Data Analysis and Machine Learning

From data to wisdom

What is data?

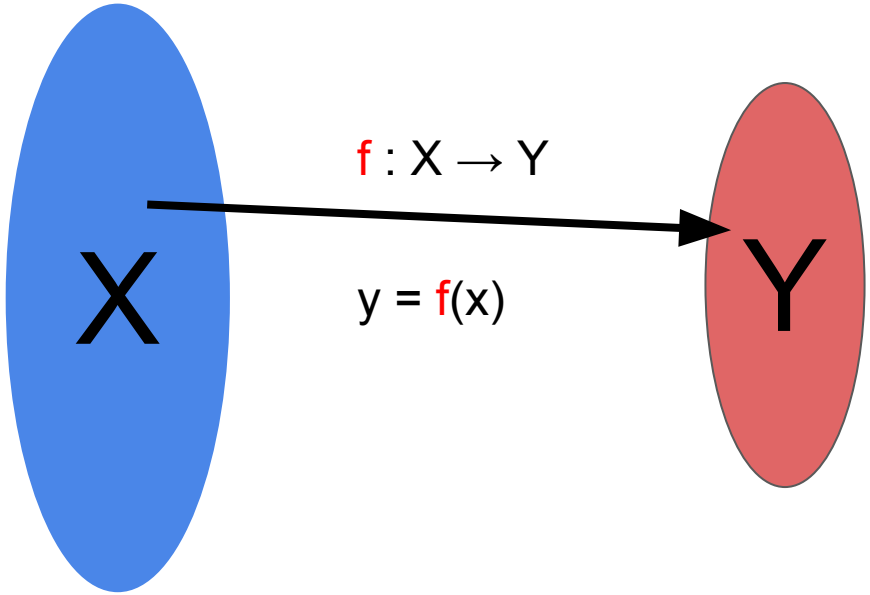
2

2

- Numeric data
  - Continuous: 2.000
  - Discrete: 2
- Categorical
  - Male/Female
  - Employed/student
- Ordinal
  - Stars in movie ranking
  - Placement in competition
- Interval
  - Distance between two values
  - Technically numeric data is an interval from 0
- Text

Machine Learning

- Formally machine learning is creating a mapping *f* from set X to set Y
- Boundary conditions define the type of machine learning
  - If we have examples of correct input-output pairs the problem is **supervised learning**
  - If we don't, the problem is generally **unsupervised learning**



Supervised learning

Regression

- Y is numerical
  - Input mostly numerical
- E.g.
  - Predict weight based on height
  - Predict car gas mileage based on engine size and weight
  - Predict credit risk based on bank account information
  - Predict dollars spent on product X based on available data

Classification

- Y is categorical (or maybe ordinal)
- Input is mostly numerical
- E.g.
  - Predict gender based on weight/height
  - Predict party affiliation based on income
  - Predict if customer belongs to the risk *class*
  - Predict car type based on price and engine size
  - Predict if tumor is benign/malignant based on measurements

Unsupervised learning

No example mappings from X to Y are known

Dimension reduction

- Y is a space of lower dimensionality
  - E.g. 4-dimensional → 2-dimensional
- It is usually desired that the space have certain qualities
  - Orthonormality
  - Optimality in a particular way

Clustering

- Y is a group label based on some (hopefully) natural grouping in the data
- The number of groups to be found in the data can be difficult to determine

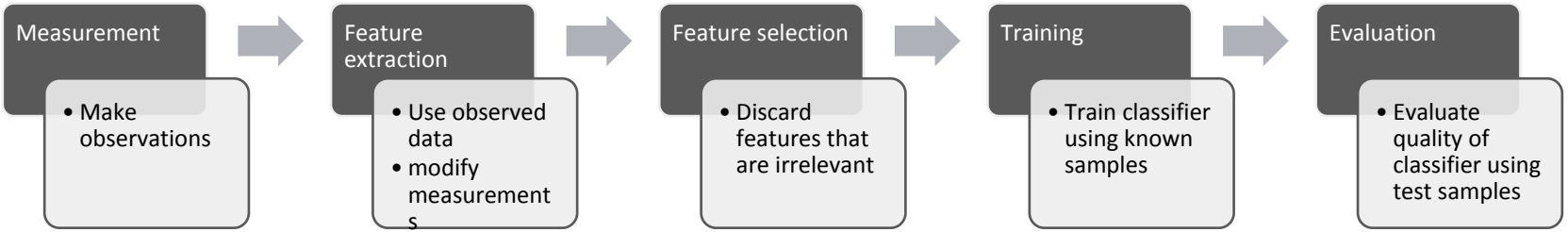
Reinforcement Learning

- Reinforcement is a special case of supervised learning
- Heavily influenced by nature
- Well suited for use on (large) streams of data
- No mappings between input and output are given
- Every time the system performs an action it receives a reward
- Algorithm optimizes towards maximum reward

Preprocessing

- Sometimes X isn’t a simple numeric vector
  - Usenet newsgroup posts
  - Image
  - Sound
- “Measure what is measurable, and make measurable what is not so” - *quote mistakenly attributed to Galileo Galilei*

