Machine Learning for Computational Linguistics

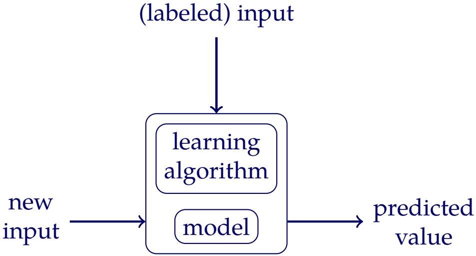
Summary

Çağrı Çöltekin

University of Tübingen
Seminar für Sprachwissenschaft

July 19, 2016

Machine learning



- Machine learning is about making predictions based on past observations
- No explicit programming, better handling of uncertainty
- We do not want to memorize, but generalize

Types of machine learning

Machine learning methods are roughly classified as

- Supervised learning requires a ‘labeled’ training data
- Unsupervised learning is about finding (latent) structure in unlabeled data
- Various notions of the two of the above exists, known as semi-supervised, self-supervised, …
- In reinforcement learning, the feedback to the system is delayed

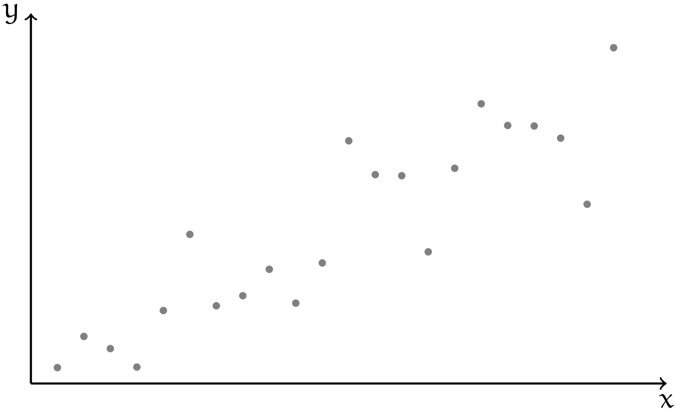
Boundaries can sometimes be difficult to draw.

Regression – classification

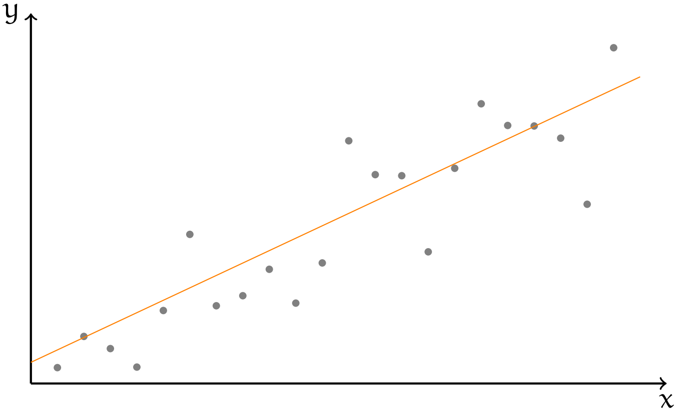
We often distinguish the machine learning methods based on what they predict

- For supervised methods,
  - regression predicts a continuous value
  - classification predicts a class label
- For unsupervised methods
  - dimensionality reduction finds continuous latent variables
  - clustering aims to discover (discrete) groups in the data

