Machine Learning Algorithms

Learning Machine Learning

Nils Reiter

CRETA
CENTER FOR REFLECTED TEXT ANALYTICS

September 26-27, 2018

Overview

Decision Trees

Evaluation (again)

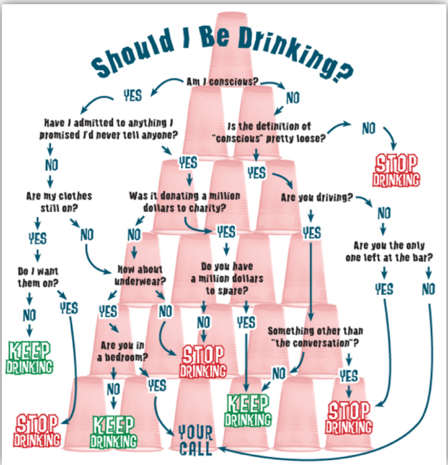
Naive Bayes

Section 1

Decision Trees

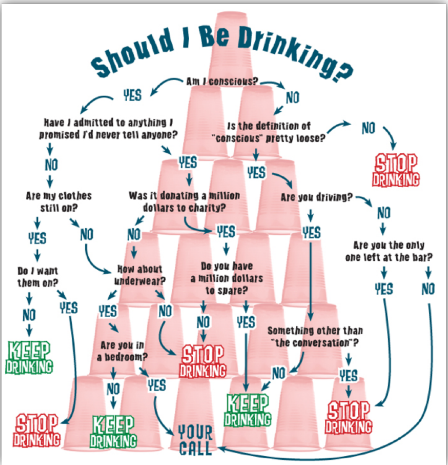
Decision Trees

Prediction Model – Toy Example



Decision Trees

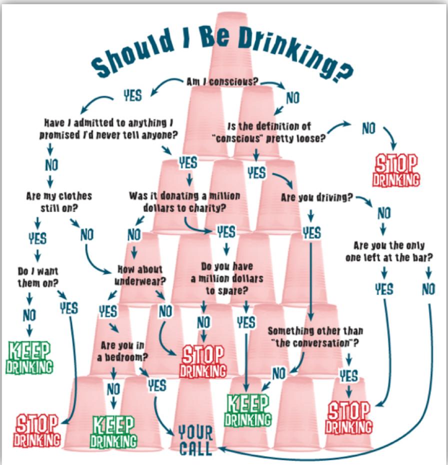
Prediction Model – Toy Example



- What are the instances?

Decision Trees

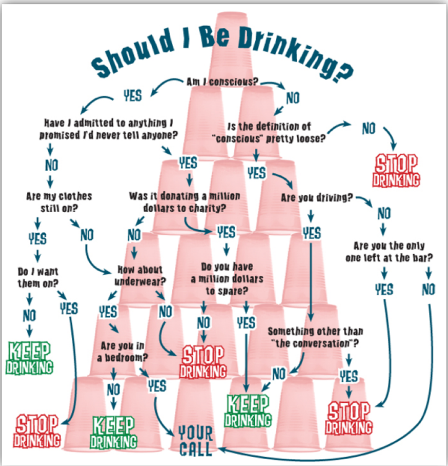
Prediction Model – Toy Example



- What are the instances?
  - Situations we are in (this is not really automatizable)

Decision Trees

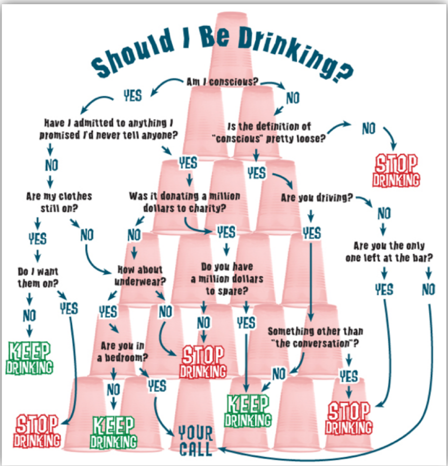
Prediction Model – Toy Example



- What are the instances?
  - Situations we are in (this is not really automatizable)
- What are the features?

