Feature Engineering

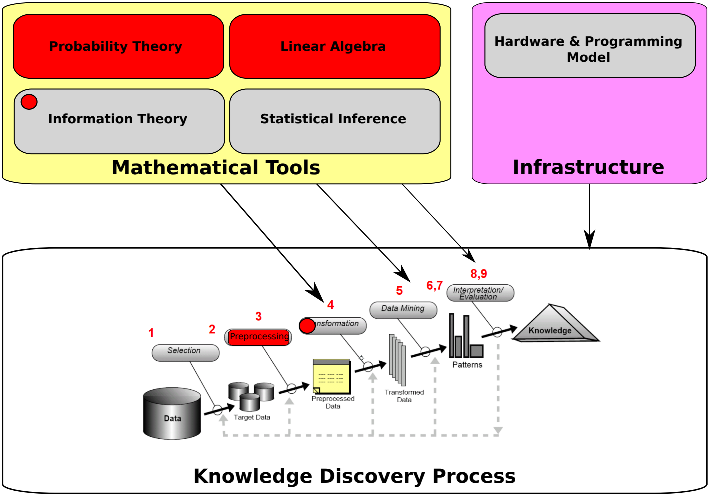
Knowledge Discovery and Data Mining 1

Roman Kern

ISDS, TU Graz

2018-10-25

Big picture: KDDM



Outline

Recap

Review of the preprocessing phase

Recap - Feature Extraction

- Example of features:
- Images → colours, textures, contours, ...
- Signals → frequency, phase, samples, spectrum, ...
- Time series → ticks, trends, self-similarities, ...
- Biomed → dna sequence, genes, ...
- Text → words, POS tags, grammatical dependencies, ...

Features encode these properties in a way suitable for a chosen algorithm

Recap - Feature Extraction

What is Part-of-Speech?

- The process to apply word classes to words within a sentence
- For example
  - Car → *noun*
  - Writing → *noun* or *verb*
  - Grow → *verb*
  - From → *preposition*

Open vs closed word classes

- Prepositions (closed, e.g. “of”, “to”, “in”)
- Verbs (open, e.g. to “google”)

Recap - Feature Extraction

Main approaches for POS tagging

- Rule based
  - ENGTWOL tagger
- Transformation based
  - Brill tagger
- Stochastic
  - HMM tagger

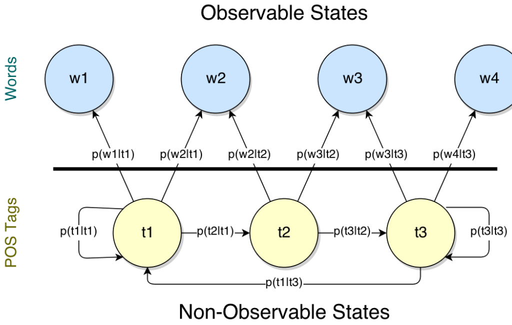
Recap - Feature Extraction

Sequence Tagging - Simplifications

- $\text{argmax}_{t_{1,n}} P(t_{1,n}|w_{1,n}) = \text{argmax}_{t_{1,n}} P(w_{1,n}|t_{1,n})P(t_{1,n})$
  - $\rightarrow$ Not feasible in practice
- Limited horizon & independence assumption:
  $P(t_{1,n}) \approx P(t_n|t_{n-1})P(t_{n-1}|t_{n-2})...P(t_2|t_1) = \prod_{i=1}^{n} P(t_i|t_{i-1})$
- Words only depend on tags: $P(w_{1,n}|t_{1,n}) \approx \prod_{i=1}^{n} P(w_i|t_i)$
- The final equation is:
- $\text{argmax}_{\hat{t}_{1,n}} \prod_{i=1}^{n} P(w_i|t_i)P(t_i|t_{i-1})$

Recap - Feature Extraction

Hidden Markov Models



Needs three matrices as input: A (transmission, POS $\mapsto$ POS), B (emission, POS $\mapsto$ Word), π (initial probabilities, POS)

Recap - Feature Extraction

Probability estimation for tagging

- How do we get such probabilities?
- $\rightarrow$ With supervised tagging we can simply use **Maximum Likelihood Estimation (MLE)** and use counts (C) from a reference corpus
  - $P(t_i|t_{i-1}) = \frac{C(t_{i-1},t_i)}{C(t_{i-1})}$
  - $P(w_i|t_i) = \frac{C(w_i,t_i)}{C(t_i)}$

Smoothing

- To account for unseen words
- Lidstone smoothing: $\frac{C(t_{i-1},t_i)+\lambda}{C(t_{i-1})+\lambda V(t_{i-1},t)}$
- Need to estimate λ, e.g. by held-out data (development data set)

Information Theory

Review of information theory

Entropy

What is Entropy?

