CSE/STAT 416
Decision Trees

Vinitra Swamy
University of Washington
July 13, 2020

* Content built on the work of Hunter Schafer and Emily Fox.
Logistics

- Reading is optional
- If you have a question, there is a high chance somebody else in the class the same question too
- Homework 3 –
  - Extension until Friday
  - Concept question #11 has been removed

Today:
- Decision Trees
Idea: Estimate probabilities $\hat{P}(y|x)$ and use those for prediction

Probability Classifier

Input $x$: Sentence from review

- Estimate class probability $\hat{P}(y = +1|x)$
- If $\hat{P}(y = +1|x) > 0.5$:
  - $\hat{y} = +1$
- Else:
  - $\hat{y} = -1$

Notes:

- Estimating the probability improves interpretability
Interpreting Score

\[
Score(x_i) = w^T h(x_i)
\]

For a given input \(x_i\), the score is computed as the dot product of the weight vector \(w\) and the hypothesis function \(h(x_i)\).

\[
\hat{y}_i = \begin{cases} 
-1 & \text{Very sure} \\
0 & \text{Not sure if} \\
+1 & \text{Very sure}
\end{cases}
\]

- \( \hat{y}_i = -1 \) indicates very sure prediction of class -1.
- \( \hat{y}_i = 0 \) indicates uncertainty (not sure if class is -1 or 1).
- \( \hat{y}_i = +1 \) indicates very sure prediction of class 1.

The probability of the prediction being correct is given by:

\[
\hat{P}(y_i = +1 | x_i) = \begin{cases} 
0 & \hat{y}_i = -1 \\
0.5 & \hat{y}_i = 0 \\
1 & \hat{y}_i = +1
\end{cases}
\]
\[ \hat{P}(y = +1|x, \hat{w}) = \text{sigmoid} \left( \hat{w}^T h(x) \right) = \frac{1}{1 + e^{-\hat{w}^T h(x)}} \]
Naïve Bayes
Idea: Naïve Bayes

\[ x = \text{“The sushi & everything else was awesome!”} \]

\[ P(y = +1 \mid x = \text{“The sushi & everything else was awesome!”})? \]

\[ P(y = -1 \mid x = \text{“The sushi & everything else was awesome!”})? \]

**Idea:** Select the class with the highest probability!

Bayes Rule: \( P(y = +1 \mid x) = \frac{P(x \mid y = +1)P(y = +1)}{P(x)} \)

\[ \frac{P\left( \text{“The sushi & everything else was awesome!”} \mid +1 \right) P(+1)}{P(\text{“The sushi & everything else was awesome!”})} \]

Since we’re just trying to find out which class has the greater probability, we can discard the divisor.
Idea: Select the class with the highest probability!

Problem: We have not seen the sentence before.

Assumption: Words are independent from each other.

\[ x = \text{“The sushi & everything else was awesome!”} \]

\[
P(\text{“The sushi & everything else was awesome!”} | +1) P(+1) \]
\[
P(\text{“The sushi & everything else was awesome!”})
\]

\[
P(\text{“The sushi & everything else was awesome!”} | + 1)
= P(\text{The} | +1) * P(\text{sushi} | +1) * P(\& | +1) * P(\text{everything} | +1) * P(\text{else} | +1) * P(\text{was} | +1) * P(\text{awesome} | +1)
\]

\[ P(\text{“awesome”} | + 1)? \]
If a feature is missing in a class everything becomes zero.

\[ P("The sushi & everything else was awesome!" \mid +1) \]
\[ = P(The \mid +1) \times P(sushi \mid +1) \times P(& \mid +1) \times P(everything \mid +1) \]
\[ \times P(else \mid +1) \times P(was \mid +1) \times P(awesome \mid +1) \]

Solutions?

- Take the log (product becomes a sum: linear classifier)
- Laplacian Smoothing (adding a constant to avoid multiplying by zero)
Naïve Bayes vs Logistic Regression

Logistic Regression:
\[ P(y = +1|x, w) = \frac{1}{1 + e^{-w^T h(x)}} \]

Naïve Bayes:
\[ P(y|x_1, x_2, \ldots, x_d) = \prod_{j=1}^{d} P(x_j|y) P(y) \]
Naïve Bayes vs Logistic Regression

**Generative vs Discriminative Classifiers**

**Generative:** defines a model for generating $x$ (e.g. Naïve Bayes)

**Discriminative:** only cares about defining and optimizing a decision boundary (e.g. Logistic Regression)
Properties

• Linear Classifier for discrete values
• Continuous Variables - Gaussian Naïve Bayes
• **Gaussian Naïve Bayes is equivalent to a Logistic Regression!**
• Naïve Bayes very efficient for discrete data: only counts
• Naïve Bayes works well for big datasets
Multiclass Classification

- Everything works with multiple classes!