General structure of supervised learning systems



Practical tasks

Recommender systems

Detecting fraud

Search engines

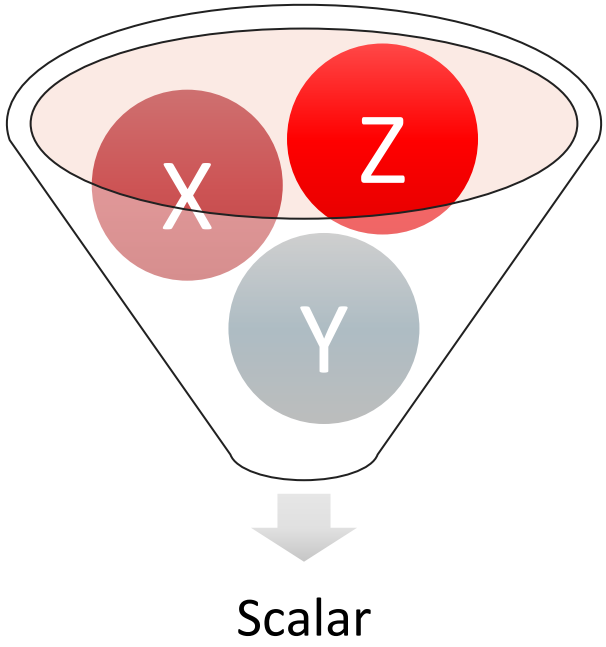
Optimizing advertising

Natural language processing

Spam filtering

Regression

Regression



Regression

- In regression we predict a scalar variable based on one or more other variables
  - Predict weight based on height
  - Predict car gas mileage based on engine volume and weight
- We typically limit ourselves to a model and accept that the model doesn't capture the entire phenomenon → some error is acceptable

$$Y = \hat{f}(x) + \epsilon$$

Error in Regression

- In regression tasks we typically minimize the Mean Squared Error (MSE)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2$$

Error in Machine learning

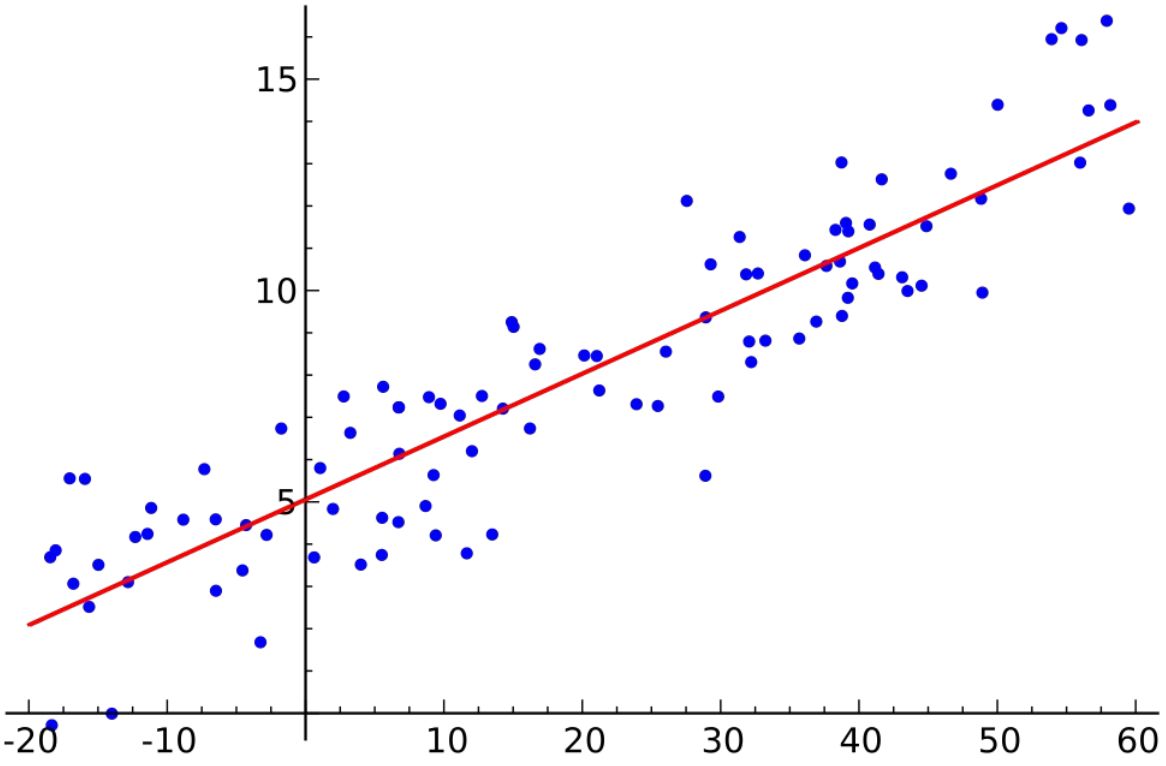
- Let's assume we have
- We want to estimate so we create a predictor
- The expectation of the squared error that we want to minimize is

$$Y = f(X) + \epsilon$$

$$\hat{Y} = \hat{f}(X)$$

$$E(Y - \hat{Y})^2 = [f(X) - \hat{f}(X)]^2 + Var(\epsilon)$$

Linear regression



Linear Regression

- In linear regression we assume that the relationship is of the form
- We then use least squares to estimate the two variables

$$Y \approx \beta_0 + \beta_1 X + \epsilon$$

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X$$

Quality metrics in linear regression

- Residual Standard Error
  - A measure of *lack of fit*
- R^2
  - Equals to correlation in the single variable case
  - A sort of generalization of correlation in the multivariate case
- The F-statistic

