CSE/STAT 416
Naïve Bayes and Decision Trees
Pre-Class Videos

Hunter Schafer
Paul G. Allen School of Computer Science & Engineering
University of Washington

April 26, 2021
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

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

\[ \hat{y}_i = -1 \quad \text{Very sure} \]
\[ \hat{y}_i = +1 \quad \text{Very sure} \]
\[ \hat{y}_i = -1 \quad \text{Not sure if} \quad \hat{y}_i = -1 \quad \text{or} \quad \hat{y}_i = +1 \]
\[ \hat{y}_i = +1 \quad \text{Very sure} \]

\[ P(y_i = +1 | x_i) = 0 \]
\[ P(y_i = +1 | x_i) = 0.5 \]
\[ P(y_i = +1 | x_i) = 1 \]

\[ \hat{P}(y = +1 | x) \]
\[ \hat{P}(y = +1|x, \hat{w}) = \text{sigmoid} \left( \hat{w}^T h(x) \right) = \frac{1}{1 + e^{-\hat{w}^T h(x)}} \]
Naïve Bayes
Idea:
Naïve Bayes

$$x = \text{"The sushi & everything else was awesome!"}$$

$$P\left(y = +1 \mid x = \text{"The sushi & everything else was awesome!"}\right)$$?

$$P\left(y = -1 \mid x = \text{"The sushi & everything else was awesome!"}\right)?$$

Idea: Select the class that is the most likely!

Bayes Rule:

$$P(y = +1 \mid x) = \frac{P(x \mid y = +1) P(y = +1)}{P(x)}$$

Example

$$P\left(\text{"The sushi & everything else was awesome!"} \mid y = +1\right) P(y = +1)$$

$$P(\text{"The sushi & everything else was awesome!"})$$

Since we’re just trying to find out which class has the greater probability, we can discard the divisor.
Naïve
Assumption

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

**Problem:** We have not seen the sentence before.

**Assumption:** Words are independent from each other.

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

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

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

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

\[
\quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \text{\&} \quad P(\text{everything} | y = +1) \cdot P(\text{else} | y = +1) \cdot P(\text{was} | y = +1) \]

\[
\quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \text{\&} \quad P(\text{awesome} | y = +1)
\]
How do we compute something like

$$P(y = +1)$$?

How do we compute something like

$$P(\text{“awesome”} \mid y = +1)$$?
If a feature is missing in a class everything becomes zero.

\[
P(\text{"The sushi & everything else was awesome!" } | y = +1) \\
= P(\text{The } | y = +1) \times P(\text{sushi } | y = +1) \times P(\& | y = +1) \\
\times P(\text{everything } | y = +1) \times P(\text{else } | y = +1) \times P(\text{was } | y = +1) \\
\times P(\text{awesome } | y = +1)
\]

Solutions?

- Take the log (product becomes a sum).
  - Generally define \(\log(0) = 0\) in these contexts
- Laplacian Smoothing (adding a constant to avoid multiplying by zero)
Logistic Regression:

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

Naïve Bayes:

\[
P(y|x_1, x_2, ..., x_d) = \prod_{j=1}^{d} P(x_j|y) \ P(y)
\]
Compare Models

**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)
CSE/STAT 416
Naïve Bayes and Decision Trees

Hunter Schafer
Paul G. Allen School of Computer Science & Engineering
University of Washington

April 26, 2021

❓ Questions? Raise hand or sli.do #cs416
💬 Before Class: Pro-rain or anti-rain person?
🎵 Listening to: Alvvays
### Compare Models

**Logistic Regression:**

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

**Naïve Bayes:**

\[
P(y|x_1, x_2, \ldots, x_d) = \prod_{j=1}^{d} P(x_j|y) \ P(y)
\]

- Based on counts of words/classes
  - Laplace Smoothing
Compare Models

**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)
Recap: What is the predicted class for this sentence assuming we have the following training set (no Laplace Smoothing).

