

- Numeric inputs:
  - #awesome, age, salary,...
  - Intuitive when multiplied by coefficient
    - e.g., 1.5 #awesome
- Categorical inputs:
  - Gender
    - (Male, Female,...)
  - Country of birth
    - (Argentina, Brazil, USA,...)
  - Zipcode
    - (10005, 98195,...)

How do we multiply category by coefficient???
Must convert categorical inputs into numeric features

Encoding categories as numeric features

\[
\mathbf{x} = \begin{align*}
\text{Country of birth} \\
(\text{Argentina, Brazil, USA,...})
\end{align*}
\]

196 categories

\[
\begin{array}{cccc}
\mathbf{x} & h_1(x) & h_2(x) & \ldots & h_{195}(x) & h_{196}(x) \\
\text{Brazil} & & & & & \\
\text{Zimbabwe} & & & & & \\
\end{array}
\]

196 features

\[
\mathbf{x} = \begin{align*}
\text{Restaurant review} \\
(\text{Text data})
\end{align*}
\]

10,000 words in vocabulary

\[
\begin{array}{cccc}
\mathbf{x} & h_1(x) & h_2(x) & \ldots & h_{9999}(x) & h_{10000}(x) \\
\text{Restaurant review} & & & & & \\
\end{array}
\]

10,000 features
Multiclass classification
using 1 versus all

Input: $\mathbf{x}$
Image pixels

Output: $\mathbf{y}$
Object in image
Multiclass classification formulation

- C possible classes:
  - y can be 1, 2,..., C
- N datapoints:

<table>
<thead>
<tr>
<th>Data point</th>
<th>x[1]</th>
<th>x[2]</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>x₁,y₁</td>
<td>2</td>
<td>1</td>
<td>△</td>
</tr>
<tr>
<td>x₂,y₂</td>
<td>0</td>
<td>2</td>
<td>♡</td>
</tr>
<tr>
<td>x₃,y₃</td>
<td>3</td>
<td>3</td>
<td>○</td>
</tr>
<tr>
<td>x₄,y₄</td>
<td>4</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Learn:
\[
P(y=△ | x) \quad \triangleleft \quad P(y=♡ | x) \quad \triangleleft \quad P(y=○ | x)
\]

1 versus all:
Estimate \( \hat{P}(y=△ | x) \) using 2-class model

+1 class: points with \( y_i = △ \)
-1 class: points with \( y_i = ♡ \) OR ○

Train classifier: \( \hat{P}(y=+1 | x) \)

Predict: \( \hat{P}(y=△ | x_i) = \hat{P}(y=+1 | x_i) \)
**1 versus all:** simple multiclass classification using $C$ 2-class models

\[
\hat{P}(y=\triangle | x_i) = \]

\[
\hat{P}(y=\heartsuit | x_i) = \]

\[
\hat{P}(y=\bigcirc | x_i) = \]

**Multiclass training**

$\hat{P}_c(y=+1|x) = \text{estimate of 1 vs all model for each class}$

**Predict most likely class**

max\_prob = 0; $\hat{y} = 0$

For $c = 1, \ldots, C$:

If $\hat{P}_c(y=+1|x_i) > \text{max\_prob}$:

$\hat{y} = c$

max\_prob = $\hat{P}_c(y=+1|x_i)$

Input: $x_i$
Summary of overfitting in logistic regression, categorical inputs, and multiclass classification

What you can do now...

- Describe symptoms and effects of overfitting in classification
  - Identify when overfitting is happening
  - Relate large learned coefficients to overfitting
  - Describe the impact of overfitting on decision boundaries and predicted probabilities of linear classifiers
- Use regularization to mitigate overfitting
  - Motivate the form of L2 regularized logistic regression quality metric
  - Describe the use of L1 regularization to obtain sparse logistic regression solutions
  - Describe what happens to estimated coefficients as tuning parameter $\lambda$ is varied
  - Interpret coefficient path plot
- Use 1-hot encoding to represent categorical inputs
- Perform multiclass classification using the 1-versus-all approach
Decision Trees

Predicting potential loan defaults
What makes a loan risky?