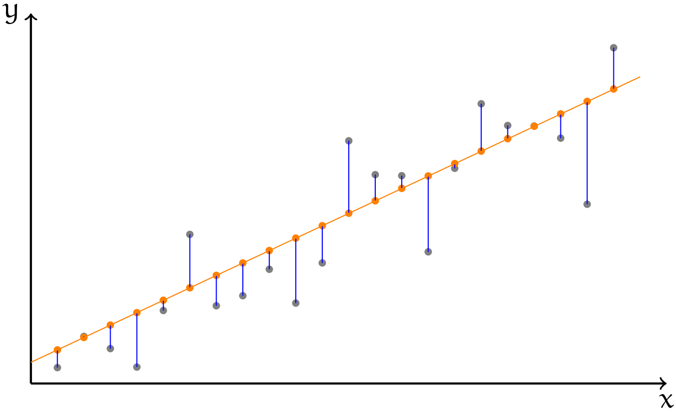
Regression



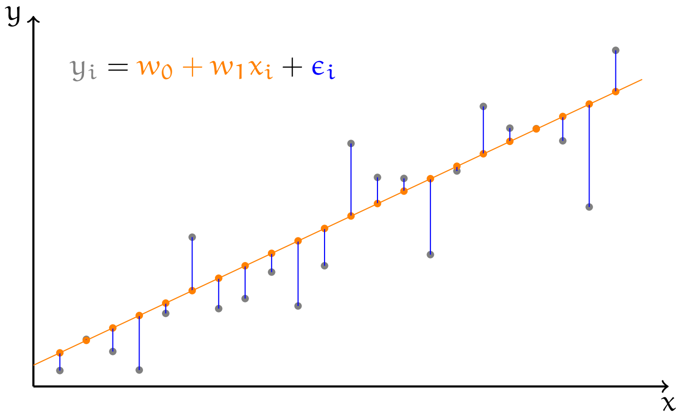
Regression



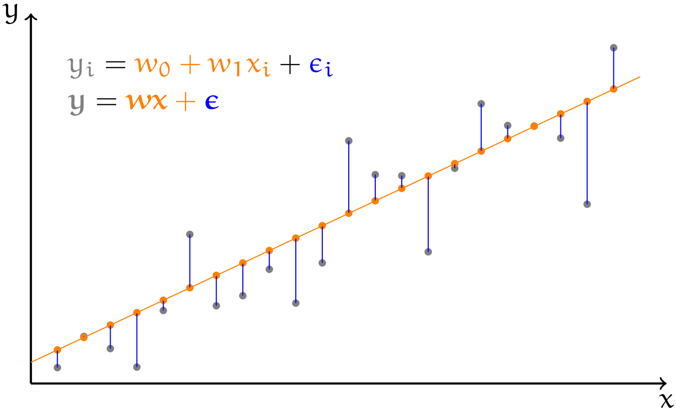
Regression



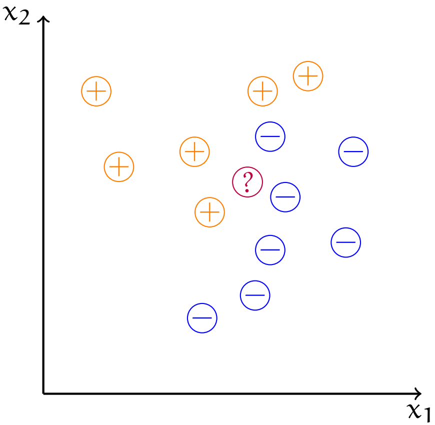
Regression



Regression

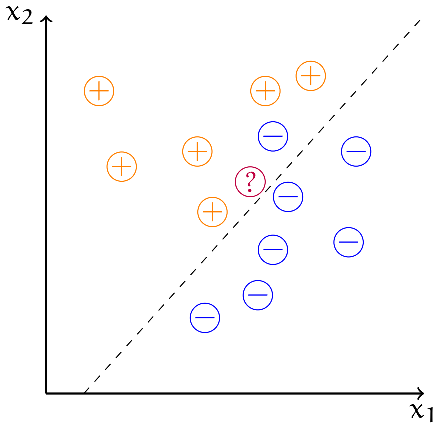


Classification



- The response (outcome) is a label. In the example: positive ⊕ or negative ⊖
- Given the features (x_1 and x_2), we want to predict the label of an unknown instance ?
- A classification algorithm finds a function that separates the classes
- Most classification methods can easily be extended to multi-class problems

Classification



- The response (outcome) is a label. In the example: positive ⊕ or negative ⊖
- Given the features (x_1 and x_2), we want to predict the label of an unknown instance ?
- A classification algorithm finds a function that separates the classes
- Most classification methods can easily be extended to multi-class problems

Logistic regression

A basic classification algorithm is logistic regression. Logistic regression is an extension of linear regression (a GLM)

$$g(p(\boldsymbol{y})) = \boldsymbol{X}\boldsymbol{w} + \boldsymbol{\epsilon}$$

- In logistic regression, we try to predict the probability of the positive/target class given the predictors
- $g()$ is the logit function, $\boldsymbol{\epsilon}$ is distributed binomially
- Alternatively, we can write the prediction of the model as

$$p(y) = \frac{1}{1 + e^{-\boldsymbol{w}\boldsymbol{x}}}$$

Note: in this notation we assume a constant input $+1$, whose coefficient is the intercept (or bias)

Training a supervised model

- Learning in a supervised model means setting the model parameters $\boldsymbol{w}$
- Typically, training is formulated as an optimization problem: we define an error function, and minimize it
- The error function is often derived such that it increases the likelihood of the data given the model parameters
- For linear regression, this turns out to be the sum of the squared error
- For logistic regression, cross entropy

Minimizing the error function

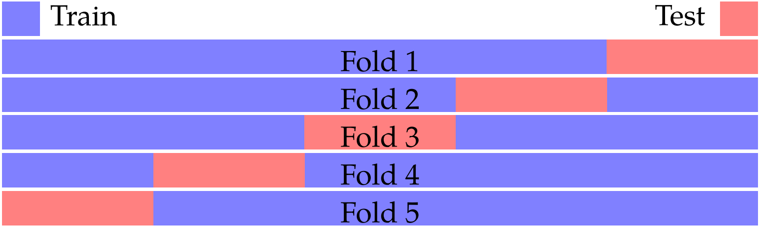
Once we define an error function $\mathsf{E}(w)$, as a function of the parameters,