$$RSE = \sqrt{\frac{1}{n-2}\sum_{ni=1}(y_i - \hat{y}_i)^2}$$

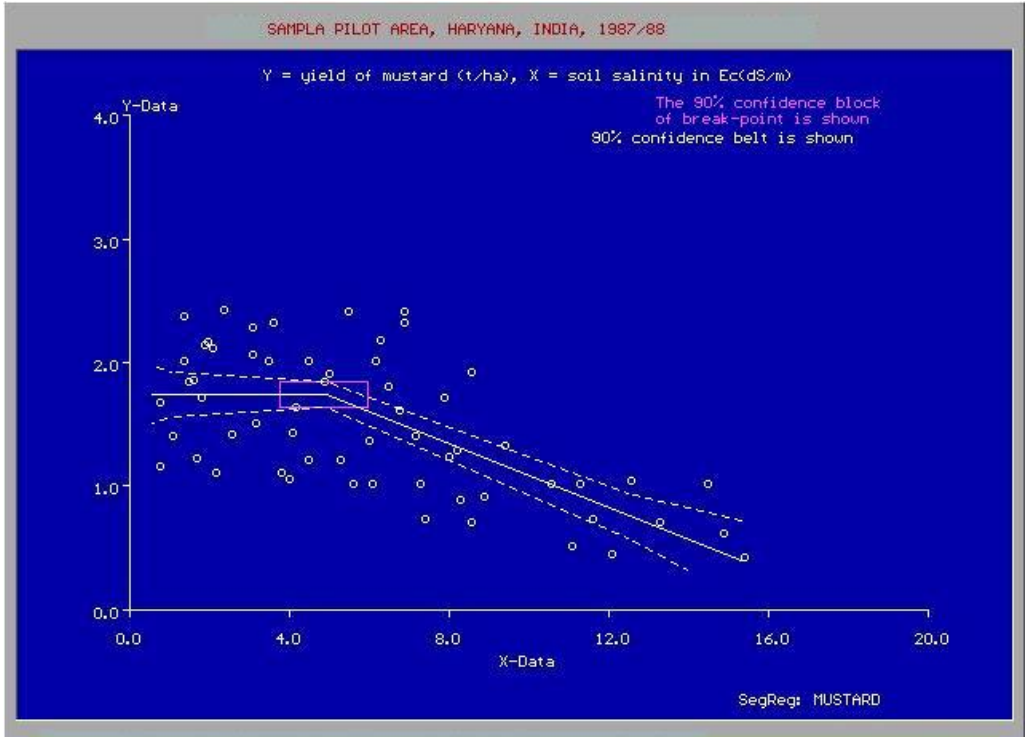
$$R^2 = 1 - \frac{\sum(y - \hat{y})^2}{\sum(y - mean(y))^2}$$

It generalizes to more variables!

Ways to uncover to potential issues in regression

- Residual plot
  - Relationship is not linear
  - Error terms correlate
  - Error terms have non-constant variance
- Plot original data
  - Outliers
  - High leverage points
- Plot correlations
  - Collinearity

Nonlinear regression

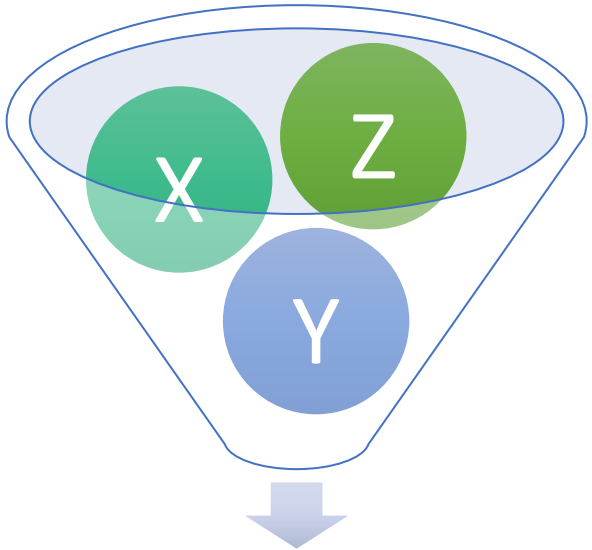


We can, but should we?