I want to buy a new house!

Credit History ★★★★★
Income ★★★
Term ★★★★★★
Personal Info ★★★

Credit history explained

Did I pay previous loans on time?

Example: excellent, good, or fair
Income

What’s my income?

Example:
$80K per year

Loan terms

How soon do I need to pay the loan?

Example: 3 years, 5 years,...
Personal information

Age, reason for the loan, marital status,…

Example: Home loan for a married couple

Credit History ★★★★★
Income ★★★
Term ★★★★★★
Personal Info ★★★

Intelligent application

Loan Applications

Intelligent loan application review system

Safe ✓
Risky X
Risky X
Classifier review

Loan Application \[ \xrightarrow{\text{Classifier MODEL}} \] Classifier MODEL

Input: \( x_i \)  
Output: \( \hat{y} \)  
Predicted class

\( \hat{y}_i = +1 \) \hspace{1cm} Safe
\( \hat{y}_i = -1 \) \hspace{1cm} Risky

This module ... decision trees

Start

Credit?  
excellent  
poor

Term?  
fair  
3 years  
5 years  
Risky  
Safe

Income?  
high  
Low

Term?  
3 years  
5 years  
Risky  
Safe
Scoring a loan application

\[ x_i = (\text{Credit} = \text{poor}, \text{Income} = \text{high}, \text{Term} = 5 \text{ years}) \]

Decision tree learning task
Decision tree learning problem

Training data: \( N \) observations \((x_i, y_i)\)

<table>
<thead>
<tr>
<th>Credit</th>
<th>Term</th>
<th>Income</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>excellent</td>
<td>3 yrs</td>
<td>high</td>
<td>safe</td>
</tr>
<tr>
<td>fair</td>
<td>5 yrs</td>
<td>low</td>
<td>risky</td>
</tr>
<tr>
<td>fair</td>
<td>3 yrs</td>
<td>high</td>
<td>safe</td>
</tr>
<tr>
<td>poor</td>
<td>5 yrs</td>
<td>high</td>
<td>risky</td>
</tr>
<tr>
<td>excellent</td>
<td>3 yrs</td>
<td>low</td>
<td>risky</td>
</tr>
<tr>
<td>fair</td>
<td>5 yrs</td>
<td>low</td>
<td>safe</td>
</tr>
<tr>
<td>poor</td>
<td>3 yrs</td>
<td>high</td>
<td>risky</td>
</tr>
<tr>
<td>poor</td>
<td>5 yrs</td>
<td>low</td>
<td>safe</td>
</tr>
<tr>
<td>fair</td>
<td>3 yrs</td>
<td>high</td>
<td>safe</td>
</tr>
</tbody>
</table>

Quality metric: Classification error

- Error measures fraction of mistakes

\[
\text{Error} = \frac{\text{# incorrect predictions}}{\text{# examples}}
\]

- Best possible value: 0.0
- Worst possible value: 1.0
How do we find the best tree?

Exponentially large number of possible trees makes decision tree learning hard!

Learning the smallest decision tree is an NP-hard problem [Hyafil & Rivest ’76]

Greedy decision tree learning
Our training data table

Assume $N = 40$, 3 features

<table>
<thead>
<tr>
<th>Credit</th>
<th>Term</th>
<th>Income</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>excellent</td>
<td>3 yrs</td>
<td>high</td>
<td>safe</td>
</tr>
<tr>
<td>fair</td>
<td>5 yrs</td>
<td>low</td>
<td>risky</td>
</tr>
<tr>
<td>fair</td>
<td>3 yrs</td>
<td>high</td>
<td>safe</td>
</tr>
<tr>
<td>poor</td>
<td>5 yrs</td>
<td>high</td>
<td>risky</td>
</tr>
<tr>
<td>excellent</td>
<td>3 yrs</td>
<td>low</td>
<td>risky</td>
</tr>
<tr>
<td>fair</td>
<td>5 yrs</td>
<td>low</td>
<td>safe</td>
</tr>
<tr>
<td>poor</td>
<td>3 yrs</td>
<td>high</td>
<td>risky</td>
</tr>
<tr>
<td>poor</td>
<td>5 yrs</td>
<td>low</td>
<td>safe</td>
</tr>
<tr>
<td>fair</td>
<td>3 yrs</td>
<td>high</td>
<td>safe</td>
</tr>
</tbody>
</table>