“he is not cool”

\[
P(y=+1) = \frac{2}{3}
\]

\[
P(\text{“he is not cool”} | y=+1)
= P(\text{“he”} | y=+1) P(\text{“is”} | y=+1) P(\text{“not”} | y=+1) P(\text{“cool”} | y=+1)
= \frac{2}{11} \cdot \frac{3}{11} \cdot \frac{1}{11} \cdot \frac{1}{11} = \frac{6}{11^4}
\]

\[
P(y=+1 | x) \propto \frac{6}{11^4} \cdot \frac{2}{3} = ... > 0
\]

<table>
<thead>
<tr>
<th>Sentence</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>this dog is cute</td>
<td>Positive</td>
</tr>
<tr>
<td>he does not like dogs</td>
<td>Negative</td>
</tr>
<tr>
<td>he is not bad he is cool</td>
<td>Positive</td>
</tr>
</tbody>
</table>

\[
P(y=-1 | x) \propto P(x | y=-1) P(y=-1)
= 0 \cdot \frac{1}{3}
\]

\[
P(\text{“cool”} | y=-1) = 0
\]
Humans often make decisions based on Flow Charts or Decision Trees.
Parametric vs. Non-Parametric Methods

Parametric Methods: make assumptions about the data distribution

- Linear Regression ⇒ assume the data is linear
- Logistic Regression ⇒ assume probability has the shape of a logistic curve and linear decision boundary
- Those assumptions result in a *parameterized* function family. Our learning task is to learn the parameters.

Non-Parametric Methods: (mostly) don’t make assumptions about the data distribution

- Decision Trees, k-NN (soon)
- We’re still learning something, but not the parameters to a function family that we’re assuming describes the data.
- Useful when you don’t want to (or can’t) make assumptions about the data distribution.
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
Did I pay previous loans on time?

Example: excellent, good, or fair
What’s my income?

Example:
$80K per year
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
Intelligent application

Loan Applications

Intelligent loan application review system

Safe ✓

Risky ×

Risky ×
Classifier review

- **Loan Application**
  - Input: \( x_i \)

- **Classifier MODEL**
  - Output: \( \hat{y}_i \)
    - \( \hat{y}_i = +1 \) Safe
    - \( \hat{y}_i = -1 \) Risky
Setup

Data (N observations, 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>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>

Evaluation: classification error

Many possible decisions: number of trees grows exponentially!
With our discussion of bias and fairness from last week, discuss the potential biases and fairness concerns that might be present in our dataset about loan safety.

**Some concerns**

- Predictions affecting economy (2008 financial crisis)
- Biases in training data → biased outcomes
  - Redlining, access to high paying jobs, etc.
- Legal constraints on which features to use + constraints on outputs (e.g., non-discrimination against race)
Decision Trees

- **Branch/Internal node**: splits into possible values of a feature
- **Leaf node**: final decision (the class value)
Brain Break
Growing Trees
Loan status: Safe  Risky

Root
6  3

# 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 leaf node, set $\hat{y} =$ majority value

Loan status: Safe Risky

credit?