http://www.bbc.com/news/magazine-37658374

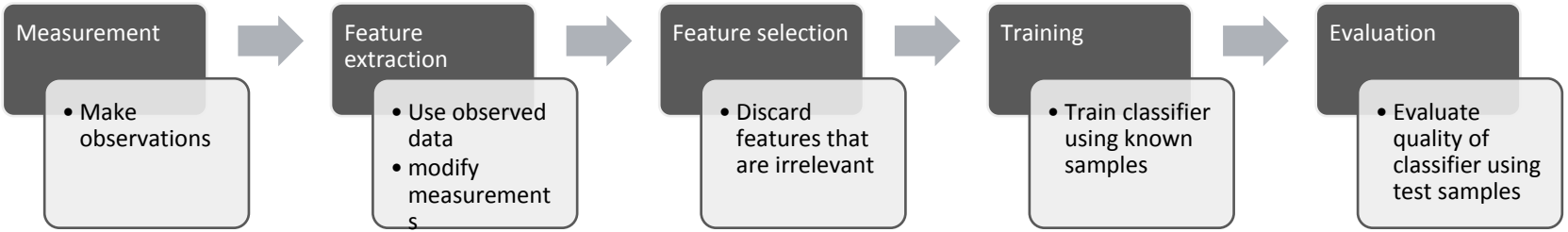
Classification

Classification



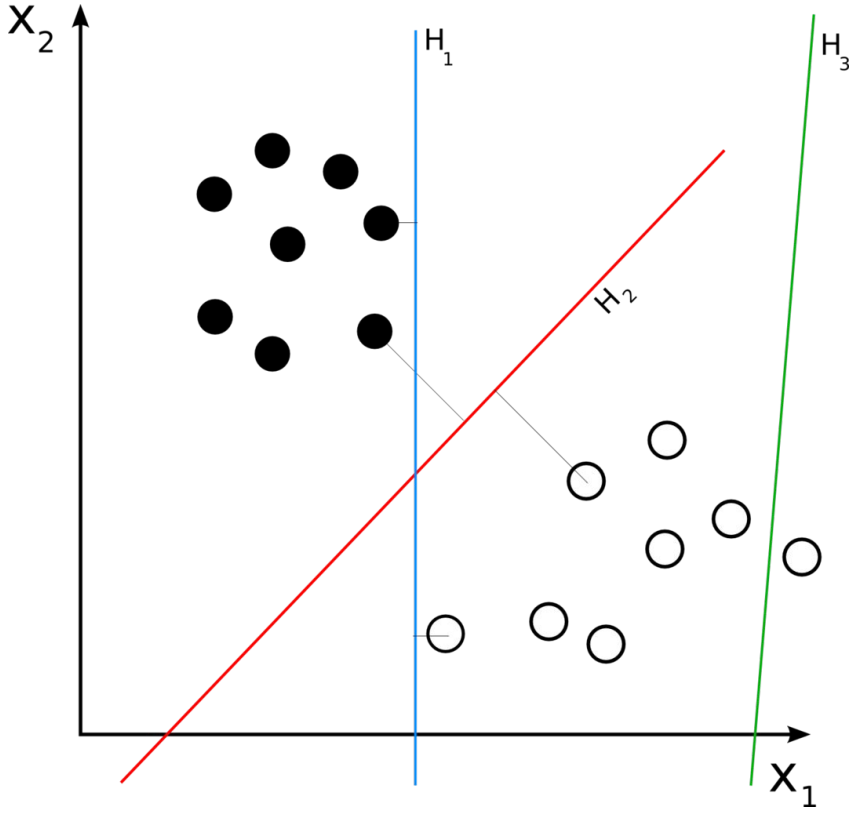
Categorical

General structure of classification systems



Classification

- Two-class classification is finding a hyperplane (H) between s.t. samples on one side are classified as belonging to one class and samples on the other are classified as belonging to the other
  - N-class case can be handled e.g. by creating N classifiers
- If it is possible to construct H so that the split is perfect, we say that the classes are **linearly separable**
  - This is not nearly always the case
  - Typically there is at least noise in the data



Logistic regression

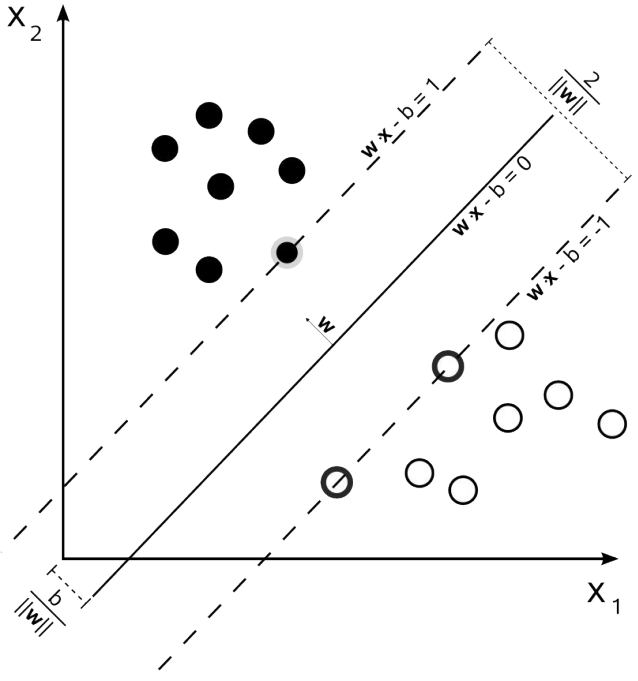
- Like regression, except form an estimate of the probability of belonging to class 1
  - Use *logistic function*
- Form an estimate using Maximum Likelihood (ML)
  - Formulation left as homework :)
- If estimated probability > 0.5, choose class 1, otherwise choose class 2

$$p(X) = \beta_0 + \beta_1 X = \frac{e^{\beta_0+\beta_1 X}}{1 + e^{\beta_0+\beta_1 X}}$$

$$\hat{p}(X) = \frac{e^{\hat{\beta}_0+\hat{\beta}_1 X}}{1 + e^{\hat{\beta}_0+\hat{\beta}_1 X}}$$

Support Vector Machine

- Support Vector Machines maximize the distance of the nearest samples to the decision hyperplane
- The nearest samples are the so-called **support vectors**



Other paradigms

- Naive Bayes approach
  - A mighty sword to wield
- K-nearest neighbor
  - Beautiful and intuitive
  - Computationally not so fun
- All draw a decision border between classes in space
  - Might not be a hyperplane

Nearest Neighbors

- Intuitively simple:
  - Find the k nearest samples in training set
  - Choose most common (or similar) class
- Performs surprisingly well against much more sophisticated systems
- Generalizes reasonably
- Choice of k can have an effect
- Produces very edgy and ugly decision borders
- Query time in default implementation is *O[DN]*, which is intractable for large groups
- Doesn’t generalize, just remembers

Distance metrics for continuous variables

- Euclidean $\sqrt{x^2 + y^2}$
- Manhattan $\sum |x - y|$
  - “Taxicab distance”
- Chebyshev distance $max(|x_i - y_i|)$

There are others for e.g. integers, boolean variables, even strings

Meditating about distance metrics is often necessary nearest neighbor classification if your data is strange