- we may be able to find an unique analytic solution (linear regression)
- if $\mathsf{E}(w)$ is convex, an iterative method can find the global minimum (logistic regression)
- if $\mathsf{E}(w)$ is not convex, then find the global minimum is often not possible, we try to find a 'good enough' local minimum (neural networks)

Evaluating supervised learning methods

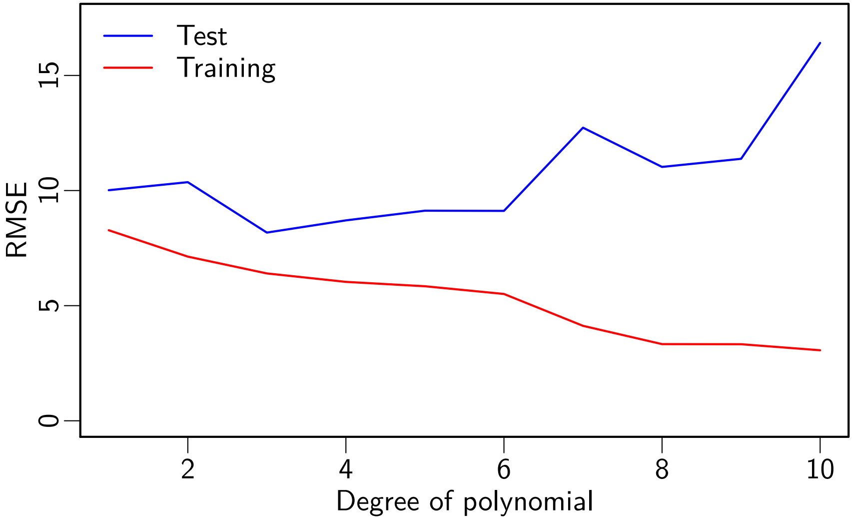
- Our aim is to fit models that are (also) useful outside the training data
- Evaluating a model on the training data is wrong: complex models tend to learn idiosyncrasies of the training data (e.g., noise)
- Success in training data does not necessarily transfer to the out of training instances
- We always evaluate our models using data outside the training set

K-fold Cross validation



- At each fold, we hold part of the data for testing, train the model with the remaining data
- Typical values for k is 5 and 10
- In stratified cross validation each fold contains (approximately) the same proportions of class labels.
- A special case, when k is equal to n (the number of data points is called leave-one-out cross validation

Training/test error



Overfitting and underfitting

- A model with high capacity can overfit the to the training data: low training error, high test error
- An overfitted model finds a too specific solution
- A model with low capacity may underfit: the model cannot approximate the target function well
- An underfitted model finds a too general solution

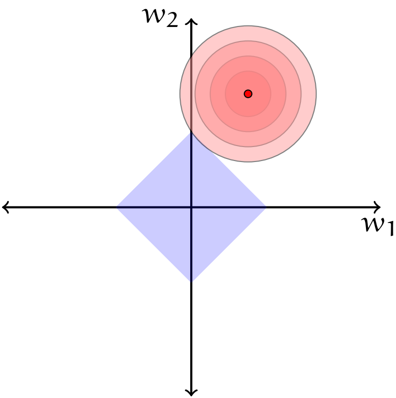
Simplicity is good for a model (prevents overfitting), but not simpler than necessary.

Regularization

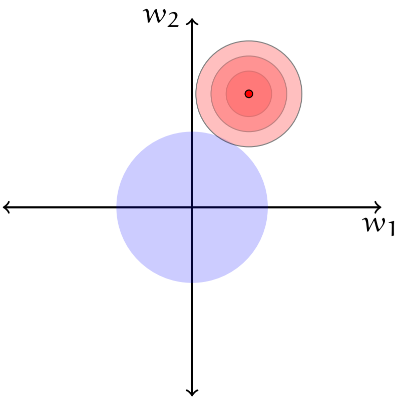
- A common solution to overfitting is to use a regularization term in the objective function
- Common choices are minimizing L2 or L1 norm of the parameters together with the error function
- Regularization prevents overfitting by constraining the model
- The *hyperparameter* λ needs to be determined (best value is found typically using *grid search*, on an additional partition of the data often called *development* set)
- The regularization terms can be interpreted as *priors* in a Bayesian setting
- Particularly, L2 regularization is equivalent to a normal prior with zero mean