Start with all the data

Loan status: Safe Risky

# of Safe loans

# of Risky loans

N = 40 examples
Compact visual notation: Root node

Loan status: Safe Risky

Root

# of Risky loans

# of Safe loans

N = 40 examples

Decision stump: Single level tree

Loan status: Safe Risky

Root

Split on Credit

Credit?

excellent

fair

poor

Subset of data with Credit = excellent
Subset of data with Credit = fair
Subset of data with Credit = poor
Visual notation: Intermediate nodes

Loan status:
Safe Risky

Root
22 18

Credit?

excellent
9 0

fair
9 4

poor
4 14

Intermediate nodes

Making predictions with a decision stump

Loan status:
Safe Risky

root
22 18

credit?

excellent
9 0

fair
9 4

poor
4 14

Safe Safe Risky

For each intermediate node, set \( \hat{y} = \text{majority value} \)
Selecting best feature to split on

How do we learn a decision stump?

Loan status:
Safe Risky

Root
22 18

Find the “best” feature to split on!

Credit?

excellent
9 0

fair
9 4

poor
4 14
How do we select the best feature?

**Choice 1: Split on Credit**

- **Loan status:** Safe Risky
- **Root:**
  - **Credit?**
    - excellent: 9 0
    - fair: 9 4
    - poor: 4 14

**Choice 2: Split on Term**

- **Loan status:** Safe Risky
- **Root:**
  - **Term?**
    - 3 years: 16 4
    - 5 years: 6 14

---

How do we measure effectiveness of a split?

**Loan status:** Safe Risky

- **Root:**
  - **Credit?**
    - excellent: 9 0
    - fair: 9 4
    - poor: 4 14

**Idea:** Calculate classification error of this decision stump

\[
\text{Error} = \frac{\text{# mistakes}}{\text{# data points}}
\]
Calculating classification error

- **Step 1:** \( \hat{y} \) = class of majority of data in node
- **Step 2:** Calculate classification error of predicting \( \hat{y} \) for this data

\[
\text{Error} = \frac{18}{22} + \frac{18}{18} = 0.45
\]

Choice 1: Split on Credit history?

**Choice 1: Split on Credit**

Does a split on Credit reduce classification error below 0.45?
Split on **Credit**: Classification error

**Choice 1:** Split on Credit

<table>
<thead>
<tr>
<th>Loan status:</th>
<th>Safe</th>
<th>Risky</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root</td>
<td>22</td>
<td>18</td>
</tr>
</tbody>
</table>

Credit?

<table>
<thead>
<tr>
<th>Credit</th>
<th>Safe</th>
<th>Risky</th>
</tr>
</thead>
<tbody>
<tr>
<td>excellent</td>
<td>9</td>
<td>0</td>
</tr>
<tr>
<td>fair</td>
<td>9</td>
<td>4</td>
</tr>
<tr>
<td>poor</td>
<td>4</td>
<td>14</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Loan status</th>
<th>Safe</th>
<th>Risky</th>
</tr>
</thead>
<tbody>
<tr>
<td>Safe</td>
<td>0</td>
<td>mistakes</td>
</tr>
<tr>
<td>Safe</td>
<td>4</td>
<td>mistakes</td>
</tr>
<tr>
<td>Risky</td>
<td>4</td>
<td>mistakes</td>
</tr>
</tbody>
</table>

Error = $igstar$ 

<table>
<thead>
<tr>
<th>Tree</th>
<th>Classification error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(root)</td>
<td>0.45</td>
</tr>
<tr>
<td>Split on credit</td>
<td>0.2</td>
</tr>
</tbody>
</table>

**Choice 2:** Split on **Term**?

**Choice 2:** Split on Term

<table>
<thead>
<tr>
<th>Loan status:</th>
<th>Safe</th>
<th>Risky</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root</td>
<td>22</td>
<td>18</td>
</tr>
</tbody>
</table>

Term?