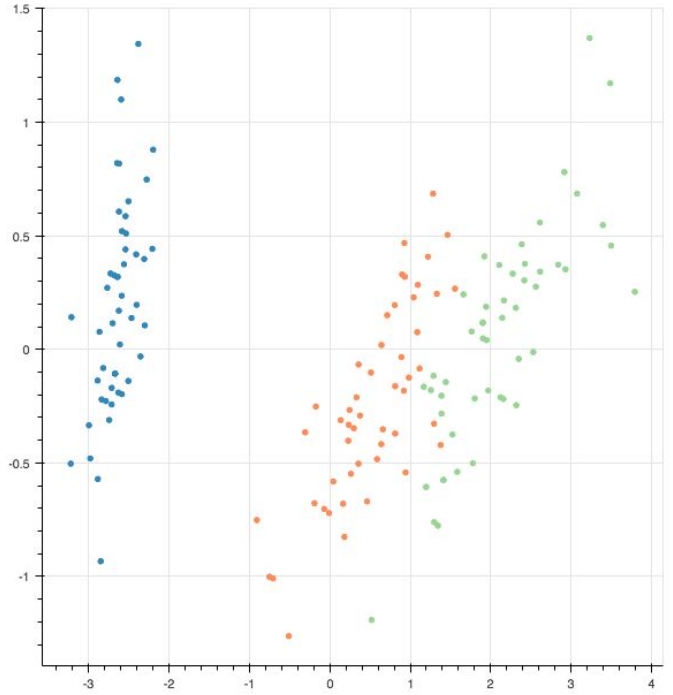
Iris

Petal = terälehti
Sepal = verholehti

Iris = Kurjenmiekka
Iris setosa = kaunokurjenmiekka
Iris versicolor = kirjokurjenmiekka
Iris virginica = virginiankurjenmiekka?
(Virginiassa esiintyvä kurjenmiekka)

Real-life data

- Class on the left is clearly separable
- Two other classes can’t be separated by a simple line
  - They are not **linearly separable**
- We must choose to minimize some error
- Alternately optimize for some metric
- A confusion matrix is often created



Confusion matrix

- A table containing all correct and incorrect classifications represents the behaviour of a classification system
- A number of metrics can be optimized for, depending on what the goals of the system are

		Predicted	
		Class 1	Class 2
Actual class	Class 1	10	3
	Class	4	20

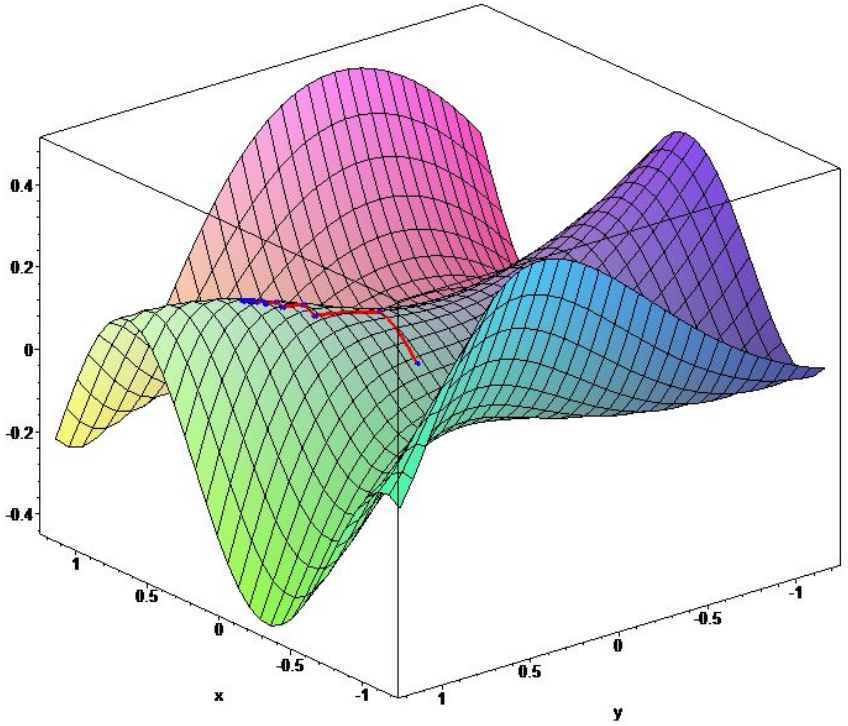
Confusion matrix

- The **accuracy** can be determined as the number of candies actually liked in the bag plus number of candies disliked not in the bag divided by all the types of candy
- The **precision** is the candies in the bag that are liked divided by all candies in the bag
- The **recall** is the number of candies liked in the bag divided by the number of candies in the store
- There are many more that can be relevant
- How could you optimize for each?

You have to choose candies from a pick'n'mix to match another person's taste. I.e. for each type of candy decide if it belongs to class *likes* or *dislikes.* Bag the likes, leave the dislikes

		Predicted	
		Likes	Dislikes
Actual class	Likes	10	3
	Dislikes	4	20

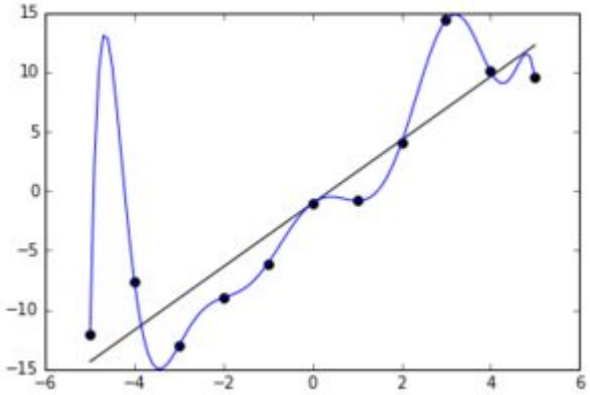
Gradient descent



- Most machine learning algoritms form a function to optimize
- For SVM it was the distance to support vectors
- The algorithm starts at a random point and takes iterative steps to find the minimum of the function
- Gradient == multidimensional derivative
- You could say this is the learning part in machine learning
    - Really it’s just optimization