- Let X be a discrete random variable with alphabet $\mathcal{X}$ and probability mass function $p(x)$
- $p(x) = Pr\{X = x\}, x \in \mathcal{X}$
- The **entropy** of a variable X is defined as
- $H(X) = -\sum_{x \in \mathcal{X}} p(x) log_2 p(x)$
- ... entropy is a measure for information content of a variable (in bits)

Note 1: By convention $0 log_2 0 = 0$
Note 2: Entropy is the lower bound on the average number of yes/no questions to guess the state of a variable.

Entropy

Entropy Example 1/2

- For example, let $\mathcal{X} = \{A, B, C, D\}$
  - ... each with the same probability of $\frac{1}{4}$
- One can encode each of the values with 2 bits
  - e.g., $A = 00, B = 01, C = 10, D = 11$

Entropy

Entropy Example 2/2

- What if the probabilities are not evenly distributed?
  - e.g., $A = \frac{1}{2}, B = \frac{1}{4}, C = \frac{1}{8}, D = \frac{1}{8}$
- One does only need 1.75 bits to encode
  - e.g., $A = 0, B = 10, C = 110, D = 111$
  - As one expects to see A in 50% of all cases

Entropy

What is Entropy?

- Entropy is a measure for **uncertainty**
- High entropy $\rightarrow$ uniform distribution
  - Histogram of the frequencies would be even
  - Values are hard to predict
- Low entropy $\rightarrow$ peaks and valleys in the distribution
  - Histogram of the frequencies would have spikes
  - Values are easier to predict
- Entropy is always non-negative
- The entropy is always less (or equal) than the logarithm of the alphabet size

Joint Entropy

What is Joint Entropy?

- The **joint entropy** of a pair of discrete random variables $(X; Y)$ with joint probability mass function $p(x; y)$ is defined by
- $H(X, Y) = -\sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p(x, y) log_2 p(x, y)$
  - ... can be generalised to cover more than 2 variables
- $H(X, Y) = H(X) + H(Y)$, if X and Y are independent from each other

Conditional Entropy

What is Conditional Entropy?

- The **conditional entropy** of Y given X is defined as:
- $H(Y|X) = -\sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p(x,y) log_2 p(y|x)$
  - ... how much uncertainty is left, once X is known
- Connection between joint entropy and conditional entropy:
- $H(Y|X) = H(X,Y) - H(X)$

Conditional Entropy

What is Specific Conditional Entropy?

- $H(Y|X=x)$ - the **specific conditional entropy** of Y given a specific value x of X
  - e.g. if $H(Y|X=x)=0$, then x accounts for all the uncertainty of Y
- $H(Y|X)=\sum_{x\in\mathcal{X}} p(x)H(Y|X=x)$

Information Gain

What is Information Gain?

- $IG(Y|X) = H(Y) - H(Y|X)$
  - ... how much is the uncertainty of Y reduced, once X is known
  - or: One has to transmit Y
    - How many bits on average would it save if both ends of the line would know X?

Information Gain

What is Relative Information Gain?

- $RIG(Y|X) = \frac{H(Y) - H(Y|X)}{H(Y)}$
  - One has to transmit Y
    - How many fraction of bits on average would it save if both ends of the line would know X?

Mutual Information

What is Mutual Information?

- The **mutual information** between random variables X and Y with joint probability mass function $p(x, y)$ and marginal probability mass functions $p(x)$ and $p(y)$ is defined as
- $I(X;Y) = \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p(x, y) log_2 \frac{p(x,y)}{p(x)p(y)}$
- The mutual information is a measure of the amount of information that one random variable contains about another random variable
- $I(X;Y) = 0$, if X and Y are independent from each other
- Conditional mutual information:
- $I(X;Y|Z) = H(X|Z) - H(X|Y,Z)$

Pointwise Mutual Information

What is Pointwise Mutual Information?

- The pointwise mutual information is defined as
- $pmi(X = x; Y = y) = i(x; y) = log_2 \frac{p(x,y)}{p(x)p(y)}$
- Can then be normalised to:
- $pmi_{norm}(X = x; Y = y) = \frac{pmi(X=x;Y=y)}{-log_2 p(x,y)}$

Example: For two binary variables:

	y = false	y = true
x = false	$p(\neg x, \neg y)$	$p(\neg x, y)$
x = true	$p(x, \neg y)$	$p(x, y)$

Relative Entropy

What is Relative Entropy?

