Reflected Text Analysis beyond Linguistics

DGfS-CL fall school

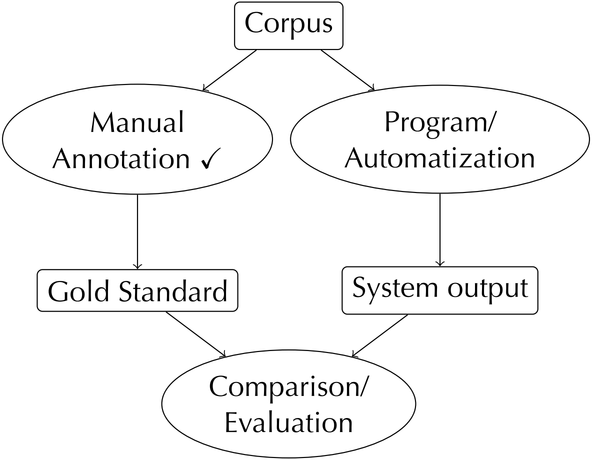
Nils Reiter,
nils.reiter@ims.uni-stuttgart.de

Sept. 9-13, 2019

Part III

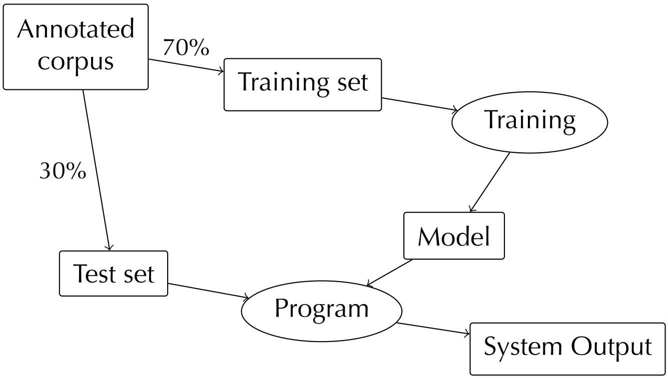
Automatisation and Machine Learning

Experiments



Flash forward: Evaluation

- Goal: Predict the quality on new data
- The program cannot have seen the data, so that it's a realistic test



Introduction

- What is machine learning?
    - Method to find patterns, hidden structures and undetected relations in data

Introduction

- What is machine learning?
  - Method to find patterns, hidden structures and undetected relations in data
- It's all around us
  - Stock market transactions
  - Search engines
  - Surveillance
  - Data-driven research & science
  - …

Introduction

- What is machine learning?
  - Method to find patterns, hidden structures and undetected relations in data
- It's all around us
  - Stock market transactions
  - Search engines
  - Surveillance
  - Data-driven research & science
  - ...
- Why is it interesting for text analysis?
  - Big data analyses
    - Automatic prediction of phenomena
    - Canonisation, Euro-centrism
    - Statements about 1000 texts more convincing than abt 10
  - Insights into data
    - By inspecting features and making error analysis

Two Parts

Prediction Model

How do we make predictions on data instances?
(e.g., how do we assign a part of speech tag for a word?)

Learning Algorithm

How do we create a prediction model, given annotated data?
(e.g. how do we create rules for assigning a part of speech tag for a word?)

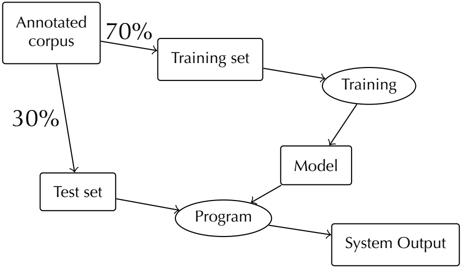
Two Parts

Prediction Model

How do we make predictions on data instances?
(e.g., how do we assign a part of speech tag for a word?)

Learning Algorithm

How do we create a prediction model, given annotated data?
(e.g. how do we create rules for assigning a part of speech tag for a word?)



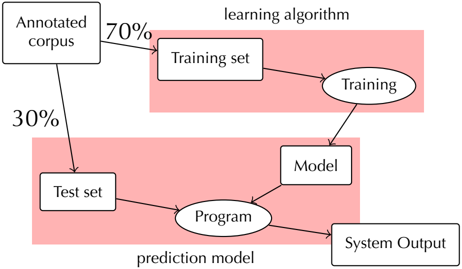
Two Parts

Prediction Model

How do we make predictions on data instances?
(e.g., how do we assign a part of speech tag for a word?)