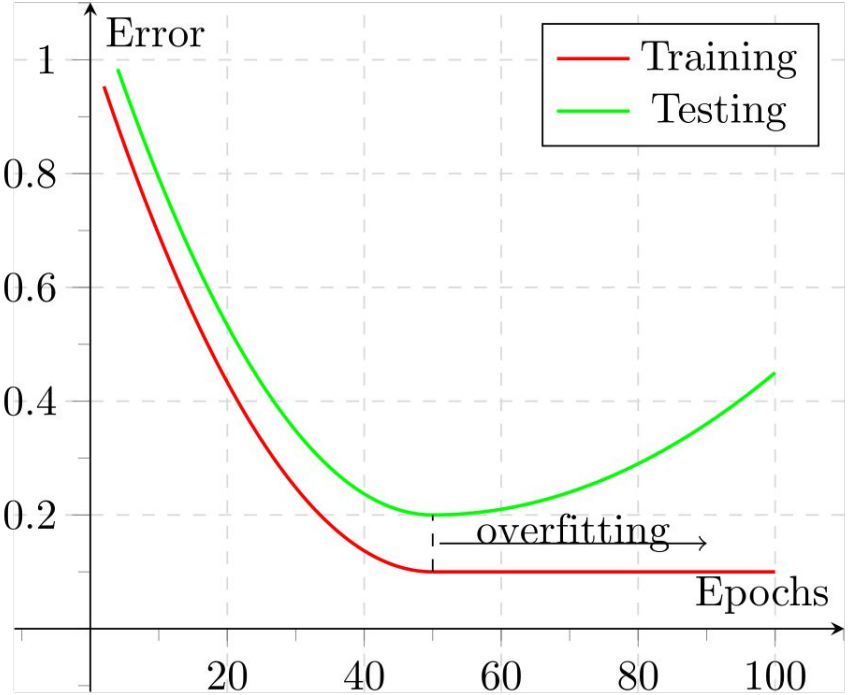
Generalization and overfitting

- Generalization is a Machine Learning model's ability to perform on unseen data
  - Using a separate test set to evaluate performance is always necessary
- If a machine learning model performs well on training data and poorly on testing data, it is said to **overfit**
- Overfitting is especially a problem if N_samples < N_variables
  - Being able to give gut estimates is a sign of a very competent data analyst
- http://scott.fortmann-roe.com/docs/BiasVariance.html



Model complexity

- Most gradient descent systems use iterations (or epochs in case of neural networks) to improve the system based on training data
- For an arbitarily complex model the classification error typically diminishes as iterations increase
- At some point the system starts **overfitting** to the training set
  - Model no longer **generalizes** as well
- Error for test set starts to gradually increase
- If the model is simple enough this will probably not happen before algorithm converges



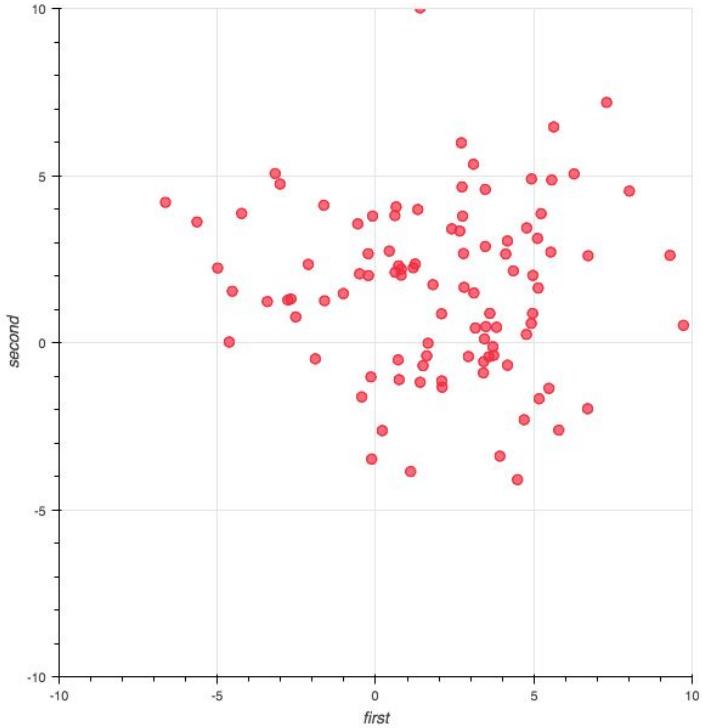
Approaches to test set-up

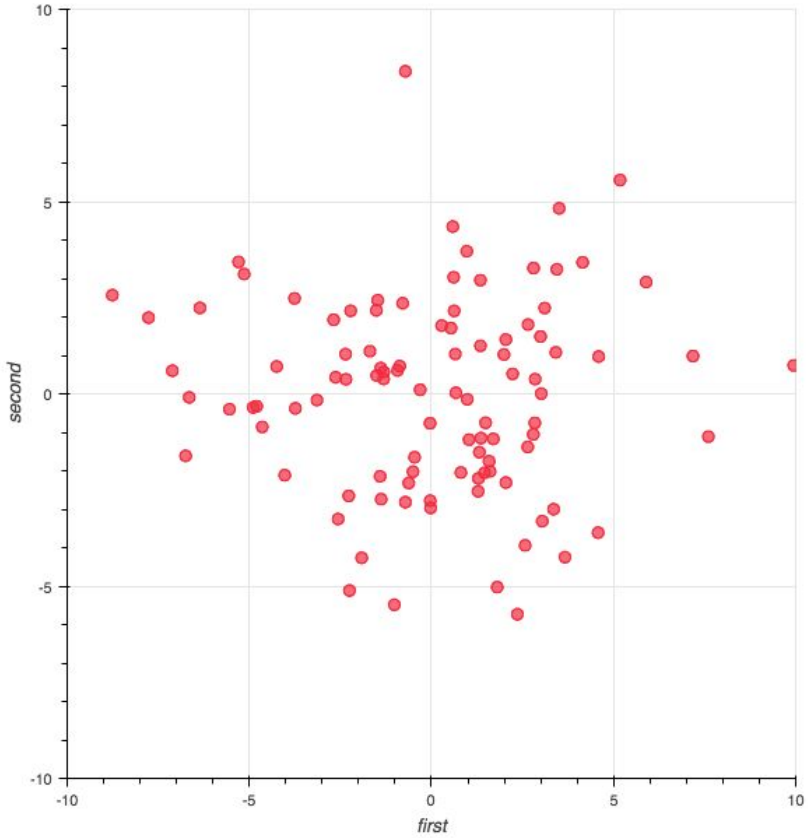
- Split data into train, test and **validate**
  - Train model based on train and use test to choose best model and optimize model parameters
  - Once you're done, validate that the test generalizes
- Cross-validation
  - Split data into k groups, use k-1 groups to train and the final to test
  - Let each group be the test group once
  - Report aggregate metric
  - May be expensive
  - 5-fold to 10-fold are good numbers
- Leave-One-Out (LOO)
  - Like the name says, leave only one datum out and classify it
  - Repeat N times where N is the number of samples
  - Often intractable for large N