- The relative entropy or between two probability mass functions $p(x)$ and $q(x)$ is defied by:
- $D(p||q) = \sum_{x \in X} p(x) log_2 \frac{p(x)}{q(x)}$
- ... also called Kullback-Leibler distance
- The relative entropy is always non-negative and zero if and only if $p = q$
- Connection between mutual information and relative entropy:
- $I(X;Y) = D(p(x,y)||p(x)p(y))$

Note: By convention $0 log_2 \frac{0}{0} = 0$, $0 log_2 \frac{0}{q} = 0$ and $p log_2 \frac{p}{0} = \infty$

Entropy Overview

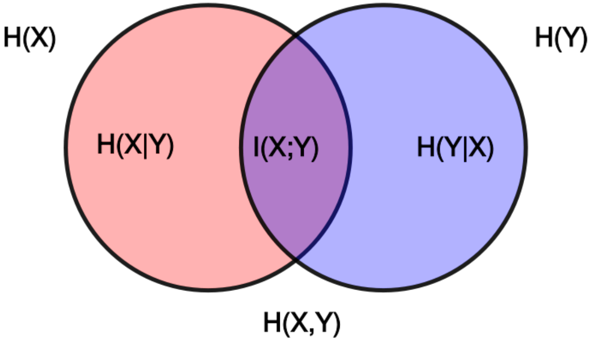


Figure: Overview of entropy, joint entropy, conditional entropy and mutual information (source: Wikipedia)

Markov Chains

Markov chains

Random variables X, Y, Z are said to form a Markov chain in that order (denoted $X \rightarrow Y \rightarrow Z$) if the conditional distribution of Z depends only on Y and is conditionally independent of X, ie if the joint probability mass function can be written as:

$$p(x, y, z) = p(x)p(y|x)p(z|y)$$

$X \rightarrow Y \rightarrow Z$ if and only if X and Z are conditionally independent given Y.

Markov chains and information theory

If $X \rightarrow Y \rightarrow Z$ then $I(X; Y) \geq I(X; Z)$

Introduction

What are features & feature engineering

Introduction

What is feature engineering?

The act to inject knowledge into a machine learning model.

What are features?

The items, that represent this knowledge suitable for machine learning algorithms.

What is a machine learning model?

The model represents the output of the learning process (knowledge representation)

Note: there is no formal definition of feature engineering

Introduction

Tasks of feature engineering

1. Understand the properties of the task - how they might interact with the strength and limitations of the model
2. Experimental work - test expectations and find out what actually works

Note: The exploration vs. experimental work characterises many data science scenarios

Introduction

Process of feature engineering

- Remove unnecessary features
- Remove redundant features
- Create new features
  - Combine existing features
  - Transform features
  - Use features from the context
  - Integrate external sources
- Modify feature types
  - e.g. from binary to numeric
- Modify feature values

Note: depending on the task and the algorithms the results might differ

Feature Engineering Goals

Goals

The task also depends on the goal of feature engineering:

1. If the goal is to get the best prediction accuracy
2. ... or an explainable model

Feature Engineering Terminology

Important Terms

Feature Set Set of features used for a task

Feature Space High dimensional space spawned by the features (range of the feature values)

Instance Single assignment of features and values (an example)

Introduction - Example

ARFF-Viewer - /home/rkern/apps/weka/weka-3-7-7/data/contact-lenses.arff

File Edit View

contact-lenses.arff

Relation: contact-lenses

No.	1: age Nominal	2: spectacle-prescrip Nominal	3: astigmatism Nominal	4: tear-prod-rate Nominal	5: contact-lenses Nominal
1	young	myope	no	reduced	none
2	young	myope	no	normal	soft
3	young	myope	yes	reduced	none
4	young	myope	yes	normal	hard
5	young	hypermetrope	no	reduced	none
6	young	hypermetrope	no	normal	soft
7	young	hypermetrope	yes	reduced	none
8	young	hypermetrope	yes	normal	hard
9	pre-presbyopic	myope	no	reduced	none
10	pre-presbyopic	myope	no	normal	soft
11	pre-presbyopic	myope	yes	reduced	none
12	pre-presbyopic	myope	yes	normal	hard
13	pre-presbyopic	hypermetrope	no	reduced	none
14	pre-presbyopic	hypermetrope	no	normal	soft
15	pre-presbyopic	hypermetrope	yes	reduced	none
16	pre-presbyopic	hypermetrope	yes	normal	none
17	presbyopic	myope	no	reduced	none
18	presbyopic	myope	no	normal	none