Learning Algorithm

How do we create a prediction model, given annotated data?
(e.g. how do we create rules for assigning a part of speech tag for a word?)



Machine Learning

Classification

- Assigning *classes* to *objects/instances/items*

Machine Learning

Classification

- Assigning *classes* to *objects/instances/items*
  - Words $\rightarrow$ parts of speech

Machine Learning
Classification

- Assigning *classes* to *objects/instances/items*
  - Words $\rightarrow$ parts of speech
  - Photo portraits $\rightarrow$ gender of the depicted person

Machine Learning

Classification

- Assigning *classes* to *objects/instances/items*
  - Words $\rightarrow$ parts of speech
  - Photo portraits $\rightarrow$ gender of the depicted person
  - Photo portraits $\rightarrow$ name of depicted person

Machine Learning

Classification

- Assigning *classes* to *objects/instances/items*
  - Words → parts of speech
  - Photo portraits → gender of the depicted person
  - ~~Photo portraits → name of depicted person~~

Machine Learning
Classification

- Assigning *classes* to *objects/instances/items*
  - Words → parts of speech
  - Photo portraits → gender of the depicted person
  - ~~Photo portraits → name of depicted person~~
  - Texts → genres

Machine Learning
Classification

- Assigning *classes* to *objects/instances/items*
  - Words → parts of speech
  - Photo portraits → gender of the depicted person
  - ~~Photo portraits → name of depicted person~~
  - Texts → genres
- Prediction model: Responsible for the classification

Machine Learning
Classification

- Assigning *classes* to *objects/instances/items*
  - Words → parts of speech
  - Photo portraits → gender of the depicted person
  - ~~Photo portraits → name of depicted person~~
  - Texts → genres
- Prediction model: Responsible for the classification
- Many different models/algorithms available:
  - Decision trees
  - Support vector machines
  - Naïve bayes
  - Neural networks
  - Bayesian networks
  - ...

Machine Learning

Features

- Decision is based on features (= properties)
- The prediction model only sees feature values!
    - What's not encoded in a feature doesn't play a role
    - It's our job to provide useful features

Evaluation

- We *always* want to know how well machine learning works
- Straightforward evaluation: Comparison with a gold standard

Evaluation

- We *always* want to know how well machine learning works
- Straightforward evaluation: Comparison with a gold standard
- Most simple metric: Accuracy
  - Percentage of correctly classified instances (the higher the better)
  - Inverse: Error rate (percentage of incorrectly classified instances)

Evaluation

- We *always* want to know how well machine learning works
- Straightforward evaluation: Comparison with a gold standard
- Most simple metric: Accuracy
  - Percentage of correctly classified instances (the higher the better)
  - Inverse: Error rate (percentage of incorrectly classified instances)
- Accuracy is nice, but not enough
  - When improving systems, we want to *compare* our accuracy with the previous accuracy
  - When developing new systems, we want to know how difficult the task is
    - E.g., 60% accuracy when distinguishing 35 parts of speech is better than 60% accuracy when distinguishing nouns and all the rest

Evaluation

Baseline

Baseline

The baseline performance is the performance of a simple system, rule or thought experiment

Evaluation

Baseline

Baseline

The baseline performance is the performance of a simple system, rule or thought experiment

- Example 1: Gender of students in Stuttgart and Cologne
  - Task: Classify students according to their gender
  - Data
    - Stuttgart: 8585 of 25 705 students are female
    - Cologne: 29 793 of 48 841 students are female
  - Majority baseline: Everyone is female (Cologne) or male (Stuttgart)
  - Classification accuracies: 61% / 66.6%

Evaluation

Baseline

Baseline

The baseline performance is the performance of a simple system, rule or thought experiment

- Example 1: Gender of students in Stuttgart and Cologne
- Example 2: Gender of arbitrary Germans
  - Task: Classify a random German according to their gender
  - male: 40.7m vs. female: 41.8m
  - Random baseline: Toss a coin
  - Classification accuracy: about 50%

Evaluation

Baseline

Baseline

The baseline performance is the performance of a simple system, rule or thought experiment

- Example 1: Gender of students in Stuttgart and Cologne
- Example 2: Gender of arbitrary Germans
- Example 3: Detecting nouns
  - Task: Classify words into noun and non-noun
  - Most words are not nouns
  - Majority baseline: Every word is a non-noun
  - Accuracy (in a German text): 81.8%

Formalities and Notation

Why formal language?