Preprocessing

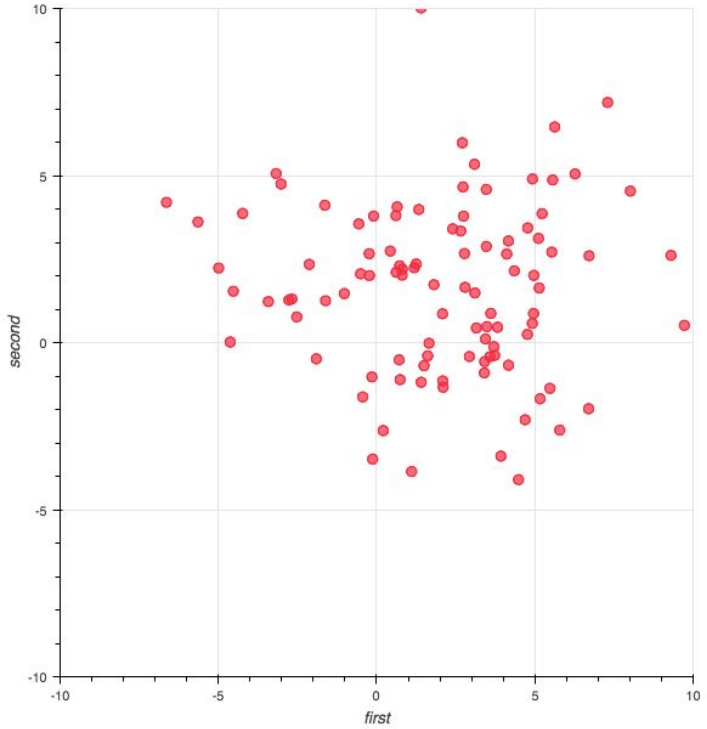
- As we’re drawing a hyperplane it’s usually handy if the axes are roughly proportional
- The least you can do with your data is to normalize each dimension to have
  - Zero mean
  - Unit variance