Figure: Features to predict which type of contact lens is most appropriate (none, soft, hard)

Introduction - Example

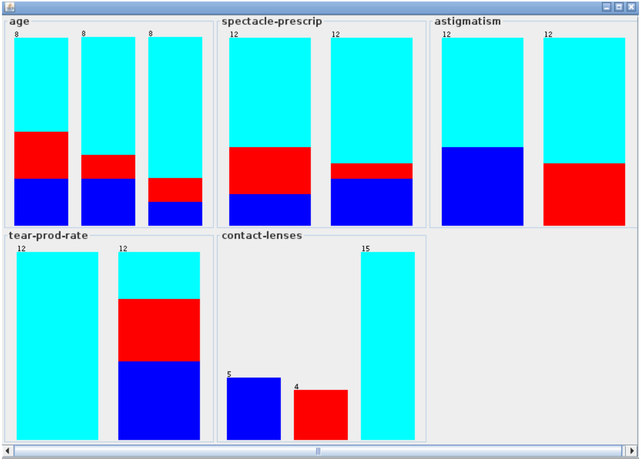


Figure: Relation between the features with the contact lens type

Introduction - Example

ver - /home/rkern/apps/weka/weka-3-7-7/data/we

File Edit View

weather.arff

Relation: weather

No.	1: outlook Nominal	2: temperature Numeric	3: humidity Numeric	4: windy Nominal	5: **play** Nominal
1	sunny	85.0	85.0	FALSE	no
2	sunny	80.0	90.0	TRUE	no
3	overcast	83.0	86.0	FALSE	yes
4	rainy	70.0	96.0	FALSE	yes
5	rainy	68.0	80.0	FALSE	yes
6	rainy	65.0	70.0	TRUE	no
7	overcast	64.0	65.0	TRUE	yes
8	sunny	72.0	95.0	FALSE	no
9	sunny	69.0	70.0	FALSE	yes
10	rainy	75.0	80.0	FALSE	yes
11	sunny	75.0	70.0	TRUE	yes
12	overcast	72.0	90.0	TRUE	yes
13	overcast	81.0	75.0	FALSE	yes
14	rainy	71.0	91.0	TRUE	no

Figure: Features to decide whether to play or not based on the weather

Introduction

- Features are assigned to instances
  - No dependencies/relationships between instances (in practical applications)
  - Thus relationships need to be flatted out (denormalisation)
  - ... by modelling features
- Complex features need to be "simplified"
  - ... by creating aggregate, compound features, e.g. by using averages
- Typically features have no fixed sequence (hence the term set)
  - ... by creating features that express sequence information

Feature Value Processing

Operations upon feature values...

Feature Processing

Feature binarisation

- Threshold numerical values to get boolean values
- Needed as some algorithms just take boolean features as input

Feature discretization

- Convert continuous features to discrete features
- Equal sized partitions? Equal interval partitions?

Feature value transformation

- Scaling of values
- Move the centre

Feature Normalisation

- Normalise the value of features
- For example, most of the features are in the range [0..1],
  - ... but one ranges [-1000 .. +1000]
- Classifiers like SVM struggle with this (other algorithms do not need this step, e.g. decision trees)

Feature Weighting

- Given numeric or binary features
  - ... encode their impact into the feature value
- Can be seen as prior of a feature
  - e.g. "term weighting" to separate potentially words with grammatical function from word with a semantic function

Feature Engineering for Text Mining

Tactics when dealing with text

Contextual Features

Bigram Features

- When working with single words as features, often the sequence information is lost
  - ... but, this could potentially a source of information
- $\rightarrow$ introduce new feature as a combination of two adjacent words

Contextual Features

Example Bigram Features

- Example: `The quick brown fox jumps over the lazy dog`
- Unigram features: `brown`, `dog`, `fox`, `lazy`, ...
- Bigram features: `brown_fox`, `fox_jumps`, `lazy_dog`, `over_the`, ...

Contextual Features

n-grams

- Bigrams can be extended for more than two words
- $\rightarrow$ n-grams
- Can be extended to allow gap in between words (skip n-grams)

Contextual Features

Character n-grams

- n-gram can be created on words, but on characters as well
- e.g. `The quick brown fox jumps over the lazy dog`
- Character tri-grams: `the`, `qui`, `uic`, `ick`, `bro`, `row`, `own`, ...