Formal language is concise, exact and unambiguous. Slides will contain both.

Formalities and Notation

Why formal language?

Formal language is concise, exact and unambiguous. Slides will contain both.

- Data set D, split into D_{train} and D_{test}
  $D_{train} \cup D_{test} = D$

Formalities and Notation

Why formal language?

Formal language is concise, exact and unambiguous. Slides will contain both.

- Data set D, split into D_{train} and D_{test}
  $D_{train} \cup D_{test} = D$
- Data objects/instances/items: $x \in D$. x_{class} represents the class label (i.e., the target category)

Formalities and Notation

Why formal language?

Formal language is concise, exact and unambiguous. Slides will contain both.

- Data set D, split into D_{train} and D_{test}
  $D_{train} \cup D_{test} = D$
- Data objects/instances/items: $x \in D$. x_{class} represents the class label (i.e., the target category)
- Feature set $F = \{f_1, f_2, \ldots, f_n\}$
  - $v(f_i)$ is a set that contains all possible values of a feature
  - I.e., we know in advance which values a feature can take!

Formalities and Notation

Why formal language?

Formal language is concise, exact and unambiguous. Slides will contain both.

- Data set D, split into D_{train} and D_{test}
  $D_{train} \cup D_{test} = D$
- Data objects/instances/items: $x \in D$. x_{class} represents the class label (i.e., the target category)
- Feature set $F = \{f_1, f_2, \ldots, f_n\}$
  - $v(f_i)$ is a set that contains all possible values of a feature
  - I.e., we know in advance which values a feature can take!
- Feature extractor $f_i(x)$ represents the value of f_i for x

Section 3

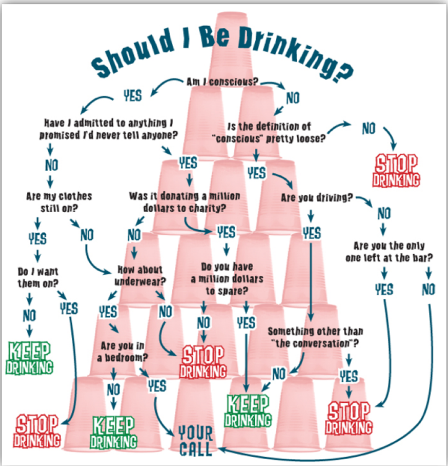
Decision Trees

Decision Trees

Prediction Model – Toy Example

Decision Trees

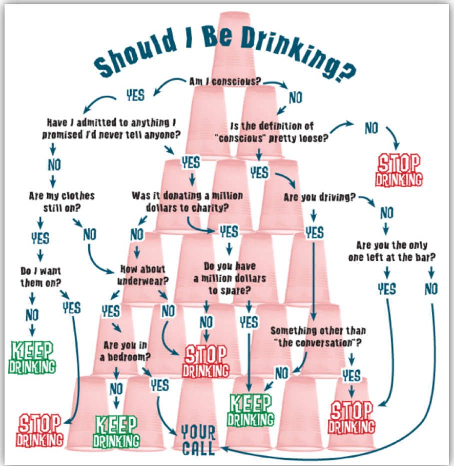
Prediction Model – Toy Example



- What are the instances?