<table>
<thead>
<tr>
<th>Term</th>
<th>Safe</th>
<th>Risky</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 years</td>
<td>16</td>
<td>4</td>
</tr>
<tr>
<td>5 years</td>
<td>6</td>
<td>14</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Loan status</th>
<th>Safe</th>
<th>Risky</th>
</tr>
</thead>
<tbody>
<tr>
<td>Safe</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Risky</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Evaluating the split on Term

Choice 2: Split on Term

<table>
<thead>
<tr>
<th>Loan status:</th>
<th>Safe</th>
<th>Risky</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root</td>
<td>22</td>
<td>18</td>
</tr>
</tbody>
</table>

Term?

<table>
<thead>
<tr>
<th>Term</th>
<th>Safe</th>
<th>Risky</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 years</td>
<td>16</td>
<td>4</td>
</tr>
<tr>
<td>5 years</td>
<td>6</td>
<td>14</td>
</tr>
</tbody>
</table>

Choice 1 vs Choice 2: Comparing split on Credit vs Term

Choice 1: Split on Credit

<table>
<thead>
<tr>
<th>Loan status:</th>
<th>Safe</th>
<th>Risky</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root</td>
<td>22</td>
<td>18</td>
</tr>
</tbody>
</table>

Credit?

<table>
<thead>
<tr>
<th>Credit</th>
<th>excellent</th>
<th>fair</th>
<th>poor</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Choice 2: Split on Term

<table>
<thead>
<tr>
<th>Loan status:</th>
<th>Safe</th>
<th>Risky</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root</td>
<td>22</td>
<td>18</td>
</tr>
</tbody>
</table>

Term?

<table>
<thead>
<tr>
<th>Term</th>
<th>Safe</th>
<th>Risky</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 years</td>
<td>16</td>
<td>4</td>
</tr>
<tr>
<td>5 years</td>
<td>6</td>
<td>14</td>
</tr>
</tbody>
</table>

Error = ________

Tree | Classification error
-----|----------------------
(root) | 0.45
Split on credit | 0.2
Split on term | 0.25

WINNER
Feature split selection algorithm

• Given a subset of data $M$ (a node in a tree)
• For each feature $h_i(x)$:
  1. Split data of $M$ according to feature $h_i(x)$
  2. Compute classification error of split
• Chose feature $h^*(x)$ with lowest classification error

Recursion & Stopping conditions
We’ve learned a decision stump, what next?

Loan status: Safe Risky

Root

Credit?

excellent 9 0

fair 9 4

poor 4 14

All data points are Safe ➔ nothing else to do with this subset of data

Leaf node

Build decision stump with subset of data where Credit = fair

Build decision stump with subset of data where Credit = poor

Tree learning = Recursive stump learning

Loan status: Safe Risky

Root

Credit?

excellent 9 0

fair 9 4

poor 4 14

Safe
**Second level**

Loan status:
- Safe
- Risky

Root:
- 22
- 18

Credit?
- excellent 9
- fair 9
- poor 4

- Safe
- Risky

Term?
- 3 years 0
- 5 years 9
- high 4
- low 0

- Risky
- Safe

Build another stump these data points

**Final decision tree**

Loan status:
- Safe
- Risky

Root:
- 22
- 18

Credit?
- excellent 9
- fair 9
- poor 4

- Safe
- Risky

Term?
- 3 years 0
- 5 years 9
- high 4
- low 0

- Risky
- Safe

Income?
- high 4
- low 0

- Risky
- Safe
Simple greedy decision tree learning

- Pick best feature to split on
- Learn decision stump with this split
- For each leaf of decision stump, recurse

When do we stop???