Decision Trees

Prediction Model – Toy Example



- What are the instances?
  - Situations we are in (this is not really automatizable)
- What are the features?
  - Consciousness
  - Clothing situation
  - Promises made
  - Whether we are driving
  - …

Decision Trees

Trees

- Well-established data structure in CS

Decision Trees

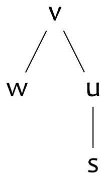
Trees

- Well-established data structure in CS
- A tree is a pair that contains
  - some value and
  - a (possibly empty) set of children
    - Children are also trees

Decision Trees

Trees

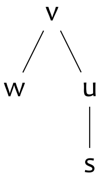
- Well-established data structure in CS
- A tree is a pair that contains
  - some value and
  - a (possibly empty) set of children
    - Children are also trees
- Formally: $\langle v, \{\langle w, \emptyset\rangle, \langle u, \{s, \emptyset\}\rangle\}\rangle$



Decision Trees

Trees

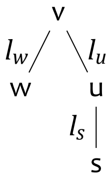
- Well-established data structure in CS
- A tree is a pair that contains
  - some value and
  - a (possibly empty) set of children
    - Children are also trees
- Formally: $\langle v, \{\langle w, \emptyset\rangle, \langle u, \{s, \emptyset\}\rangle\}\rangle$
- Recursive definition: "A tree is something and a tree"
  - Recursion is an important ingredient in many algorithms and data structures



Decision Trees

Trees

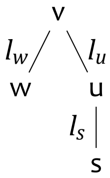
- Well-established data structure in CS
- A tree is a pair that contains
  - some value and
  - a (possibly empty) set of children
    - Children are also trees
- Formally: $\langle v, \{\langle w, \emptyset\rangle, \langle u, \{s, \emptyset\}\rangle\}\rangle$
- Recursive definition: "A tree is something and a tree"
  - Recursion is an important ingredient in many algorithms and data structures
- If the tree has labels on the edges, the pair becomes a triple
  - $\langle v, l_v, \{\langle w, l_w, \emptyset\rangle, \langle u, l_u\{s, \emptyset\}\rangle\}\rangle$



Decision Trees

Trees

- Well-established data structure in CS
- A tree is a pair that contains
  - some value and
  - a (possibly empty) set of children
    - Children are also trees
- Formally: $\langle v, \{\langle w, \emptyset\rangle, \langle u, \{s, \emptyset\}\rangle\}\rangle$
- Recursive definition: “A tree is something and a tree”
  - Recursion is an important ingredient in many algorithms and data structures
- If the tree has labels on the edges, the pair becomes a triple
  - $\langle v, l_v, \{\langle w, l_w, \emptyset\rangle, \langle u, l_u \{s, \emptyset\}\rangle\}\rangle$



Decision Trees

Prediction Model

- Each non-leaf node in the tree represents one feature
- Each leaf node represents a class label
- Each branch at this node represents one possible feature value
  - Number of branches = $|v(f_i)|$ (number of possible values)

Decision Trees

Prediction Model

- Each non-leaf node in the tree represents one feature
- Each leaf node represents a class label
- Each branch at this node represents one possible feature value
  - Number of branches = $|v(f_i)|$ (number of possible values)
- Make a prediction for x:
  1. Start at root node
  2. If it's a leaf node
     - assign the class label
  3. Else
     - Check node which feature is to be tested (f_i)
     - Extract $f_i(x)$
     - Follow corresponding branch
     - Go to 2

Decision Trees

Example Task

- D_{train}: A deck of 12 playing cards (selected out of 52)
- Target classes: Their symbols ♣♠♢♡
- Features
  - f_1: Does it show a number? $v(f_1) = \{0, 1\}$
  - f_2: Is it black or red? $v(f_2) = \{b, r\}$
  - f_3: Is it even, odd, or a face card? $v(f_3) = \{e, o, f\}$

Disclaimer: This task is artificial, because there is no connection of the features and the target classes in a full deck. It only serves to illustrate the algorithm.

Decision Trees

Example Task

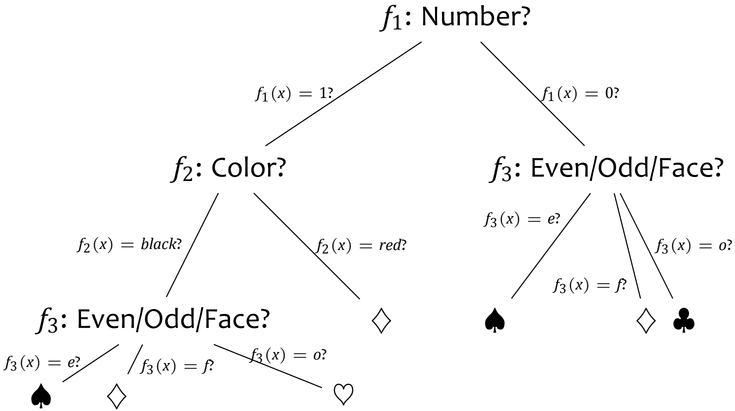


Figure: Example Prediction Model. The model is entirely made up and is not expected to perform well, but it can be used for classification right away.