Decision Trees

Prediction Model – Toy Example



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

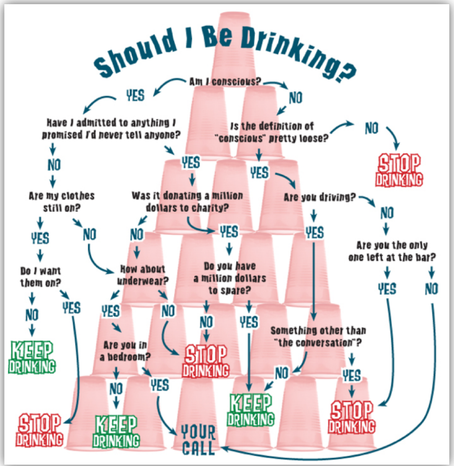
Decision Trees

Prediction Model – Toy Example

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

Decision Trees

Prediction Model – Toy Example



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

Decision Trees

Trees

- Well-established data structure in CS

Decision Trees

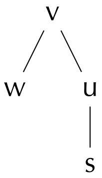
Trees

- 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

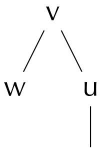
- Well-established data structure in CS
- A tree is a pair that contains
  - some value and
  - a (possibly empty) set of children
    - Children are also trees
- Formally: $\langle v, \{\langle w, \emptyset\rangle, \langle u, \{\langle s, \emptyset\rangle\}\rangle\}\rangle$



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

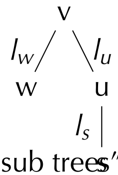


- Formally: $\langle v, \{\langle w, \emptyset\rangle, \langle u, \{\langle s, \emptyset\rangle\}\rangle\}\rangle$
- Recursive definition: "A tree is something and a bunch of sub trees"
  - Recursion is an important ingredient in many algorithms and data structures

Decision Trees

Trees

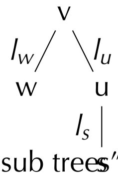
- Well-established data structure in CS
- A tree is a pair that contains
  - some value and
  - a (possibly empty) set of children
    - Children are also trees
- Formally: $\langle v, \{\langle w, \emptyset\rangle, \langle u, \{\langle s, \emptyset\rangle\}\rangle\}\rangle$
- Recursive definition: “A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



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: $\langle v, \{\langle w, \emptyset \rangle, \langle u, \{\langle s, \emptyset \rangle\}\rangle\}\rangle$
- Recursive definition: "A tree is something and a bunch of sub trees"
  - Recursion is an important ingredient in many algorithms and data structures
- If the tree has labels on the edges, the pair becomes a triple
  - $\langle v, l_v, \{\langle w, l_w, \emptyset \rangle, \langle u, l_u \{\langle s, l_s, \emptyset \rangle\}\rangle\}\rangle$



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 ♣♠♢♡
  - (obvious to humans, but needs to be made explicit for the computer)
- 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? $v(f_3) = \{e, o, f\}$
  - Features can be extracted automatically

Disclaimer: This task is artificial, because there is no connection between features and target classes in a full deck (features are evenly distributed).

Decision Trees

Example Task

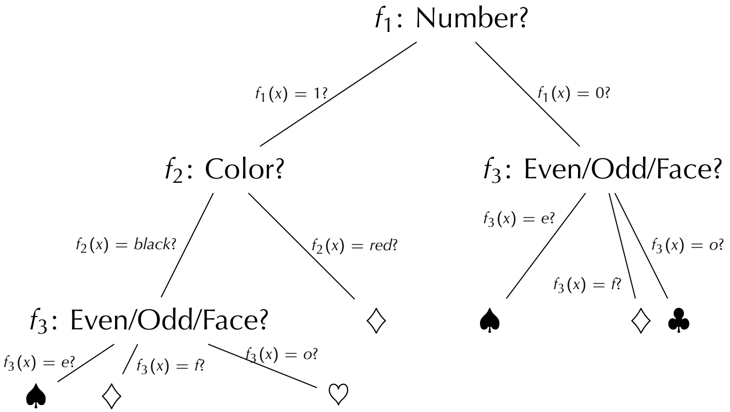


Figure: Example Prediction Model. The model is entirely made up and is not expected to perform well, but it can be used for classification right away.