L1 and L2 regularization

L1: $J(w) = E(w) + \lambda \|w\|_1$



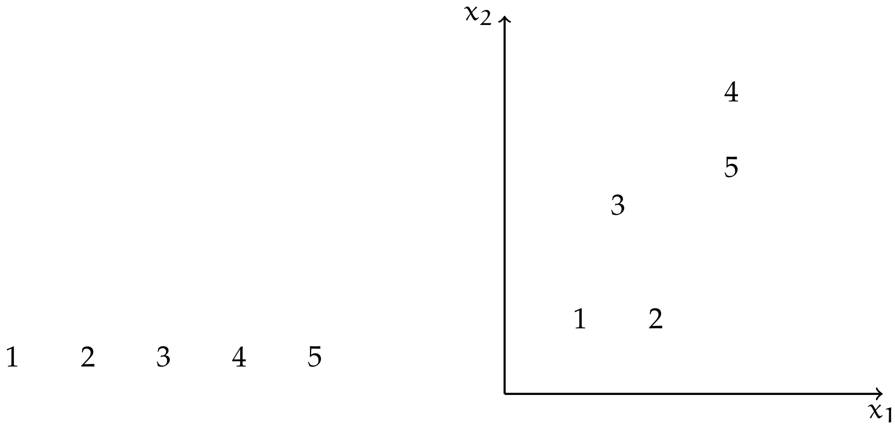
L2: $J(w) = E(w) + \lambda \|w\|_2$



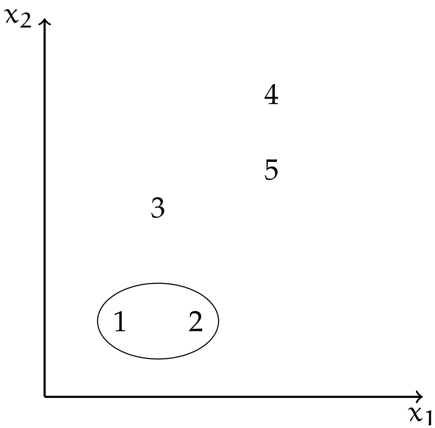
Unsupervised learning

- In unsupervised learning, we do not have labels
- The aim is to discover structure in the data
- Clustering aims to find groups in the data
- Dimensionality reduction expresses a high-dimensional data with a lower dimension while preserving most of the information