Decision Trees

Learning Algorithm

- Core idea: The tree represents splits of the training data
  1. Start with the full data set D_{train} as D
  2. If D only contains members of a single class:
     - Done.
  3. Else:
     - Select a feature f_i
     - Extract feature values of all instances in D
     - Split the data set according to f_i: $D = D_v \cup D_w \cup D_u \ldots$
     - Go back to 2
- Remaining question: How to select features?

Decision Trees

Feature Selection

- What is a good feature?
  - One that maximizes homogeneity in the split data set

Decision Trees

Feature Selection

- What is a good feature?
  - One that maximizes homogeneity in the split data set
- "Homogeneity"
  - Increase
    $\{\spadesuit\spadesuit\spadesuit\heartsuit\} = \{\heartsuit\} \cup \{\spadesuit\spadesuit\spadesuit\}$
  - No increase
    $\{\spadesuit\spadesuit\spadesuit\heartsuit\} = \{\spadesuit\} \cup \{\spadesuit\spadesuit\heartsuit\}$

Decision Trees

Feature Selection

- What is a good feature?
  - One that maximizes homogeneity in the split data set
- "Homogeneity"
  - Increase
    $\{\spadesuit\spadesuit\spadesuit\heartsuit\} = \{\heartsuit\} \cup \{\spadesuit\spadesuit\spadesuit\} \leftarrow$ better split!
  - No increase
    $\{\spadesuit\spadesuit\spadesuit\heartsuit\} = \{\spadesuit\} \cup \{\spadesuit\spadesuit\heartsuit\}$
- Homogeneity: Entropy/information — Shannon (1948)

Decision Trees

Feature Selection

- What is a good feature?
  - One that maximizes homogeneity in the split data set
- “Homogeneity”
  - Increase
    $\{\spadesuit\spadesuit\spadesuit\heartsuit\} = \{\heartsuit\} \cup \{\spadesuit\spadesuit\spadesuit\} \leftarrow$ better split!
  - No increase
    $\{\spadesuit\spadesuit\spadesuit\heartsuit\} = \{\spadesuit\} \cup \{\spadesuit\spadesuit\heartsuit\}$
- Homogeneity: Entropy/information Shannon (1948)
- Rule: Always select the feature with the highest *information gain* (IG)

Decision Trees

Entropy (Shannon 1948)

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_b p(x_i)$$

Examples

- $H([4]) = -\frac{4}{4} \log_b \frac{4}{4} = 0$
- $H([3,1]) = -\frac{3}{4} \log_b \frac{3}{4} - \frac{1}{4} \log_b \frac{1}{4} = 0.562$
- $H([2,2]) = 0.693$

Decision Trees

Feature Selection (2)

$$\{\spadesuit\spadesuit\spadesuit\heartsuit\}$$

$$/ \quad \backslash$$

$$\{\heartsuit\}\{\spadesuit\spadesuit\spadesuit\}$$

$$\begin{aligned} H(\{\spadesuit\spadesuit\spadesuit\heartsuit\}) &= H([3,1]) \\ &= 0.562 \\ H(\{\heartsuit\}) &= H([1]) = 0 \\ H(\{\spadesuit\spadesuit\spadesuit\}) &= H([3]) \\ &= 0 \end{aligned}$$

$$\{\spadesuit\spadesuit\spadesuit\heartsuit\}$$

$$/ \quad \backslash$$

$$\{\spadesuit\}\{\spadesuit\spadesuit\heartsuit\}$$

$$\begin{aligned} H(\{\spadesuit\spadesuit\spadesuit\heartsuit\}) &= H([3,1]) \\ &= 0.562 \\ H(\{\spadesuit\}) &= H([1]) = 0 \\ H(\{\spadesuit\spadesuit\heartsuit\}) &= H([2,1]) \\ &= 0.637 \end{aligned}$$

Decision Trees

Feature Selection (3)

$\{\spadesuit\spadesuit\spadesuit\heartsuit\}$
/ \
$\{\heartsuit\}\{\spadesuit\spadesuit\spadesuit\}$

$\{\spadesuit\spadesuit\spadesuit\heartsuit\}$
/ \
$\{\spadesuit\}\{\spadesuit\spadesuit\heartsuit\}$

$$\begin{aligned} H(\{\spadesuit\spadesuit\spadesuit\heartsuit\}) &= 0.562 \\ H(\{\heartsuit\}) &= 0 \\ H(\{\spadesuit\spadesuit\spadesuit\}) &= 0 \end{aligned}$$

$$\begin{aligned} H(\{\spadesuit\spadesuit\spadesuit\heartsuit\}) &= 0.562 \\ H(\{\spadesuit\}) &= 0 \\ H(\{\spadesuit\spadesuit\heartsuit\}) &= 0.637 \end{aligned}$$

$$\begin{aligned} IG(f_1) &= H(\{\spadesuit\spadesuit\spadesuit\heartsuit\}) - \varnothing\left(H(\{\heartsuit\}), H(\{\spadesuit\spadesuit\spadesuit\})\right) \\ &= 0.562 - 0 = 0.562 \\ IG(f_2) &= H(\{\spadesuit\spadesuit\spadesuit\heartsuit\}) - \varnothing\left(H(\{\spadesuit\}), H(\{\spadesuit\spadesuit\heartsuit\})\right) \\ &= 0.562 - (\frac{3}{4}0.637 + \frac{1}{4}0) \\ &= 0.562 - 0.562 - 0.477 = 0.085 \end{aligned}$$