Stopping condition 1: All data agrees on y

- All data in these nodes have same y value
  - Nothing to do

Root
- Credit?
  - Excellent
    - Safe
    - 9 0
  - Fair
    - Safe
    - 9 4
  - Term?
    - 5 years
      - Safe
      - 9 0
    - 3 years
      - Risky
      - 4 0

- Income?
  - Poor
    - 4 14
  - Income?
    - High
      - 4 14
    - Low
      - 0 9

- Term?
  - Risky
    - 0 2
  - Safe
    - 5 3

Stopping condition 2: Already split on all features

Already split on all possible features
→ Nothing to do

Greedy decision tree learning

• **Step 1:** Start with an empty tree

• **Step 2:** Select a feature to split data

• For each split of the tree:
  • **Step 3:** If nothing more to, make predictions
  • **Step 4:** Otherwise, go to **Step 2** & continue (recurse) on this split
Is this a good idea?

Proposed stopping condition 3:
Stop if no split reduces the classification error

Stopping condition 3:
Don’t stop if error doesn’t decrease???

\[
y = x[1] \text{xor} x[2]
\]

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>False</td>
</tr>
</tbody>
</table>

\[
\text{y values: } \begin{cases} \text{True} & 2 \\ \text{False} & 2 \end{cases}
\]

\[
\text{Root Error} = \frac{2 + 2}{2 + 2} = 1
\]

<table>
<thead>
<tr>
<th>Tree</th>
<th>Classification error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(root)</td>
<td>0.5</td>
</tr>
</tbody>
</table>
Consider split on $x[1]$

$y = x[1] \text{xor} x[2]$

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>False</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>False</td>
</tr>
</tbody>
</table>

$y$ values

<table>
<thead>
<tr>
<th>True</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Root

| 2 | 2 |

Error = ________

=________

<table>
<thead>
<tr>
<th>Tree</th>
<th>Classification error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(root)</td>
<td>0.5</td>
</tr>
<tr>
<td>Split on $x[1]$</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Consider split on $x[2]$

$y = x[1] \text{xor} x[2]$

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>False</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>False</td>
</tr>
</tbody>
</table>

$y$ values

<table>
<thead>
<tr>
<th>True</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Root

| 2 | 2 |

<table>
<thead>
<tr>
<th>$x[2]$</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
</tr>
<tr>
<td>False</td>
</tr>
</tbody>
</table>

Neither features improve training error... Stop now???

Error = $\frac{1+1}{2+2} = 0.5$

<table>
<thead>
<tr>
<th>Tree</th>
<th>Classification error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(root)</td>
<td>0.5</td>
</tr>
<tr>
<td>Split on $x[1]$</td>
<td>0.5</td>
</tr>
<tr>
<td>Split on $x[2]$</td>
<td>0.5</td>
</tr>
</tbody>
</table>
Final tree with stopping condition 3

\[ y = x[1] \text{ xor } x[2] \]

<table>
<thead>
<tr>
<th>(x[1])</th>
<th>(x[2])</th>
<th>(y)</th>
</tr>
</thead>
<tbody>
<tr>
<td>False</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>False</td>
</tr>
</tbody>
</table>

Tree

<table>
<thead>
<tr>
<th>Classification error</th>
</tr>
</thead>
<tbody>
<tr>
<td>with stopping condition 3</td>
</tr>
</tbody>
</table>

Without stopping condition 3

Condition 3 (stopping when training error doesn't improve) is not recommended!

\[ y = x[1] \text{ xor } x[2] \]

<table>
<thead>
<tr>
<th>(x[1])</th>
<th>(x[2])</th>
<th>(y)</th>
</tr>
</thead>
<tbody>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>False</td>
</tr>
</tbody>
</table>

Tree

<table>
<thead>
<tr>
<th>Classification error</th>
</tr>
</thead>
<tbody>
<tr>
<td>with stopping condition 3</td>
</tr>
<tr>
<td>without stopping condition 3</td>
</tr>
</tbody>
</table>
Decision tree learning: 
*Real valued features*

How do we use real values inputs?