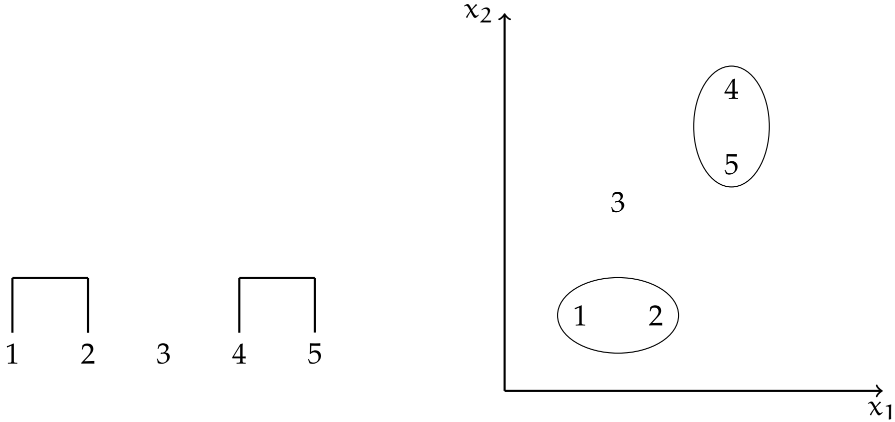
Clustering: hierarchical



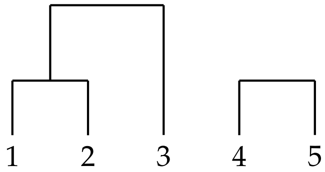
Clustering: hierarchical

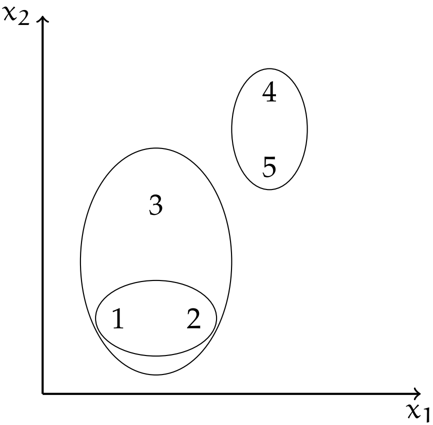


Clustering: hierarchical

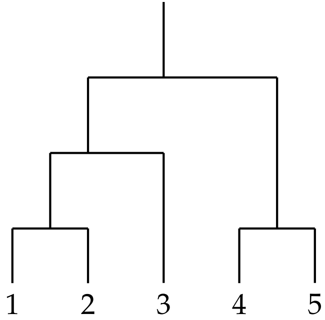


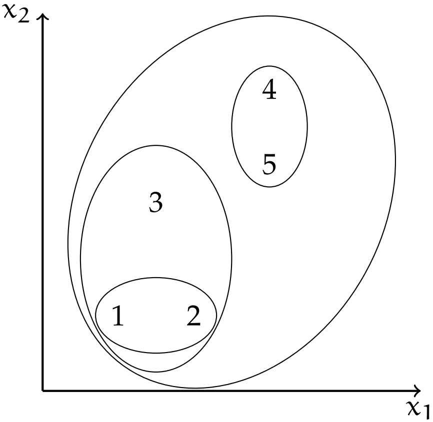
Clustering: hierarchical



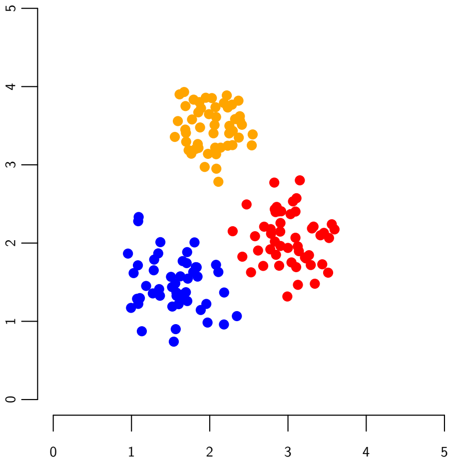


Clustering: hierarchical



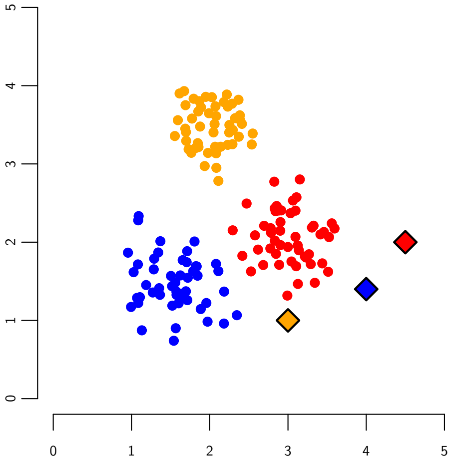


Clustering: k-means



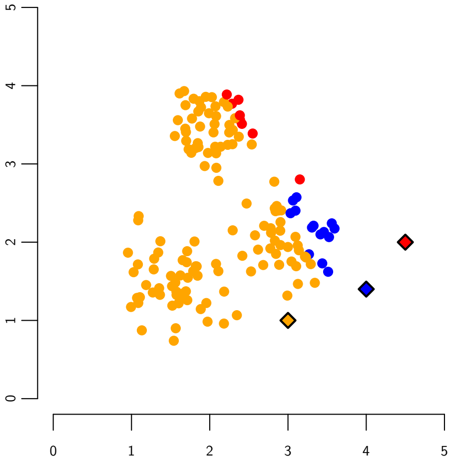
- The data
- Set cluster centroids randomly
- Assign data points to closest centroid
- Recalculate the centroids

Clustering: k-means



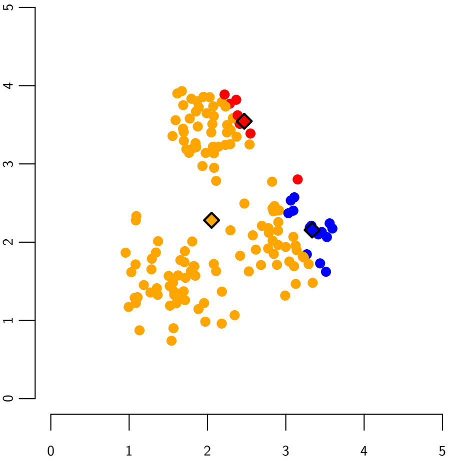
- The data
- Set cluster centroids randomly
- Assign data points to closest centroid
- Recalculate the centroids

Clustering: k-means



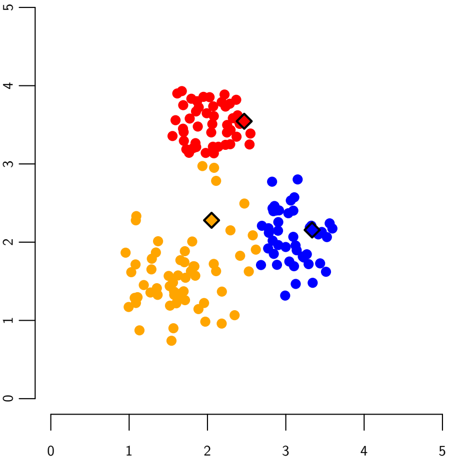
- The data
- Set cluster centroids randomly
- Assign data points to closest centroid
- Recalculate the centroids

Clustering: k-means



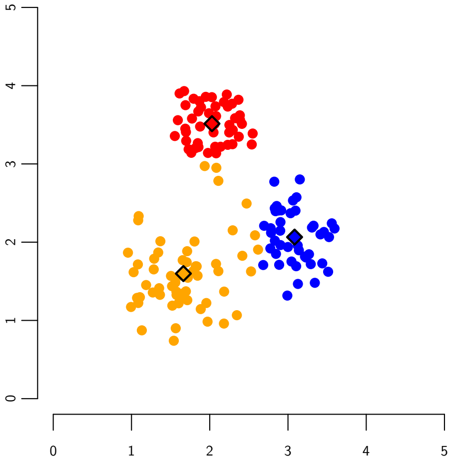
- The data
- Set cluster centroids randomly
- Assign data points to closest centroid
- Recalculate the centroids