Let's Train a Decision Tree

Initial Situation

$$
\begin{aligned}
C &= \{\clubsuit\spadesuit\diamondsuit\heartsuit\} \\
D_{train} &= \{7\clubsuit, A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit, 5\diamondsuit, \\
&\quad 8\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}
\end{aligned}
$$

Let's Train a Decision Tree

Initial Situation

$$
\begin{aligned}
C &= \{\clubsuit\spadesuit\diamondsuit\heartsuit\} \\
D_{train} &= \{7\clubsuit, A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit, 5\diamondsuit, \\
&\quad 8\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}
\end{aligned}
$$

Class	Frequency	%
♠	4	33.3
♢	4	33.3
♡	3	25
♣	1	8.3

Let's Train a Decision Tree

Initial Situation

$$C = \{♣♠♢♡\}$$

$$D_{train} = \{7♣, A♠, Q♠, K♠, J♠, 5♢, 8♢, 3♢, 7♢, 3♡, 7♡, 5♡\}$$

Class	Frequency	%
♠	4	33.3
♢	4	33.3
♡	3	25
♣	1	8.3

$$H(♠♠♠♠♢♢♢♢♡♡♡♣) = H([4, 4, 3, 1]) = 1.286057$$

Let's Train a Decision Tree

f_1: Does it show a number?

- Splitting D according to f_1 yields
  - $\{7\clubsuit, 5\diamondsuit, 8\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - $\{A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- Intuitively: Is this good?

Let's Train a Decision Tree

f_1: Does it show a number?

- Splitting D according to f_1 yields
  - $\{7\clubsuit, 5\diamondsuit, 8\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - $\{A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- Intuitively: Is this good?
- Calculate entropies
  - $H([4, 3, 1]) = 0.9743148$
  - $H([4]) = 0$

Let's Train a Decision Tree

f_1: Does it show a number?

- Splitting D according to f_1 yields
  - $\{7\clubsuit, 5\diamondsuit, 8\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - $\{A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- Intuitively: Is this good?
- Calculate entropies
  - $H([4, 3, 1]) = 0.9743148$
  - $H([4]) = 0$
- Weighted average of entropy
  - $\frac{8}{12}H([4, 3, 1]) + \frac{4}{12}H([4]) = 0.6495432$

Let's Train a Decision Tree

f_1: Does it show a number?

- Splitting D according to f_1 yields
  - $\{7\clubsuit, 5\diamondsuit, 8\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - $\{A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- Intuitively: Is this good?
- Calculate entropies
  - $H([4,3,1]) = 0.9743148$
  - $H([4]) = 0$
- Weighted average of entropy
  - $\frac{8}{12}H([4,3,1]) + \frac{4}{12}H([4]) = 0.6495432$
- Calculate information gain for feature f_1
  - $IG(f_1) = H([4,4,3,1]) - 0.6495432 = 0.6365142$

Let's Train a Decision Tree

f_2: Is it black or red?

- Splitting D according to f_2 yields
  - $\{5\diamondsuit, 8\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - $\{7\clubsuit, A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- Intuitively: Is this good? Better than f_1?

Let's Train a Decision Tree

f_2: Is it black or red?

- Splitting D according to f_2 yields
  - $\{5\diamondsuit, 8\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - $\{7\clubsuit, A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- Intuitively: Is this good? Better than f_1?
- Calculate entropies
  - $H([4,3]) = 0.6829081$
  - $H([4,1]) = 0.5004024$

Let's Train a Decision Tree

f_2: Is it black or red?

- Splitting D according to f_2 yields
  - $\{5\diamondsuit, 8\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - $\{7\clubsuit, A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- Intuitively: Is this good? Better than f_1?
- Calculate entropies
  - $H([4,3]) = 0.6829081$
  - $H([4,1]) = 0.5004024$
- Weighted average of entropy
  - $\frac{7}{12}H([4,3]) + \frac{5}{12}H([4,1]) = 0.6068641$

Let's Train a Decision Tree

f_2: Is it black or red?