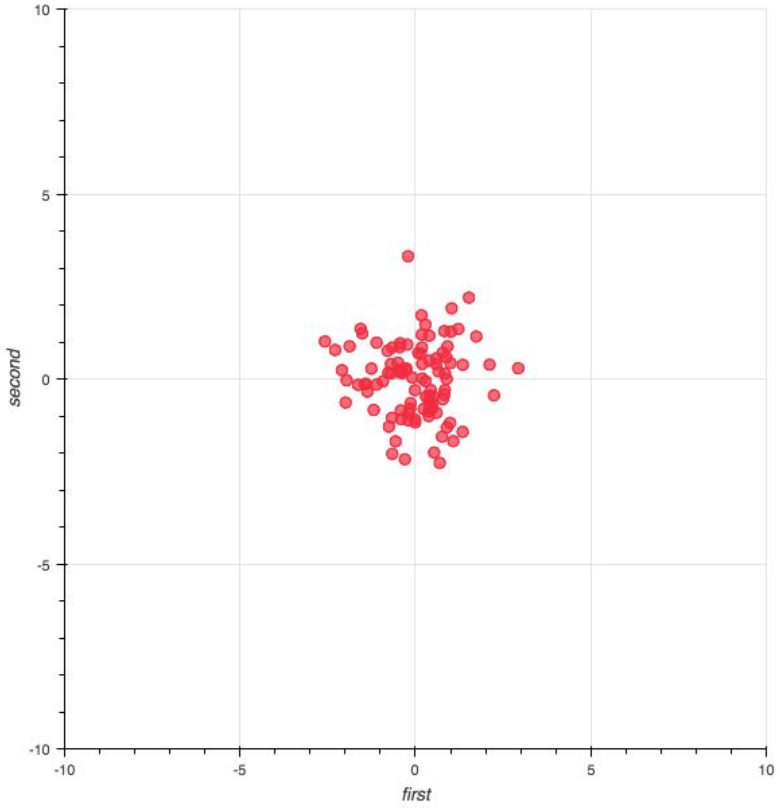
Zero-Mean Unit variance





Zero-Mean Unit variance

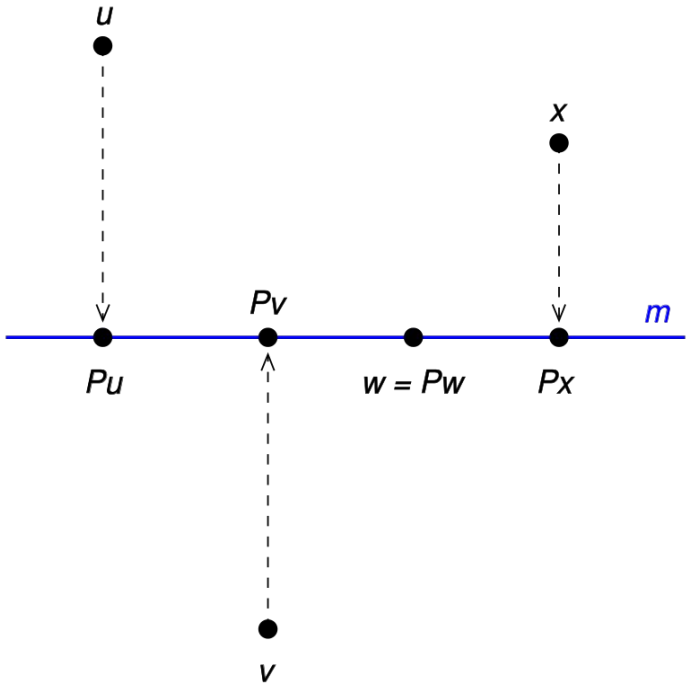




Dimensionality reduction

The curse of dimensionality

- High-dimensional spaces function in strange ways
- Some methods are intractable or just slow if the number of dimensions is high
- Human beings are only able to visualize ~3 dimensions
- Some spaces are sparse
  - Only <1% of values are nonzero
- N(dimensions) > N(samples) is generally a risky proposition
  - Many ways to go awry
  - Most dimensions are probably not relevant to the phenomenon



Examples of high dimensional data

- Genome data
  - N = the number of cells in a DNA Microarray
  - Not sparse
- Shopping data
  - N = the number of items available
  - Typically sparse
- Text data
  - N = the number of morphological units in the corpus
  - Typically sparse

Principal **C**omponent **A**nalysis

- Principal component analysis finds a projection, where
  - The first principal component (PC) maximizes variance along 1 dimension
  - The second PC is orthogonal and uncorrelated with the first and maximizes the variance in 2 dimensions with the first PC
  - Etc.
- PCA is very widely used
- It can be thought of as revealing the structure that maximizes the variance in the data
- Scaling the dimensions affects the results
  - You should always scale before doing PCA

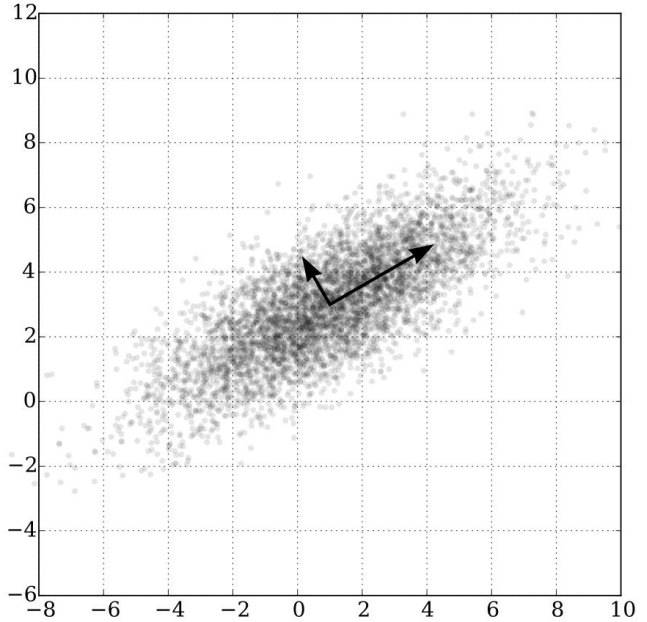


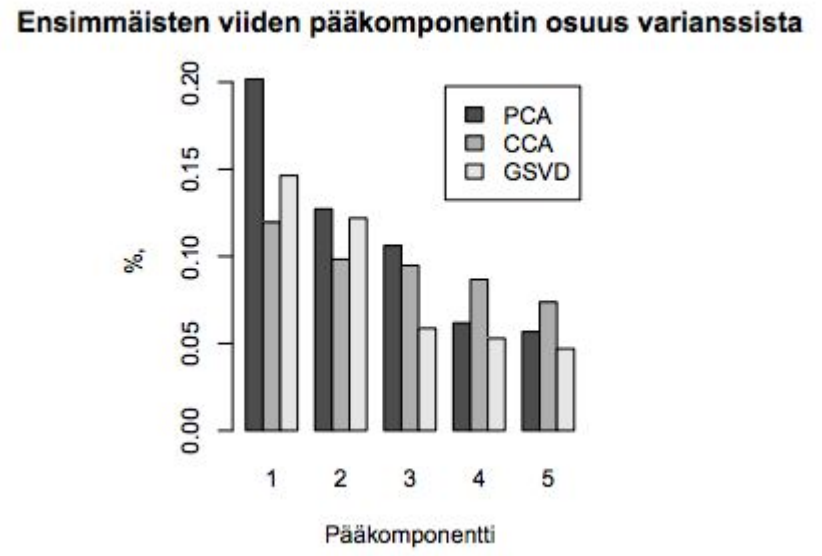
Image Wikimedia commons user Nicoguaro CC-BY-4

PCA uses

- Too many dimensions for your algorithm?
  - PCA it down to 10-20
- Too many dimensions to plot?
  - PCA it down to 2-3
- Data taking up too much space?
  - PCA it down until the principal components explain e.g. 99% of the variance
  - A form of lossy compression
- Data hairy and has lots of correlation etc.?
  - PCA it to make it more edible to algorithms and less understandable to humans

Proportion of variance

- The first principal components typically explain a large proportion of the variance
  - The proportion equals the proportion of the eigenvalues of the matrix
- This means we can usually get very similar results by projecting the data down to N < 10 Principal component axis as we would for the high-dimensional N >> 1000 data



Feature selection

- Sometimes it makes sense to just leave out some of the data
- Some data points may be entirely redundant
- Others may correlate so strongly that it doesn’t make sense to keep them
    - Some ML methods assume that all variables are independent and identically distributed
- Lasso is an algorithm for this
- PCA can also be used