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)
  - (= the highest reduction in entropy = the highest increase in homogeneity)

Decision Trees

Entropy (Shannon 1948)

$$\underbrace{H(X)}_{\text{entropy}} = -\sum_{i=1}^{\overset{\text{number of classes present in } X}{n}} \overset{\text{relative frequency of the class}}{p(x_i)} \log_b p(x_i)$$

Examples (with $b = 2$)

- $H(\{\spadesuit\spadesuit\spadesuit\spadesuit\}) = -\frac{4}{4}\log_2\frac{4}{4} = 0$
- $H(\{\spadesuit\spadesuit\spadesuit\heartsuit\}) = -\left(\underbrace{\frac{3}{4}\log_2\frac{3}{4}}_{\spadesuit} + \underbrace{\frac{1}{4}\log_2\frac{1}{4}}_{\heartsuit}\right) = 0.562$
- $H(\{\spadesuit\spadesuit\heartsuit\heartsuit\}) = \ldots = 0.693$

Decision Trees

Entropy (Shannon 1948)

$$\underbrace{H(X)}_{\text{entropy}} = -\sum_{i=1}^{\overset{\text{number of classes present in } X}{n}} \overset{\text{relative frequency of the class}}{p(x_i)} \log_b p(x_i)$$

$$\boxed{\begin{array}{c}\log_b(x) = y \\ \text{exactly if} \\ b^y = x\end{array}}$$

Examples (with $b = 2$)

- $H(\{\spadesuit\spadesuit\spadesuit\spadesuit\}) = -\frac{4}{4}\log_2\frac{4}{4} = 0$
- $H(\{\spadesuit\spadesuit\spadesuit\heartsuit\}) = -\left(\underbrace{\frac{3}{4}\log_2\frac{3}{4}}_{\spadesuit} + \underbrace{\frac{1}{4}\log_2\frac{1}{4}}_{\heartsuit}\right) = 0.562$
- $H(\{\spadesuit\spadesuit\heartsuit\heartsuit\}) = \ldots = 0.693$

Decision Trees

Feature Selection (2)

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

/ \

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

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

/ \

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

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

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

Decision Trees

Feature Selection (3)

$$\begin{array}{c} \{\spadesuit\spadesuit\spadesuit\heartsuit\} \\ / \quad \backslash \\ \{\heartsuit\}\{\spadesuit\spadesuit\spadesuit\} \end{array} \qquad\qquad \begin{array}{c} \{\spadesuit\spadesuit\spadesuit\heartsuit\} \\ / \quad \backslash \\ \{\spadesuit\}\{\spadesuit\spadesuit\heartsuit\} \end{array}$$

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

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

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	%
$\spadesuit$	4	33.3
$\diamondsuit$	4	33.3
$\heartsuit$	3	25
$\clubsuit$	1	8.3

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	%
$\spadesuit$	4	33.3
$\diamondsuit$	4	33.3
$\heartsuit$	3	25
$\clubsuit$	1	8.3

$$
\begin{aligned}
H(\spadesuit\spadesuit\spadesuit\spadesuit\diamondsuit\diamondsuit\diamondsuit\diamondsuit\heartsuit\heartsuit\heartsuit\clubsuit) &= H([4, 4, 3, 1]) \\
&= 1.286057
\end{aligned}
$$

Let's Train a Decision Tree

f_1: Does it show a number?

- Splitting D according to f_1 yields
  - Yes: $\{7\clubsuit, 5\diamondsuit, 8\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - No: $\{A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- Intuitively: Is this good?

Let's Train a Decision Tree

f_1: Does it show a number?

- Splitting D according to f_1 yields
  - Yes: $\{7\clubsuit, 5\diamondsuit, 8\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - No: $\{A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- 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
  - Yes: $\{7\clubsuit, 5\diamondsuit, 8\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - No: $\{A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- 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
  - Yes: $\{7\clubsuit, 5\diamondsuit, 8\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - No: $\{A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- 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
  - Red: $\{5\diamondsuit, 8\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - Black: $\{7\clubsuit, A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- 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
  - Red: $\{5\diamondsuit, 8\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - Black: $\{7\clubsuit, A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- 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
  - Red: $\{5\diamondsuit, 8\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - Black: $\{7\clubsuit, A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- 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
  - Red: $\{5\diamondsuit, 8\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - Black: $\{7\clubsuit, A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- 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
  - Even: $\{8\diamondsuit\}$
  - Odd: $\{7\clubsuit, 5\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - Face: $\{A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- 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
  - Even: $\{8\diamondsuit\}$
  - Odd: $\{7\clubsuit, 5\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - Face: $\{A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- 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
  - Even: $\{8\diamondsuit\}$
  - Odd: $\{7\clubsuit, 5\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - Face: $\{A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- 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
  - Even: $\{8\diamondsuit\}$
  - Odd: $\{7\clubsuit, 5\diamondsuit, 3\diamondsuit, 7\diamondsuit, 3\heartsuit, 7\heartsuit, 5\heartsuit\}$
  - Face: $\{A\spadesuit, Q\spadesuit, K\spadesuit, J\spadesuit\}$
- 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