- Splitting D according to f_2 yields
  - $\{5\diamondsuit, 8\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - $\{7\clubsuit, A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- Intuitively: Is this good? Better than f_1?
- Calculate entropies
  - $H([4, 3]) = 0.6829081$
  - $H([4, 1]) = 0.5004024$
- Weighted average of entropy
  - $\frac{7}{12}H([4, 3]) + \frac{5}{12}H([4, 1]) = 0.6068641$
- Calculate information gain for feature f_2
  - $IG(f_2) = H([4, 4, 3, 1]) - 0.6068641 = 0.6791933$

Let's Train a Decision Tree

f_3: Is it even, odd, or a face?

- Splitting D according to f_3 yields
  - $\{8\diamondsuit\}$
  - $\{7\clubsuit, 5\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - $\{A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- Intuitively: Is this good? Better than f_1 or f_2?

Let's Train a Decision Tree

f_3: Is it even, odd, or a face?

- Splitting D according to f_3 yields
  - $\{8\diamondsuit\}$
  - $\{7\clubsuit, 5\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - $\{A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- Intuitively: Is this good? Better than f_1 or f_2?
- Calculate entropies
  - $H([1]) = 0$
  - $H([1, 3, 3]) = 1.004242$
  - $H([4]) = 0$

Let's Train a Decision Tree

f_3: Is it even, odd, or a face?

- Splitting D according to f_3 yields
  - $\{8\diamondsuit\}$
  - $\{7\clubsuit, 5\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - $\{A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- Intuitively: Is this good? Better than f_1 or f_2?
- Calculate entropies
  - $H([1]) = 0$
  - $H([1, 3, 3]) = 1.004242$
  - $H([4]) = 0$
- Weighted average of entropies
  - $\frac{1}{12}H([1]) + \frac{7}{12}H([1, 3, 3]) + \frac{4}{12}H([0]) = 0.5858081$

Let's Train a Decision Tree

f_3: Is it even, odd, or a face?

- Splitting D according to f_3 yields
  - $\{8\diamondsuit\}$
  - $\{7\clubsuit, 5\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - $\{A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- Intuitively: Is this good? Better than f_1 or f_2?
- Calculate entropies
  - $H([1]) = 0$
  - $H([1, 3, 3]) = 1.004242$
  - $H([4]) = 0$
- Weighted average of entropies
  - $\frac{1}{12}H([1]) + \frac{7}{12}H([1, 3, 3]) + \frac{4}{12}H([0]) = 0.5858081$
- Calculate information gain for feature f_3
  - $IG(f_3) = H([4, 4, 3, 1]) - 0.5858081 = 0.7002492$

Let's Train a Decision Tree

First Feature

Feature	Information gain
f_1	0.637
f_2	0.679
f_3	0.7

Let's Train a Decision Tree

First Feature

Feature	Information gain
f_1	0.637
f_2	0.679
f_3	0.7

- The algorithm selects f_3 as the first feature!

Let's Train a Decision Tree

First Feature

Feature	Information gain
f_1	0.637
f_2	0.679
f_3	0.7

- The algorithm selects f_3 as the first feature!
- Next, we continue *recursively* with each sub set
  - $\{8\diamondsuit\}$
  - $\{7\clubsuit, 5\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - $\{A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$

Let's Train a Decision Tree

First Feature

Feature	Information gain
f_1	0.637
f_2	0.679
f_3	0.7

- The algorithm selects f_3 as the first feature!
- Next, we continue *recursively* with each sub set
  - $\{8\diamondsuit\}$
    - ✓ No further action needed!
  - $\{7\clubsuit, 5\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - $\{A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
    - ✓ No further action needed!

Let's Train a Decision Tree

Final Tree

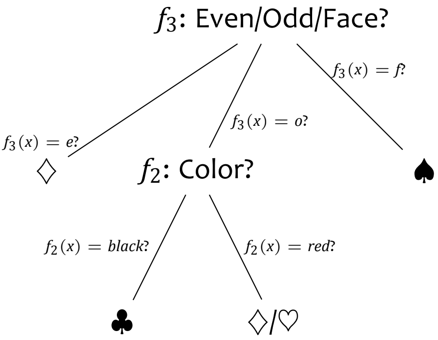


Figure: Final prediction model according to the training we did in class

Decision Trees

Summary

- Classification algorithm
- Built around trees, recursive learning and prediction
- Pros
  - Highly transparent
  - Reasonably fast
  - Dependencies between features can be incorporated into the model
- Cons
  - Often not very good
  - No pairwise dependencies
  - May lead to overfitting
  - Only nominal features
- Variants exist

Section 2

Evaluation (again)

Evaluation (again)