<table>
<thead>
<tr>
<th>Income</th>
<th>Credit</th>
<th>Term</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>$105 K</td>
<td>excellent</td>
<td>3 yrs</td>
<td>Safe</td>
</tr>
<tr>
<td>$112 K</td>
<td>good</td>
<td>5 yrs</td>
<td>Risky</td>
</tr>
<tr>
<td>$73 K</td>
<td>fair</td>
<td>3 yrs</td>
<td>Safe</td>
</tr>
<tr>
<td>$69 K</td>
<td>excellent</td>
<td>5 yrs</td>
<td>Safe</td>
</tr>
<tr>
<td>$217 K</td>
<td>excellent</td>
<td>3 yrs</td>
<td>Risky</td>
</tr>
<tr>
<td>$120 K</td>
<td>good</td>
<td>5 yrs</td>
<td>Safe</td>
</tr>
<tr>
<td>$64 K</td>
<td>fair</td>
<td>3 yrs</td>
<td>Risky</td>
</tr>
<tr>
<td>$340 K</td>
<td>excellent</td>
<td>5 yrs</td>
<td>Safe</td>
</tr>
<tr>
<td>$60 K</td>
<td>good</td>
<td>3 yrs</td>
<td>Risky</td>
</tr>
</tbody>
</table>
Threshold split

Loan status:
Safe  Risky

Split on the feature Income

Income?

< $60K
8 13

>= $60K
14 5

Subset of data with Income >= $60K

Finding the best threshold split

Infinite possible values of t

Income = t*

Income < t*
Income >= t*

Income

$10K

Safe

Risky

$120K
Consider a threshold between points

Same classification error for any threshold split between $v_A$ and $v_B$

Only need to consider mid-points

Finite number of splits to consider
Threshold split selection algorithm

- **Step 1:** Sort the values of a feature \( h_j(x) \):
  
  Let \( \{v_1, v_2, v_3, \ldots, v_N\} \) denote sorted values.

- **Step 2:**
  - For \( i = 1 \ldots N-1 \)
    - Consider split \( t_i = \frac{(v_i + v_{i+1})}{2} \)
    - Compute classification error for threshold split \( h_j(x) \geq t_i \)
  - Chose the \( t^* \) with the lowest classification error

Visualizing the threshold split

Threshold split is the line \( \text{Age} = 38 \)
Split on Age $\geq 38$

Depth 2: Split on Income $\geq $60K

Threshold split is the line $\text{Income} = 60K$
Each split partitions the 2-D space

Decision trees vs logistic regression: 
*Example*
Logistic regression

<table>
<thead>
<tr>
<th>Feature</th>
<th>Value</th>
<th>Weight Learned</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h_0(x)$</td>
<td>1</td>
<td>0.22</td>
</tr>
<tr>
<td>$h_1(x)$</td>
<td>$x[1]$</td>
<td>1.12</td>
</tr>
<tr>
<td>$h_2(x)$</td>
<td>$x[2]$</td>
<td>-1.07</td>
</tr>
</tbody>
</table>

Depth 1: Split on $x[1]$

- $y$ values
  - $-$
  - $+$

- $x[1] < -0.07$
  - 13
  - 3
- $x[1] \geq -0.07$
  - 4
  - 11
Depth 2

Threshold split caveat

For threshold splits, same feature can be used multiple times
Decision boundaries

Comparing decision boundaries

Logistic Regression

Degree 1 features

Degree 2 features

Degree 6 features

Decision Tree

Depth 1

Depth 2

Depth 10

Degree 2 features
Predicting probabilities with decision trees

Loan status:
Safe  Risky

Root
18  12

Credit?

excellent
9  2

fair
6  9

poor
3  1

Depth 1 probabilities

\[ P(y = \text{Safe} \mid x) \]
\[ = \frac{3}{3 + 1} = 0.75 \]
Depth 2 probabilities

Y values
- +
root 18 13

X1

X1 < -0.07
13 3
X1 >= -0.07
4 11

X2

X2 < 1.55
7 0
X2 >= 1.55
6 3
X2 >= 1.55
1 11
X2 >= 1.55
3 0

Comparison with logistic regression

Logistic Regression (Degree 2)

Decision Trees (Depth 2)
Summary of decision trees

What you can do now

• Define a decision tree classifier
• Interpret the output of a decision trees
• Learn a decision tree classifier using greedy algorithm
• Traverse a decision tree to make predictions
  – Majority class predictions