Clustering: k-means



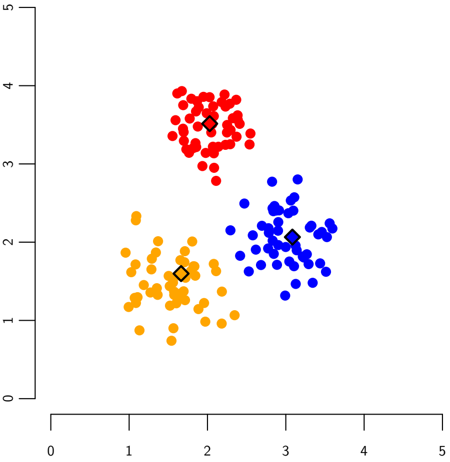
- The data
- Set cluster centroids randomly
- Assign data points to closest centroid
- Recalculate the centroids

Clustering: k-means



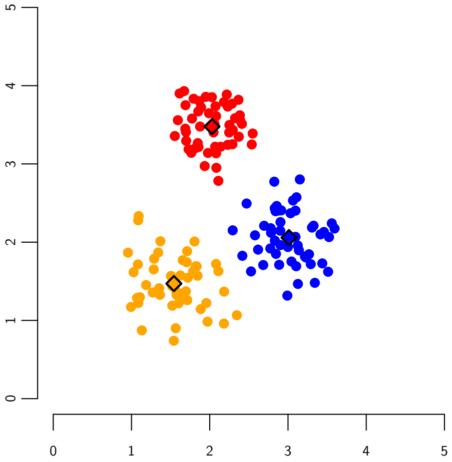
- The data
- Set cluster centroids randomly
- Assign data points to closest centroid
- Recalculate the centroids

Clustering: k-means



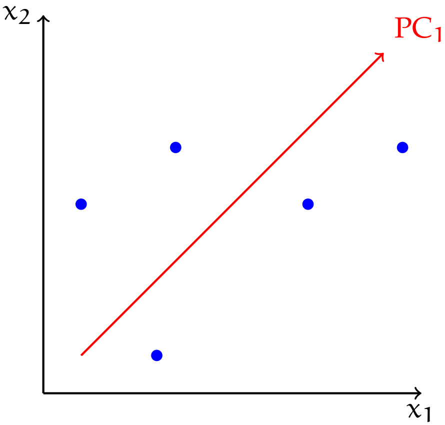
- The data
- Set cluster centroids randomly
- Assign data points to closest centroid
- Recalculate the centroids

Clustering: k-means



- The data
- Set cluster centroids randomly
- Assign data points to closest centroid
- Recalculate the centroids

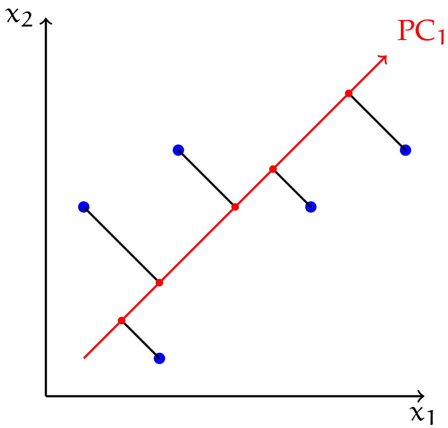
Principal component analysis



PCA can be viewed as

- finding the direction of the largest variance

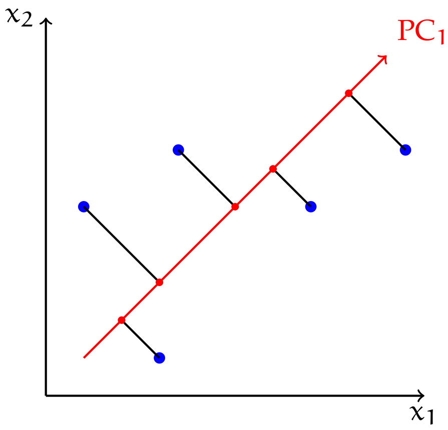
Principal component analysis



PCA can be viewed as

- finding the direction of the largest variance
- finding the projection with the least reconstruction error

Principal component analysis



PCA can be viewed as

- finding the direction of the largest variance
- finding the projection with the least reconstruction error
- finding a lower dimensional latent Gaussian variable such that the observed variable is a mapping of the latent variable to a higher dimensional space (with added noise).