Input: \(x\)

Image pixels

Output: \(y\)

Object in image

Take max of:

\[P(\text{Labrador retriever}|x), P(\text{golden retriever}|x), P(\text{redbone}|x), P(\text{bloodhound}|x), P(\text{Rhodesian ridgeback}|x)\]
Decision Trees
How do we make decisions?

A line might not always support our decisions.
What makes a loan risky?

I want to buy a new house!

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

Loan Application
Credit history explained

Did I pay previous loans on time?

Example: excellent, good, or fair

Credit History ★★★★★
Income ★★★
Term ★★★★★
Personal Info ★★★
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,...
Age, reason for the loan, marital status,...

**Example:** Home loan for a married couple
Intelligent application
Classifier review

\[ \hat{y}_i = +1 \]  
Safe

\[ \hat{y}_i = -1 \]  
Risky

Input: \( x_i \)

Loan Application

Classifier MODEL

Output: \( \hat{y} \)
Predicted class
Setup

Data (N observations, 3 features)

Evaluation: classification error

Many possible decisions: number of trees grows exponentially!
Decision Trees

- **internal node**: testing a feature
- **branch**: splits into possible values of a feature
- **leaf**: final decision (the class value)
• Grow the trees using a greedy approach
• What do we need?
Visual Notation

Loan status: **Safe**  **Risky**

Root

<table>
<thead>
<tr>
<th></th>
<th>Safe</th>
<th>Risky</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>N</strong></td>
<td>6</td>
<td>3</td>
</tr>
</tbody>
</table>

# of Safe loans

# of Risky loans

N = 9 examples
Decision stump: 1 level

Loan status: Safe Risky

Split on Credit

<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>safe</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>

Subset of data with Credit = excellent
Subset of data with Credit = fair
Subset of data with Credit = poor
Making predictions

For each intermediate node, set $\hat{y} =$ majority value

Loan status: Safe  Risky

Root
6 3

credit?

excellent
2 0

fair
3 1

poor
1 2

Safe

Safe

Risky
How do we select the best feature?

* Select the split with lowest classification error

**Choice 1: Split on Credit**

- **Loan status:** Safe, Risky
  - **Root**
    - Credit?
      - **excellent**
        - 2 0
      - **fair**
        - 3 1
      - **poor**
        - 1 2

**Choice 2: Split on Term**

- **Loan status:** Safe, Risky
  - **Root**
    - Term?
      - **3 years**
      - **5 years**
Calculate the node values.

<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>safe</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>

Choice 2: Split on Term

Loan status:
Safe  Risky

Root
6 3

Term?

3 years
5 years

pollev.com/cse416
How do we select the best feature?

* Select the split with lowest classification error

**Choice 1: Split on Credit**

- **Loan status:** Safe Risky
- **Root:** 6 3
- **Credit?**
  - excellent: 2 0
  - fair: 3 1
  - poor: 1 2

**Choice 2: Split on Term**

- **Loan status:** Safe Risky
- **Root:** 6 3
- **Term?**
  - 3 years: 4 1
  - 5 years: 2 2
How do we measure effectiveness of a split?

**Loan status:**
- Safe
- Risky

**Root:**
- 6 mistakes
- 3 data points

**Credit?**

- **excellent:**
  - 2 mistakes
  - 0 data points
- **fair:**
  - 3 mistakes
  - 1 data point
- **poor:**
  - 1 mistake
  - 2 data points

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

**Error:**
\[
\frac{\text{# mistakes}}{\text{# data points}}
\]
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

<table>
<thead>
<tr>
<th>Tree</th>
<th>Classification error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(root)</td>
<td>0.33</td>
</tr>
</tbody>
</table>
Choice 1: Split on Credit history?

Does a split on Credit reduce classification error below 0.33?