Precision and Recall

- Accuracy is a single number for the entire classification
- Do some of the classes fare better than others?
- There are two metrics for this: Precision and Recall
  - Both are calculated *per class* (and can be averaged again)

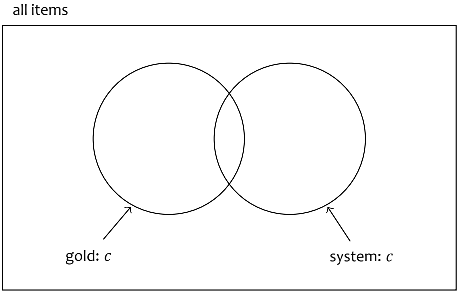


Figure: Identifying true/false positives/negatives

Evaluation (again)

Precision and Recall

- Accuracy is a single number for the entire classification
- Do some of the classes fare better than others?
- There are two metrics for this: Precision and Recall
    - Both are calculated *per class* (and can be averaged again)

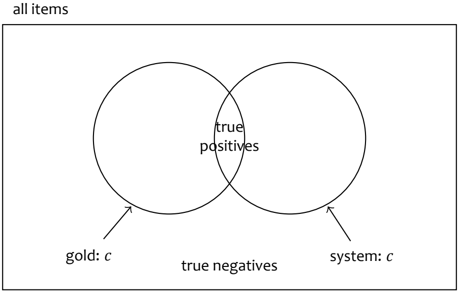


Figure: Identifying true/false positives/negatives

Evaluation (again)

Precision and Recall

- Accuracy is a single number for the entire classification
- Do some of the classes fare better than others?
- There are two metrics for this: Precision and Recall
  - Both are calculated *per class* (and can be averaged again)

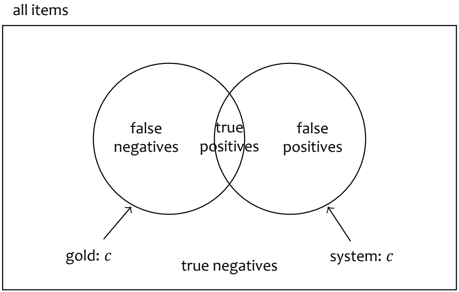
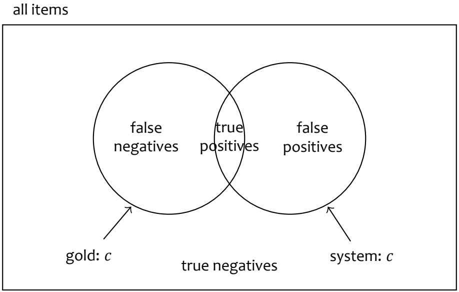


Figure: Identifying true/false positives/negatives

Evaluation

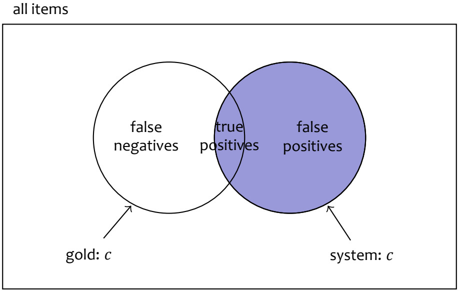
Precision and Recall



- **true positives** Correctly identified items of class c
- **true negatives** Correctly identified items of other classes
- **false positives** System predicts c, but it's another class
- **false negatives** System predicts something else, but it's c

Evaluation

Precision and Recall

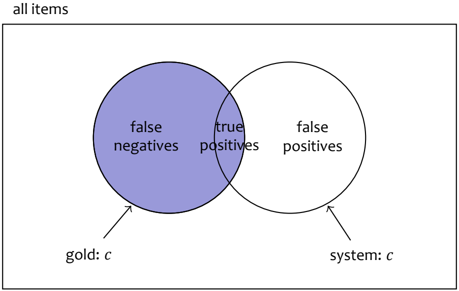


precision How many of the items predicted as c are actually correct?

$P = \frac{tp}{tp+fp}$

Evaluation

Precision and Recall



precision How many of the items predicted as c are actually correct?

$P = \frac{tp}{tp+fp}$

recall How many of the items that are c are actually identified?

$R = \frac{tp}{tp+fn}$

Evaluation

Precision and Recall

precision How many of the items *predicted as c* are actually correct?

recall How many of the items that *are in class c* are actually found by the system?