Non-linear relationships

In a linear model, $y = w_0 + w_1 x_1 + \ldots + w_k x_k$

- The outcome is *linearly-related* to the predictors
- The effects of the inputs are *additive*

This is not always the case:

- Some predictors affect the outcome in a non-linear way
  - The effect may be strong or positive only in a certain range of the variable (e.g., age)
  - Some effects are periodic (e.g., many measures of time)
- Some predictors interact
  '*not bad*' is not '*not*' + '*bad*' (e.g., for sentiment analysis)

Dealing with non-linearities

- Non-linear transformations, kernels, feature engineering
  - Note that both

$$y = w_0 + w_1 x_1 + w_2 x_1^2$$

    and

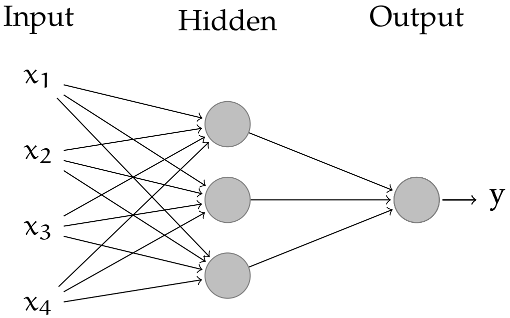
$$y = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_1 x_2$$

    are still linear in weights.

- Artificial neural networks

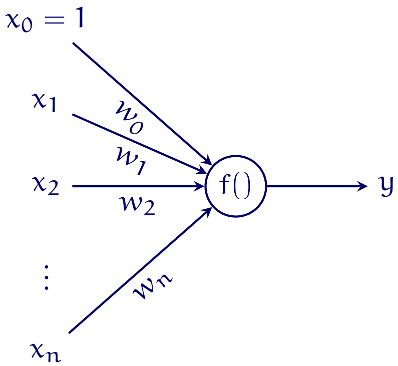
Fully-connected feed-forward networks

Multi-layer perceptron



- Multi-layer perceptron (a fully-connected network with a single hidden layer) is a universal function approximator
- The network can be trained using backpropagation algorithm

Artificial neuron



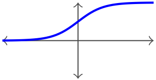
- Every unit in an ANN performs a simple operation: apply a *activation function*, $f()$, to weighted sum of its inputs.

$$y = f(\boldsymbol{wx})$$

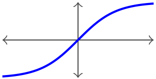
- Typical activation functions include
  - Logistic sigmoid
  - Hyperbolic tangent (tanh)
  - ReLU

Activation functions in neural networks

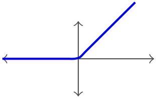
- Output layer activation is determined based on the function of the network
  - linear functions for regression
  - logistic sigmoid for binary classification
  - softmax for multi-class classification
- Common hidden layer activation functions are



Logistic sigmoid $\frac{1}{1+e^{-x}}$

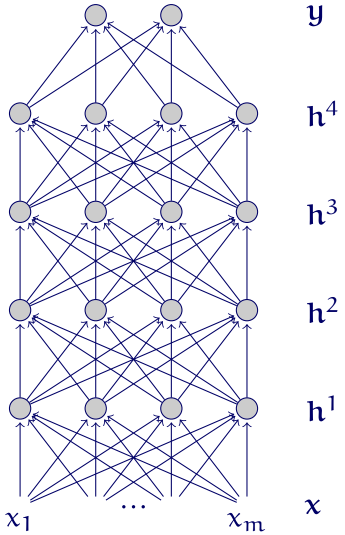


Hyperbolic tangent (tanh) $\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$



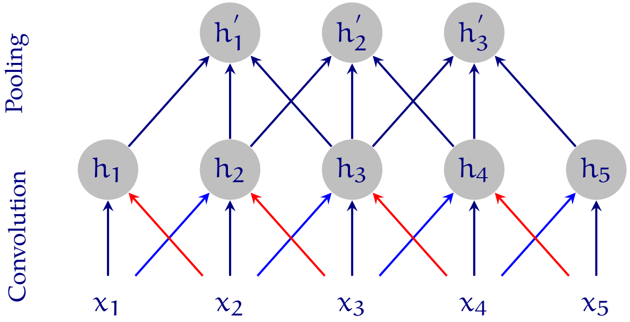
ReLU $\max(0, x)$

Deep neural networks



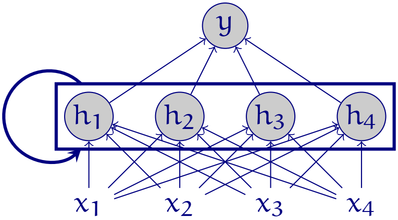
- $\mathbf{x}$ is the input vector
- $\mathbf{y}$ is the output vector
- $\mathbf{h}^1 \dots \mathbf{h}^m$ are the hidden layers (learned/useful representations)
- Deep neural networks are particularly useful if problem can be solved by a hierarchy of features
- Problems in learning: vanishing or exploding gradients

