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

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

196 categories

1-hot encoding

\[
\begin{array}{c|ccccc}
& h_1(x) & h_2(x) & \ldots & h_{195}(x) & h_{196}(x) \\
\hline
\text{Brazil} & 0 & 1 & 0 & 0 & 0 \\
\text{Zimbabwe} & 0 & 0 & 0 & 0 & 1 \\
\end{array}
\]

196 features

\[ x = \begin{array}{c}
\text{Restaurant review} \\
(\text{Text data})
\end{array} \]

10,000 words in vocabulary

Bag of words

\[
\begin{array}{c|ccccc}
& h_1(x) & h_2(x) & \ldots & h_{9999}(x) & h_{10000}(x) \\
\hline
\text{Restaurant review} & 2 & 0 & 0 & 0 & 3 \\
\end{array}
\]

10,000 features
Multiclass classification using 1 versus all

Multiclass classification

Input: $x$
Image pixels

Output: $y$
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>$x_1,y_1$</td>
<td>2</td>
<td>1</td>
<td>△</td>
</tr>
<tr>
<td>$x_2,y_2$</td>
<td>0</td>
<td>2</td>
<td>❤</td>
</tr>
<tr>
<td>$x_3,y_3$</td>
<td>3</td>
<td>3</td>
<td>○</td>
</tr>
<tr>
<td>$x_4,y_4$</td>
<td>4</td>
<td>1</td>
<td>○</td>
</tr>
</tbody>
</table>

Learn:

- $P(y=\triangle | x)$
- $P(y=\heartsuit | x)$
- $P(y=\bigcirc | x)$

1 versus all:

Estimate $\hat{P}(y=\triangle | x)$ using 2-class model

- $+1$ class: points with $y_i=\triangle$
- $-1$ class: points with $y_i=\heartsuit$ OR $\bigcirc$

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

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

\[
\hat{P}(y=\triangle | x_i) = \hat{P}_D(y=-1 | x_i, w_D) \\
\hat{P}(y=\heartsuit | x_i) = \hat{P}_D(y=+1 | x_i, w_D) \\
\hat{P}(y=\bigcirc | x_i) = \hat{P}_E(y=0 | x_i, w_E)
\]

©2018 Emily Fox

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

©2018 Emily Fox
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 ✘
Risky ✘
Classifier review

Loan Application → Classifier MODEL

Input: $x_i$

Output: $\hat{y}$
Predicted class

$\hat{y}_i = +1$
Safe

$\hat{y}_i = -1$
Risky

This module ... decision trees

Start

- Credit?
  - excellent → Safe
  - poor → Credit?
    - fair → Term?
      - 3 years → Risky
      - 5 years → Safe
    - Credit?
      - fair → Term?
        - 3 years → Risky
        - 5 years → Safe
      - Credit?
        - poor → Income?
          - high → Term?
            - 3 years → Risky
            - 5 years → Safe
          - low → Term?
            - 3 years → Risky
            - 5 years → 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>low</td>
<td>safe</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

(All data)

# 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?

Subset of data with Credit = excellent

Subset of data with Credit = fair

Subset of data with Credit = poor

©2018 Emily Fox
Visual notation: Intermediate nodes

Loan status: Safe Risky

Root

Credit?

excellent
fair
poor

Loan status: Safe Risky

Making predictions with a decision stump

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

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**

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

**Choice 2: Split on Term**

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

How do we measure effectiveness of a split?

**Loan status:** Safe  Safe  Risky

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

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

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

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

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

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

Choice 1: Split on **Credit** history?

**Choice 1:** Split on **Credit**

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

- Loan status: Safe Risky
- Root: 22 correct 18 mistakes
- \( \hat{y} = \text{majority class Safe} \)
- Error: 0.45
**Split on Credit: Classification error**

**Choice 1: Split on Credit**

Loan status: Safe Risky

Root 22 18

Credit?

excellent 9 0

Safe

0 mistakes

fair 9 4

Safe

4 mistakes

poor 4 14

Risky

4 mistakes

Error = \( \frac{0 \cdot 4 + 4}{40} \)

= 0.2

<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**

Loan status: Safe Risky

Root 22 18

Term?