excellent 2 0
Safe

fair 3 1
Safe

poor 1 2
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 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 1
```
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
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?

Error = \frac{\text{# mistakes}}{\text{# data points}}

Idea: Calculate classification error of this decision stump

Loan status: Safe Risky

Credit?

excellent
2 0

fair
3 1

poor
1 2
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

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

- **Root:**
  - 6 correct
  - 3 mistakes

\( \hat{y} = \text{majority class} \)

- **Error**

\[ \text{Error} = \frac{3}{9} = 0.33 \]

<table>
<thead>
<tr>
<th>Tree</th>
<th>Classification error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(root)</td>
<td>0.33</td>
</tr>
</tbody>
</table>
Does a split on Credit reduce classification error below 0.33?

Choice 1: Split on Credit

Loan status: Safe Risky

Credit?

excellent
2 0

fair
3 1

poor
1 2

Root
6 3
Choice 1: Split on Credit

Loan status:
Safe Risky

Root
6 3

Credit?

excellent
2 0

fair
3 1

poor
1 2

0 mistakes
1 mistake
1 mistake

Error = \frac{0 + 1 + 1}{9} = \frac{2}{9}
= 0.22

Tree | Classification error
---|---
(root) | 0.33
Split on credit | 0.22
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

Root
6 3

Term?

3 years
4 1
Safe
1 mistake

5 years
2 2
Risky
2 mistakes

Error = \frac{1 + 2}{9} = \frac{3}{9} = 0.33

<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>
<tr>
<td>Split on term</td>
<td>0.33</td>
</tr>
</tbody>
</table>
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

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>Split Conditions</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>
Split Selection

Split(node)

- Given $M$, the subset of training data at a node
- For each (remaining) feature $h_j(x)$:
  - Split data $M$ on feature $h_j(x)$
  - Compute the classification error for the split
- Chose feature $h^*_j(x)$ with the lowest classification error

In practice, allow multiple splits per feature.
Greedy & Recursive Algorithm

BuildTree(node)
  o If termination criterion is met:
    o Stop
  o Else:
    o Split(node)
    o For child in node:
      o BuildTree(child)
Loan status: Safe Risky

Split on Credit

Credit?

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

Subset of data with Credit = excellent:
excellent
excellent

Subset of data with Credit = fair:
fair
fair

Subset of data with Credit = poor:
poor
poor
For now: Stop when all points are in one class

Loan status: Safe Risky

Root

Credit?

excellent
good

fair

poor

Safe

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

Leaf node
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 → Safe
fair: 3 1
poor: 1 2

Term?
3 years: 2 0 → Safe
5 years: 1 1

Income?
high: 0 2 → Risky
Low: 1 0 → Safe

Build another stump these data points
Think

1 min

What predictions should the below decision tree output for the following datapoints?

<table>
<thead>
<tr>
<th>Credit</th>
<th>Term</th>
<th>Income</th>
</tr>
</thead>
<tbody>
<tr>
<td>excellent</td>
<td>5 yrs</td>
<td>high</td>
</tr>
<tr>
<td>fair</td>
<td>3 yrs</td>
<td>low</td>
</tr>
<tr>
<td>poor</td>
<td>5 yrs</td>
<td>(missing)</td>
</tr>
</tbody>
</table>

Loan status: Safe Risky

Root

Credit?

excellent

fair

poor

Term?

Income?

3 years

5 years

high

low

Safe

Risky

Safe

Safe

???
What predictions **should** the below decision tree output for the following datapoints?

### Loan status:
- Safe
- Risky

<table>
<thead>
<tr>
<th>Credit</th>
<th>Term</th>
<th>Income</th>
</tr>
</thead>
<tbody>
<tr>
<td>excellent</td>
<td>5 yrs</td>
<td>high</td>
</tr>
<tr>
<td>fair</td>
<td>3 yrs</td>
<td>low</td>
</tr>
<tr>
<td>poor</td>
<td>5 yrs</td>
<td>(missing)</td>
</tr>
</tbody>
</table>

- **Root**:
  - 6 3

- **Credit?**
  - excellent: 2 0
  - fair: 3 1
  - poor: 1 2

- **Term?**
  - 3 years 1 0
  - 5 years 2 1

- **Income?**
  - high 0 2
  - low 1 0
Brain Break
<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 Income

Income?

< $60K
8  13

>= $60K
14  5

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

Similar to our simple, threshold model when discussing Fairness!

Infinite possible values of $t$

Income $= t^*$

$\text{Income} < t^*$

$\text{Income} \geq t^*$

Income

$10K$

$120K$

Safe

Risky
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 $[v_1, v_2, ..., v_N]$ denote sorted values

- **Step 2:**
  - For $i = [1, ..., N - 1]$
    - Consider split $t_i = \frac{v_i + v_{i+1}}{2}$
    - Compute classification error for threshold split $h_j(x) \geq t_i$
    - Choose the $t^*$ with the lowest class. error
Visualizing the threshold split

Threshold split is the line $Age = 38$
Split on Age ≥ 38

Predict Safe

Predict Risky
Each split partitions the 2-D space

- Age $\geq$ 35
- Income $\geq$ $45K$
- Income $\geq$ $45K$
- Age $\geq$ 38
- Income $\geq$ 60K
- Age $< 38$
- Income $< 60K$
Depth 1: Split on $x[1]$
Depth 2

Splitting on same feature twice is allowed!
Threshold split caveat

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

- Decision boundaries can be complex!
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:
  - Stop when tree reaches certain height (e.g., 4 levels)
  - Stop when leaf has $\leq$ some num of points (e.g., 20 pts)
    - Will be the stopping condition for HW
  - Stop if split won’t significantly decrease error by more than some amount (e.g., 10%)

- Other methods include growing full tree and pruning back

- Fine-tune hyperparameters with validation set or CV
In Practice

- Trees can be used for classification or regression (CART)
  - Classification: Predict majority class for root node
  - Regression: Predict average label for root node

- In practice, we don’t minimize classification error but instead some more complex metric to measure quality of split such as Gini Impurity or Information Gain (not covered in 416)

- Can also be used to predict probabilities
Predicting probabilities

Loan status:
Safe Risky

Credit?

excellent
Safe

fair
Risky

poor
Safe

P(y = Safe | x)

\[
P(y = \text{Safe} \mid x) = \frac{3}{3 + 1} = 0.75
\]
What you can do now:

- Define the assumptions and modeling for Naïve Bayes
- 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