Convolutional neural networks (CNNs)



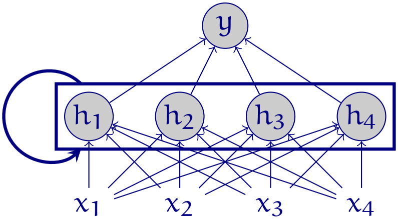
- Convolution transforms input by replacing each input unit by a weighted some of its neighbors
- Typically it is followed by pooling
- CNNs are useful to detect local features with some amount of location invariance
- Sparse connectivity makes CNNs computationally efficient

Recurrent neural networks



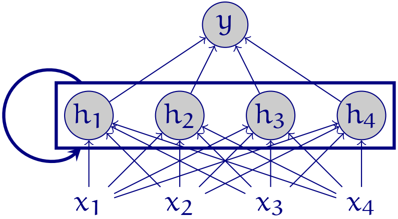
- Recurrent neural networks are similar to the standard feed-forward networks

Recurrent neural networks



- Recurrent neural networks are similar to the standard feed-forward networks
- But they include loops that feed the previous output (of the hidden layers) back to the hidden layer

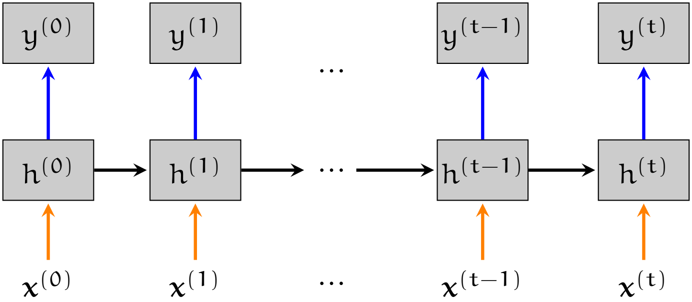
Recurrent neural networks



- Recurrent neural networks are similar to the standard feed-forward networks
- But they include loops that feed the previous output (of the hidden layers) back to the hidden layer
- Forward calculation is straightforward, learning becomes somewhat tricky

Unrolling a recurrent network

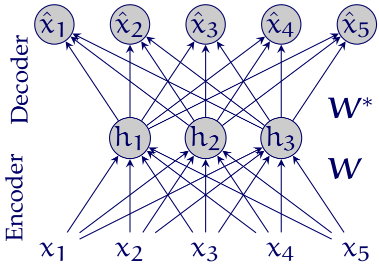
Back propagation through time (BPTT)



Gated recurrent neural networks

- Recurrent networks are suitable for sequence learning
- However, they cannot hold the information for long: long-distance dependencies are difficult to capture
- Gated recurrent networks (e.g., LSTMs) keep solve this problem by explicitly storing and removing information

Unsupervised learning in ANNs



- Autoencoders (figure) trained to predict their input
- Another alternative is Restricted Boltzmann machines (RBMs)
- The aim is to learn useful representations of input at the hidden layer
- It is common to train multiple hidden layers using unsupervised methods, and use them as features in a classifier (layer-wise greedy training)

On variants of gradient descent

$$w \leftarrow w - \alpha \sum_{i}^{n} \nabla J_i(w)$$

- A more efficient approach is to use *stochastic gradient descent*, updating weights for each training instance ($w \leftarrow w - \alpha \nabla J(w)$)
- Often a compromise between to two (*mini-batch*) is used
- We often do not want to keep learning rate fixed, a (lenear) *decay*
- There are some algorithms that update the learning rate (α) update the learning rate in smarter ways (*adagrad*, *adam*, *rmsprop*)
- Sometimes applying a *momentum* is useful, which uses a weighted average of gradients, instead of the gradient calculated at a single point

Things we did not cover

- Many (classification) methods, notably
  - Support vector machines
  - Rule learning, decision trees, random forests
  - Naive Bayes
  - ...
- Probabilistic (Bayesian) inference and learning
- Sequence models (e.g., HMMs)

Projects

- You are strongly encouraged to discuss your project with me soon: please schedule an appointment
- Try simpler models first, add complexity if needed
- You do not have to produce state-of-the-art results, however,
  - make an effort to get good results
  - use proper methodology, evaluation methods/metrics
- You can simplify the data/task if you do not have the necessary computational resources
- Your results should be reproducible
- Use of a version management system (e.g., git) is strongly recommended

Term papers

- You are required to report your results in a term paper
- Make sure you reserve enough time for writing it
- Use ACL 2016 style files for your term papers
  `http://acl2016.org/files/acl2016.zip`
- You paper should not be longer than 6 pages (excluding references)
- No lower limit, but, make sure you sufficiently
  - introduce the problem
  - describe the model(s), data, evaluation procedure
  - present and discuss your results
- If writing a paper is new for you, the Internet is full of wisdom on how to write term papers, make use of them
- Submit your paper via email not later than Sept 15