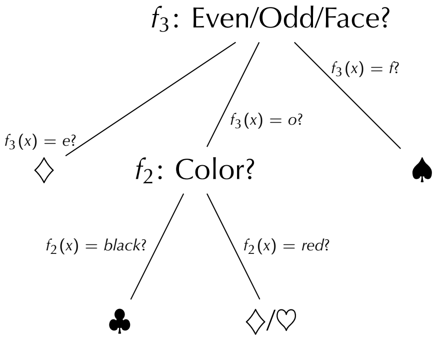


Figure: Final prediction model

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 4

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)

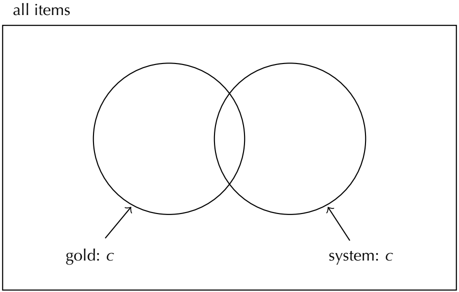


Figure: Identifying true/false positives/negatives

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)

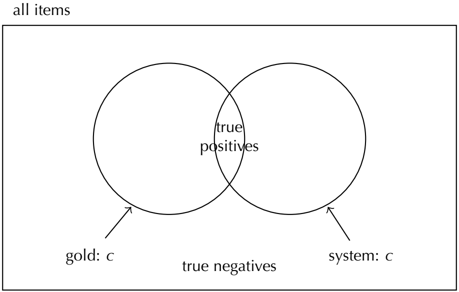


Figure: Identifying true/false positives/negatives

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)

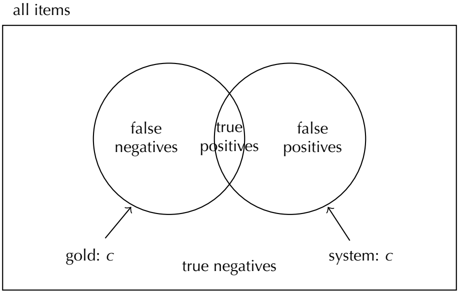
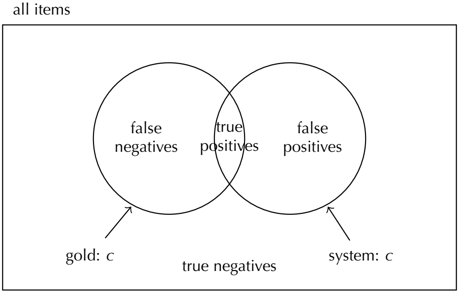


Figure: Identifying true/false positives/negatives

Evaluation

Precision and Recall



true positives Correctly identified items of class c

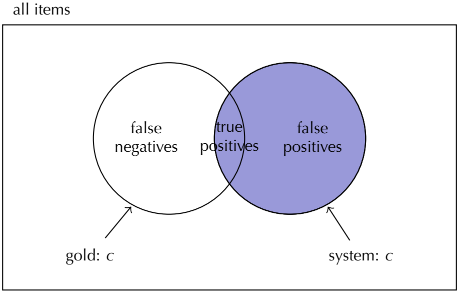
true negatives Correctly identified items of other classes

false positives System predicts c, but it's another class

false negatives System predicts something else, but it's c

Evaluation

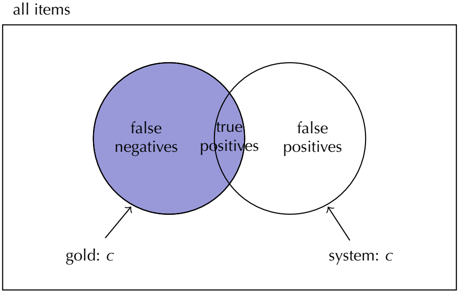
Precision and Recall



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