- Precision and recall measure different kinds of errors the systems make
  - Precision errors are often easier to spot for humans
  - Recall errors are hurtful, if only instances of one class are looked at or analyzed – missing instances will never be found
- Average P/R values over all classes are often given
- Sometimes combined into an f_1-score
  - $f_1 = 2\frac{precision*recall}{precision+recall}$
  - 'harmonic mean' between the two

Section 3

Naive Bayes

Naive Bayes

Prediction Model

- Probabilistic model
  (i.e., takes probabilities into account)
- Probabilities are estimated on training data (relative frequencies)

Naive Bayes

Prediction Model

$$prediction(x) = \underset{c \in C}{\operatorname{argmax}}\, p(c|f_1(x), f_2(x), \ldots, f_n(x))$$

(i.e., we calculate the probability for each possible class c, given the feature values of the item x, and we assign most probably class)
In our case:

$$prediction(x) = \underset{c \in \{\clubsuit\spadesuit\heartsuit\diamondsuit\}}{\operatorname{argmax}}\, p(c|f_1(x), f_2(x), \ldots, f_n(x))$$

- argmax: Select the argument that maximizes the expression
- How exactly do we calculate $p(c|f_1(x), f_2(x), \ldots, f_n(x))$?

Naive Bayes

Prediction Model

$$p(c|f_1, \ldots, f_n) \quad =$$

Naive Bayes

Prediction Model

$$p(c|f_1, \ldots, f_n) = \frac{p(c, f_1, f_2, \ldots, f_n)}{p(f_1, f_2, \ldots, f_n)}$$

Naive Bayes

Prediction Model

$$p(c|f_1, \ldots, f_n) = \frac{p(c, f_1, f_2, \ldots, f_n)}{p(f_1, f_2, \ldots, f_n)} = \frac{p(f_1, f_2, \ldots, f_n, c)}{p(f_1, f_2, \ldots, f_n)}$$

Naive Bayes

Prediction Model

$$p(c|f_1,\ldots,f_n) = \frac{p(c,f_1,f_2,\ldots,f_n)}{p(f_1,f_2,\ldots,f_n)} = \frac{p(f_1,f_2,\ldots,f_n,c)}{p(f_1,f_2,\ldots,f_n)}$$

denominator is constant, so we skip it

$$\propto p(f_1|f_2,\ldots,f_n,c)p(f_2|f_3,\ldots,f_n,c)\ldots p(c)$$

Naive Bayes

Prediction Model

$$p(c|f_1,\ldots,f_n) = \frac{p(c,f_1,f_2,\ldots,f_n)}{p(f_1,f_2,\ldots,f_n)} = \frac{p(f_1,f_2,\ldots,f_n,c)}{p(f_1,f_2,\ldots,f_n)}$$

denominator is constant, so we skip it

$$\propto p(f_1|f_2,\ldots,f_n,c)p(f_2|f_3,\ldots,f_n,c)\ldots p(c)$$

Now we assume feature independence

$$= p(f_1|c)p(f_2|t)\ldots p(c)$$

Naive Bayes

Prediction Model

$$
\begin{aligned}
p(c|f_1,\ldots,f_n) &= \frac{p(c,f_1,f_2,\ldots,f_n)}{p(f_1,f_2,\ldots,f_n)} = \frac{p(f_1,f_2,\ldots,f_n,c)}{p(f_1,f_2,\ldots,f_n)} \\
& \quad \text{denominator is constant, so we skip it} \\
&\propto p(f_1|f_2,\ldots,f_n,c)p(f_2|f_3,\ldots,f_n,c)\ldots p(c) \\
& \quad \text{Now we assume feature independence} \\
&= p(f_1|c)p(f_2|t)\ldots p(c) \\
prediction(x) &= \operatorname*{argmax}_{c\in C} p(f_1(x)|c)p(f_2(x)|c)\ldots p(c)
\end{aligned}
$$

How do we get $p(f_i(x)|c)$? This is what the model has stored!

Naive Bayes

Learning Algorithm

- Very simple
  1. For each feature $f_i \in F$
     - Count frequency tables from the training set:

		C (classes)			
		c_1	c_2	...	c_m
$v(f_i)$	a	3	2	...	
	b	5	7	...	
	c	0	1	...	
	$\sum$	8	10		

  2. Calculate conditional probabilities
     - Divide each number by the sum of the entire column
     - E.g., $p(a|c_1) = \frac{3}{3+5+0}$ $\quad p(b|c_2) = \frac{7}{2+7+1}$

Naive Bayes – Example Task

Feature f_1: Number?

		C (classes)			
		♣	♠	♡	♢
$v(f_1)$	y	1	0	3	4
	n	0	4	0	0
	$\sum$	1	4	3	4

$$p(f_1 = y|\spadesuit) = 0 \qquad p(f_1 = n|\spadesuit) = 1$$

$$p(f_1 = y|\diamondsuit) = 1 \qquad p(f_1 = n|\diamondsuit) = 0$$

Naive Bayes – Example Task

Feature f_2: Color?