3 years 16 4

Safe

4 mistakes

5 years 6 14

Risky

4 mistakes
Evaluating the split on Term

Choice 2: Split on Term

\[
\text{Error} = \frac{4 + 6}{40} = 0.25
\]

<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>0.20</td>
</tr>
<tr>
<td>Split on term</td>
<td>0.25</td>
</tr>
</tbody>
</table>

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

Choice 1: Split on Credit

Choice 2: Split on Term

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

Safe

Leaf node

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

Tree learning = Recursive stump learning

Loan status: Safe Risky

Root

Credit?

excellent 9 0

fair 9 4

poor 4 14

Safe

Build decision stump with subset of data where Credit = fair

Build decision stump with subset of data where Credit = poor
Second level

Loan status:
Safe Risky

Root
22 18

Credit?

excellent
9 0
Safe

fair
9 4

Term?

3 years
0 4
Risky

5 years
9 0
Safe

poor
4 14

high
4 5
Risky

low
0 9

Income?

Build another stump these data points

Final decision tree

Loan status:
Safe Risky

Root
22 18

Credit?

excellent
9 0
Safe

Fair
9 4

Term?

3 years
0 4
Risky

5 years
9 0
Safe

poor
4 14

high
4 5

low
0 9

Term?

3 years
0 2
Risky

5 years
4 3
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
**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

Pick feature split leading to lowest classification error

Stopping conditions 1 & 2

Recursion
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>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>

\[ y \text{ values } \quad \text{True } \quad \text{False} \]

Root

\[ \begin{array}{cc}
2 & 2 \\
\end{array} \]

\[ y = \text{safe} \]

\[ \text{Error} = \frac{2}{2+2} = 0.5 \]

<table>
<thead>
<tr>
<th>Tree</th>
<th>Classification error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(root)</td>
<td>0.5</td>
</tr>
</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>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>

$y$ values  
True False

Root
2 2

Error = \frac{1 + 1}{4} = 0.5

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>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>

$y$ values  
True False

Root
2 2

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

Neither features improve training error... Stop now???
Final tree with stopping condition 3

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

<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>

\( y \) values:
- True
- False

Root:
- 2
- 2

Predict True

Tree | Classification error
---|----------------------
with stopping condition 3 | 0.5

Without stopping condition 3

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

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

<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>

\( y \) values:
- True
- False

Root:
- 2
- 2

\( x[1] \):
- True
- 1
- 1

\( x[2] \):
- False
- 0
- 1

Tree | Classification error
---|----------------------
with stopping condition 3 | 0.5
without stopping condition 3 | 0
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* threshold to choose

Income < t*

Income >= t*

Safe
Risky

Income

$10K

$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 \( t_i = (v_i + v_{i+1}) / 2 \)
    - 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 $\geq 38$

Age & Income
\begin{tabular}{|c|c|}
\hline
0 & $0K$
\hline
10 & $10K$
\hline
20 & $20K$
\hline
30 & $30K$
\hline
40 & $40K$
\hline
\end{tabular}

\begin{itemize}
\item Split on Age $\geq 38$
\item Predict Risky
\item Predict Safe
\end{itemize}

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

Income & Age
\begin{tabular}{|c|c|}
\hline
$0K$ & 0
\hline
$10K$ & 10
\hline
$20K$ & 20
\hline
$30K$ & 30
\hline
$40K$ & 40
\hline
\end{tabular}

\begin{itemize}
\item Threshold split is the line $\text{Income} = 60K$
\end{itemize}
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 $x[1]$
Depth 2

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

Comparing decision boundaries
Predicting probabilities with decision trees

Loan status:
Safe
Risky

Root
18 12

Credit?

excellent
9 2
Safe

fair
6 9
Risky

poor
3 1
Safe

P(y = Safe | x) = \frac{3}{3 + 1} = 0.75

Depth 1 probabilities
Depth 2 probabilities

Comparison with logistic regression
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