External Sources

Integrate external sources

- Integrate evidence from external sources
    - e.g. WordNet for semantic relations
- Example: `The quick brown fox jumps over the lazy dog`
- Features: `the`, `quick`, `brown`, `fox`, **`canine`**, **`canid`**, `jumps`, ...
- Added *canine* and *canid* from the hypernyms found in WordNet

External Sources

Noun

- S: (n) **fox** (alert carnivorous mammal with pointed muzzle and ears and a bushy tail; most are predators that do not hunt in packs)
  - *direct hyponym* / *full hyponym*
  - ***direct hypernym*** / *inherited hypernym* / *sister term*
    - S: (n) canine, canid (any of various fissiped mammals with nonretractile claws and typically long muzzles)
- S: (n) dodger, **fox**, slyboots (a shifty deceptive person)
- S: (n) **fox** (the grey or reddish-brown fur of a fox)
- S: (n) **Fox**, Charles James Fox (English statesman who supported American independence and the French Revolution (1749-1806))
- S: (n) **Fox**, George Fox (English religious leader who founded the Society of Friends (1624-1691))
- S: (n) **Fox** (a member of an Algonquian people formerly living west of Lake Michigan along the Fox River)
- S: (n) **Fox** (the Algonquian language of the Fox)

Verb

- S: (v) flim-flam, play a joke on, play tricks, trick, fob, **fox**, pull a fast one on, play a trick on (deceive somebody) *"We tricked the teacher into thinking that class would be cancelled next week"*
- S: (v) confuse, throw, **fox**, befuddle, fuddle, bedevil, confound, discombobulate (be confusing or perplexing to; cause to be unable to think clearly) *"These questions confuse even the experts"; "This question completely threw me"; "This question befuddled even the teacher"*
- S: (v) **fox** (become discolored with, or as if with, mildew spots)

Figure: Wordnet entry for the word fox, the first sense contains the hypernyms canine and canid.

Feature Selection

Less is more - sometimes...

Feature Selection

Feature engineering can be classified into two use cases

- Modelling for **prediction accuracy**
  - Default, if the goal is to have a productive system
  - ... with optimal performance
- Modelling for **explanations**
  - When the model should be easy to interpret
  - ... one can acquire better knowledge of the problem
  - ... and then to improve the feature engineering task

Feature Selection

If a model uses **fewer features**

- ... it is easier to interpret
- ... it will generalise better (less risk of overfitting)
- ... it will be faster (more efficient)

Feature Selection

Curse of Dimensionality

- The problem of having too many features
- More features make the model more expressive
  - but not all of the features are relevant
- The higher the dimensionality, the higher the chances of spurious features

Feature Selection

Approaches towards reducing the complexity

- Feature selection
- Regularisation

Feature Selection

Feature selection

- Approach: select the sub-set of all features without redundant or irrelevant features
- Set-of-all-subset problem $\rightarrow$ NP hard
- Need to find more practical approaches
  - Unsupervised, e.g. heuristics
  - Supervised, e.g. using a training data set

Feature Selection

- Simple approach $\rightarrow$ use heuristics
- Black & white lists
    - ... list contains features, which either should not be used
    - ... or an exclusive list of features

Feature Selection

Example for black list

- Stop-word list for textual features
  - ... list of frequent word, that carry little semantics
  - e.g. the, you, again, can, ...
- Advantage: simple, yet effective
- Disadvantage: some may carry semantic, used in phrases or named entities ("The The"), homonyms (can.v vs. can.n)

Feature Selection

Unsupervised approach

- Unsupervised ranked feature selection
- Scoring function to rank the feature according to their importance
  - ... then just use the top 5% (10%, 25%, ...)
  - e.g. for textual data use the frequency of words within a reference corpus

Feature	Count	Freq.
the	3,032,573	0.879
in	2,919,623	0.846
a	2,903,352	0.841
of	2,888,379	0.837
is	2,639,282	0.765
and	2,634,096	0.763
⋮	⋮	⋮
with	1,703,251	0.494

Table: Top 50 word within the Wikipedia, the top ranked word (the) occurs in 88% of all instances.