C (classes)

$v(f_2)$		♣	♠	♡	♢
	b	0	0	3	4
	r	1	4	0	0
	$\sum$	1	4	3	4

$$p(f_2 = r|\spadesuit) = 0 \qquad p(f_2 = b|\spadesuit) = 1$$
$$p(f_2 = r|\diamondsuit) = 1 \qquad p(f_2 = b|\diamondsuit) = 0$$

Naive Bayes – Example Task

Feature f_3: Odd/Even/Face?

		C (classes)			
		$\clubsuit$	$\spadesuit$	$\heartsuit$	$\diamondsuit$
$v(f_3)$	o	1	0	3	3
	e	0	0	0	1
	f	0	4	0	0
	$\sum$	1	4	3	4

$$p(f_3 = o|\spadesuit) = 0 \quad p(f_3 = e|\spadesuit) = 0 \quad p(f_3 = f|\spadesuit) = 1$$

$$p(f_3 = o|\diamondsuit) = \frac{3}{4} \quad p(f_3 = e|\diamondsuit) = \frac{1}{4} \quad p(f_3 = f|\diamondsuit) = 0$$

Naive Bayes – Example Task

Prediction

$$
\begin{aligned}
prediction(K\spadesuit) &= \operatorname*{argmax}_{c \in \{\spadesuit\clubsuit\heartsuit\diamondsuit\}} p(c|n,b,f) \\
p(\clubsuit|n,b,f) &= p(f_1 = n|\clubsuit) * p(f_2 = b|\clubsuit) * p(f_3 = f|\clubsuit) \\
&= 0 \\
p(\heartsuit|n,b,f) &= p(f_1 = n|\heartsuit) * p(f_2 = b|\heartsuit) * p(f_3 = f|\heartsuit) \\
&= 0 \\
p(\spadesuit|n,b,f) &= p(f_1 = n|\spadesuit) * p(f_2 = b|\spadesuit) * p(f_3 = f|\spadesuit) \\
&= 1 * 1 * 1 = 1
\end{aligned}
$$

We predict ♠

Naive Bayes – Example Task

Prediction

$$
\begin{aligned}
prediction(6\diamondsuit) &= \operatorname*{argmax}_{c \in \{\spadesuit\clubsuit\heartsuit\diamondsuit\}} p(c|y, r, e) \\
p(\clubsuit|y, r, e) &= p(f_1 = y|\clubsuit) * p(f_2 = r|\clubsuit) * p(f_3 = e|\clubsuit) \\
&= 0 \\
p(\heartsuit|y, r, e) &= p(f_1 = y|\heartsuit) * p(f_2 = r|\heartsuit) * p(f_3 = e|\heartsuit) \\
&= 1 * 1 * 0 = 0 \\
p(\diamondsuit|y, r, e) &= p(f_1 = y|\diamondsuit) * p(f_2 = r|\diamondsuit) * p(f_3 = e|\diamondsuit) \\
&= 1 * 1 * \frac{1}{4} = \frac{1}{4}
\end{aligned}
$$

We predict $\diamondsuit$

Naive Bayes – Example Task

Prediction

$$
\begin{aligned}
prediction(K\diamondsuit) &= \operatorname*{argmax}_{c \in \{\spadesuit\clubsuit\heartsuit\diamondsuit\}} p(c|n,r,f) \\
p(\clubsuit|n,r,f) &= p(f_1 = y|\clubsuit) * p(f_2 = r|\clubsuit) * p(f_3 = e|\clubsuit) \\
&= 0 \\
p(\heartsuit|n,r,f) &= p(f_1 = y|\heartsuit) * p(f_2 = r|\heartsuit) * p(f_3 = e|\heartsuit) \\
&= 0 \\
p(\diamondsuit|n,r,f) &= p(f_1 = y|\diamondsuit) * p(f_2 = r|\diamondsuit) * p(f_3 = e|\diamondsuit) \\
&= 0
\end{aligned}
$$

Oops, all probabilities are zero

Naive Bayes

Smoothing

- Whenever multiplication is involved, zeros are dangerous
- Smoothing is used to avoid zeros
- Different possibilities
- Simple: Add something to the probabilities
  - $\frac{x_i+a}{N+ad}$
  - E.g., $p(f_3 = e|\spadesuit) = \frac{0+1}{4+1*4}$

Naive Bayes

- ‘Naive’: Assuming feature independence is usually wrong
  - Even in our toy example, f_1 and f_3 are highly dependent
- Pros
  - Easy to implement, fast
  - Small models
- Cons
  - Naive: Feature dependence not modeled
  - Fragile for unseen data (without smoothing)

References I

Shannon, Claude E. “A mathematical theory of communication”. In: *The Bell System Technical Journal* 27.3 (July 1948), pp. 379–423.