Choice 1: Split on Credit

Loan status: Safe Risky

Root

6 3

Credit?

excellent

2 0

fair

3 1

poor

1 2
Choice 1: Split on Credit

Loan status:
Safe Risky

<table>
<thead>
<tr>
<th>Credit</th>
<th>Safe</th>
<th>Risky</th>
</tr>
</thead>
<tbody>
<tr>
<td>excellent</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>fair</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>poor</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Error = 

<table>
<thead>
<tr>
<th>Tree</th>
<th>Classification error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(root)</td>
<td>0.33</td>
</tr>
<tr>
<td>Split on credit</td>
<td>0.22</td>
</tr>
</tbody>
</table>
Choice 2: Split on Term

Loan status:
Safe  Risky

Root
6 3

Term?

3 years
4 1
Safe

5 years
2 2
Risky
Evaluating the split on Term

**Choice 2: Split on Term**

**Loan status:**
- Safe
- Risky

**Tree Classification error**
- (root) = 0.33
- Split on credit = 0.22
- Split on term = 0.33
Choice 1 vs Choice 2: Comparing split on credit vs term

Choice 1: Split on Credit

Loan status:
Safe Risky

Root
6 3

Credit?
excellent
2 0
poor
1 2

Winner

Choice 2: Split on Term

Loan status:
Safe Risky

Root
6 3

Term?
3 years
4 1
5 years
2 2

Tree

<table>
<thead>
<tr>
<th></th>
<th>Classification error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(root)</td>
<td>0.33</td>
</tr>
<tr>
<td>split on credit</td>
<td>0.22</td>
</tr>
<tr>
<td>split on loan term</td>
<td>0.33</td>
</tr>
</tbody>
</table>
• Given a subset of data M (a node in a tree)

• For each remaining 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
Greedy Algorithm

- If split is perfect (classification error = 0) or out of features:
  - Stop
- Else:
  - repeat split selection with next stump
Decision stump: 1 level

Loan status: Safe Risky

Root:
6 3

Split on Credit

Credit?

excellent
2 0
Subset of data with Credit = excellent

fair
3 1
Subset of data with Credit = fair

poor
1 2
Subset of data with Credit = poor
- Stop if all points are in one class

**Loan status:**
- Safe
- Risky

**Root**

<table>
<thead>
<tr>
<th>Credit?</th>
<th>Safe</th>
<th>Risky</th>
</tr>
</thead>
<tbody>
<tr>
<td>Excellent</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Fair</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>Poor</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

All data points are Safe nothing else to do with this subset of data
Tree learning = Recursive stump learning

Loan status: Safe Risky

Root
6 3

Credit?

excellent
2 0

Safe

fair
3 1

Build decision stump with subset of data where Credit = fair

poor
2 1

Build decision stump with subset of data where Credit = poor
Loan status:
Safe Risky

Credit?
excellent 2 0
fair 3 1
poor 1 2

Term?
3 years 0 1
5 years 1 1

Income?
high 0 2
Low 1 0

Build another stump these data points
### Real valued features

<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>
Loan status:
Safe
Risky

Split on Income

Income?

< $60K
8 13

>= $60K
14 5

Subset of data with Income >= $60K
Best threshold?

Infinite possible values of $t$

Income < $t^*$

Income = $t^*$

Income >= $t^*$

Income

~$10K$

~$120K$

Safe

Risky
Threshold between points

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

Income

$10K$

$120K$

Safe

Risky
Only need to consider midpoints

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
---|---
$\geq 38$ | Predict Risky
$< 38$ | Predict Safe

Income

Age

0 10 20 30 40 ...

$\ldots$

$0K$ $40K$ $80K$

...
Each split partitions the 2-D space.
Depth 1: Split on $x[1]$
Depth 2
Threshold split caveat

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

- Decision boundaries can be complex!

![Graph showing decision boundaries at different depths](image)

- Depth 1
- Depth 2
- Depth 10
Overfitting

- Deep decision trees are prone to overfitting
  - Decision boundaries are interpretable but not stable
  - Small change in the dataset leads to big difference in the outcome

- Overcoming Overfitting:
  - Early stopping
    - Fixed length depth
    - Stop if error does not considerably decrease
  - Pruning
    - Grow full length trees
    - Prune nodes to balance a complexity penalty
Predicting probabilities

Loan status: Safe Risky

Credit?

excellent
9 2
Safe

fair
6 9
Risky

poor
3 1
Safe

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