Feature Selection

Supervised approaches

- Filter approaches
- Wrapper approaches

Feature Selection

Supervised approaches

Filter approaches

- Compute some measure for estimating the ability to discriminate between classes
- Typically measure feature weight and select the best n features $\rightarrow$ supervised ranked feature selection
- Problems:
  - Redundant features (correlated features will all have similar weights)
  - Dependant features (some features may only be important in combination)

Feature Selection - Information Gain

Information gain as ranking function

- Recall: $IG(Y|X) = H(Y) - H(Y|X)$
- Select features by IG
  1. Compute the IG for each feature
  2. Rank the features based on IG
  3. Select the top-k features

Example

Features on contact lenses

```
Ranked attributes:
 0.5488  4 tear-prod-rate
 0.377   3 astigmatism
 0.0395  2 spectacle-prescrip
 0.0394  1 age
```

Feature Selection

Supervised approaches

Wrapper approaches

- Search through the space of all possible feature subsets
- Each search subset is tried out with a learning algorithm

Feature Selection - Wrapper Approach

Wrapper approach

- General algorithm:
  1. Initial subset selection
  2. Try a subset with a learner
  3. Modify the feature subset
  4. Rerun the learner
  5. Measure the difference
  6. GOTO 2
- Advantages: combination of features, ignore redundant/irrelevant features
- Disadvantage: computationally intensive
- 2 basic ways for i) initial subset selection, ii) modification of subset: forward selection and backward elimination

Feature Selection - Wrapper Approach

Forward Selection

- Start with empty set
- Add each feature not in set
- Pick the one with the highest increase
- Stop if there is no increase

Backward Elimination

- Start with full feature set
- Try to remove features

Feature Interactions

- From the field of statistics
  - e.g., remove redundant features based on correlation
  - e.g., Principal Component Analysis (PCA)
- Non-trivial relationship between variables (features)
  - Link to Information Theory, e.g., Interaction Information

Regularisation

- Basic idea: Introduce a penalty for **complexity of a model**
- The more features, the higher the complexity
  - e.g., if the number of feature exceeds the number of observations
- Typically the regularizer is integrated into the cost function (or loss function)
- Example (negative log-likelihood): $cost(f) = -l(f) + regularizer(f)$
  - e.g. L_0 ... taking the number of non-zero features, L_1 ... sum of the feature values, ...

Feature Transformation

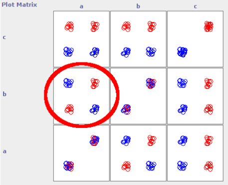
- Some algorithms can only solve certain problems
- e.g. Perceptrons apply only to linearly separable data
- XOR problem for single layer perceptrons
- $\rightarrow$ two main approaches: i) non-linear transformation of the features, ii) more sophisticated algorithms

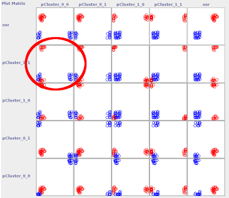
Feature Transformation

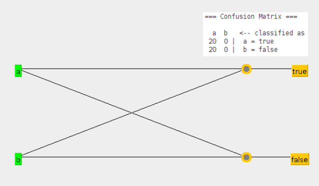
Feature transformation

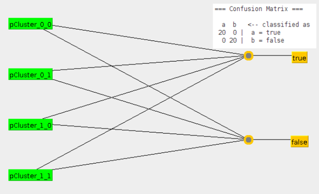
- Map features into high-dimensional space
    - Create more features
- The more features, the higher the dimensionality
- The higher the dimensionality, the higher the chances that the problem is linearly separable

Feature Transformation









Left: original features, which cannot be separated by a single layer perceptron;
Right: features transformed into a higher dimensional space, is linear separable

Feature Transformation

Kernel trick

- Some algorithms employ a scalar product of the features (e.g. SVMs)
- Transform into higher dimensionality "on-the-fly"
  - ... by introducing a (kernel) function
- Original: $< x, y >$, with kernel function: $\varphi(x, y)$
- Number of different well-known kernel functions (e.g. Gaussian kernel)
  - ... which often require parameters (to tune)

Thank You!

Next up: Data Matrices

Further information

http://www.cs.cmu.edu/~awm/tutorials
http://www.icg.isy.liu.se/courses/infotheory/lect1.pdf
http://www.cs.princeton.edu/courses/archive/spring10/cos424/slides/18-feat.pdf
http://ufal.mff.cuni.cz/~zabokrtsky/courses/npfl104/html/feature_engineering.pdf
http://www.ke.tu-darmstadt.de/lehre/archiv/ss06/web-mining/wm-features.pdf