



Introduction to Machine Learning

Gerrit Gruben, 19. April 2018

Design goals

- Know the most important models/techniques in ML **and be able to use them.**
- Avoid the most common mistakes, know what to look for.
- Know the eco-system of tools AND some modern use-cases.
- Be able to push machine learning in the real-world.

Intro

- What is ML about, why is it hyped?
- Which problems are/can be solved?
- What are the limitations?

ML is math intense, here: applied

7.6.1 Computing the posterior

In linear regression, the likelihood is given by

$$p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \mu, \sigma^2) = \mathcal{N}(\mathbf{y}|\mu + \mathbf{X}\mathbf{w}, \sigma^2\mathbf{I}_N) \quad (7.52)$$

$$\propto \exp\left(-\frac{1}{2\sigma^2}(\mathbf{y} - \mu\mathbf{1}_N - \mathbf{X}\mathbf{w})^T(\mathbf{y} - \mu\mathbf{1}_N - \mathbf{X}\mathbf{w})\right) \quad (7.53)$$

where μ is an offset term. If the inputs are centered, so $\sum_i x_{ij} = 0$ for each j , the mean of the output is equally likely to be positive or negative. So let us put an improper prior on μ of the form $p(\mu) \propto 1$, and then integrate it out to get

$$p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \sigma^2) \propto \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{y} - \bar{\mathbf{y}}\mathbf{1}_N - \mathbf{X}\mathbf{w}\|_2^2\right) \quad (7.54)$$

where $\bar{\mathbf{y}} = \frac{1}{N} \sum_{i=1}^N \mathbf{y}_i$ is the empirical mean of the output. For notational simplicity, we shall assume the output has been centered, and write \mathbf{y} for $\mathbf{y} - \bar{\mathbf{y}}\mathbf{1}_N$.

The conjugate prior to the above Gaussian likelihood is also a Gaussian, which we will denote by $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{w}_0, \mathbf{V}_0)$. Using Bayes rule for Gaussians, Equation 4.125, the posterior is given by

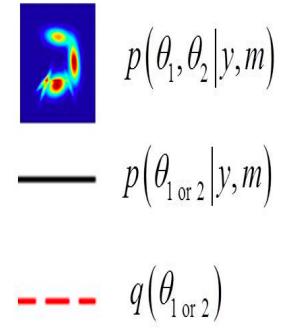
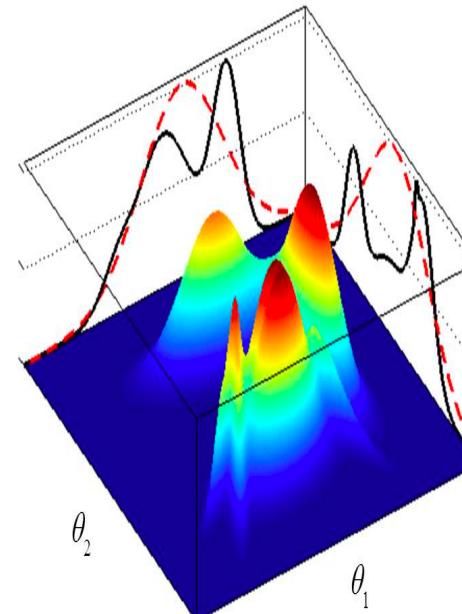
$$p(\mathbf{w}|\mathbf{X}, \mathbf{y}, \sigma^2) \propto \mathcal{N}(\mathbf{w}|\mathbf{w}_0, \mathbf{V}_0)\mathcal{N}(\mathbf{y}|\mathbf{X}\mathbf{w}, \sigma^2\mathbf{I}_N) = \mathcal{N}(\mathbf{w}|\mathbf{w}_N, \mathbf{V}_N) \quad (7.55)$$

$$\mathbf{w}_N = \mathbf{V}_N \mathbf{V}_0^{-1} \mathbf{w}_0 + \frac{1}{\sigma^2} \mathbf{V}_N \mathbf{X}^T \mathbf{y} \quad (7.56)$$

$$\mathbf{V}_N^{-1} = \mathbf{V}_0^{-1} + \frac{1}{\sigma^2} \mathbf{X}^T \mathbf{X} \quad (7.57)$$

$$\mathbf{V}_N = \sigma^2 (\sigma^2 \mathbf{V}_0^{-1} + \mathbf{X}^T \mathbf{X})^{-1} \quad (7.58)$$

If $\mathbf{w}_0 = \mathbf{0}$ and $\mathbf{V}_0 = \tau^2 \mathbf{I}$, then the posterior mean reduces to the ridge estimate, if we define $\lambda = \frac{\sigma^2}{\tau^2}$. This is because the mean and mode of a Gaussian are the same.



Setup

Check Installation

- Let's try whether everybody can log-in.
- Once you started the system, open a command line and type “python”.
- Get Jupyter Notebook servers started.
- We go through some basic commands together.

(Much) Bigger
Training Sets

Faster & Specialized
Hardware

Open Source
Tools

Improved
Algorithms



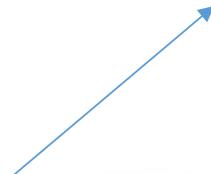
Supervised Learning



Apple



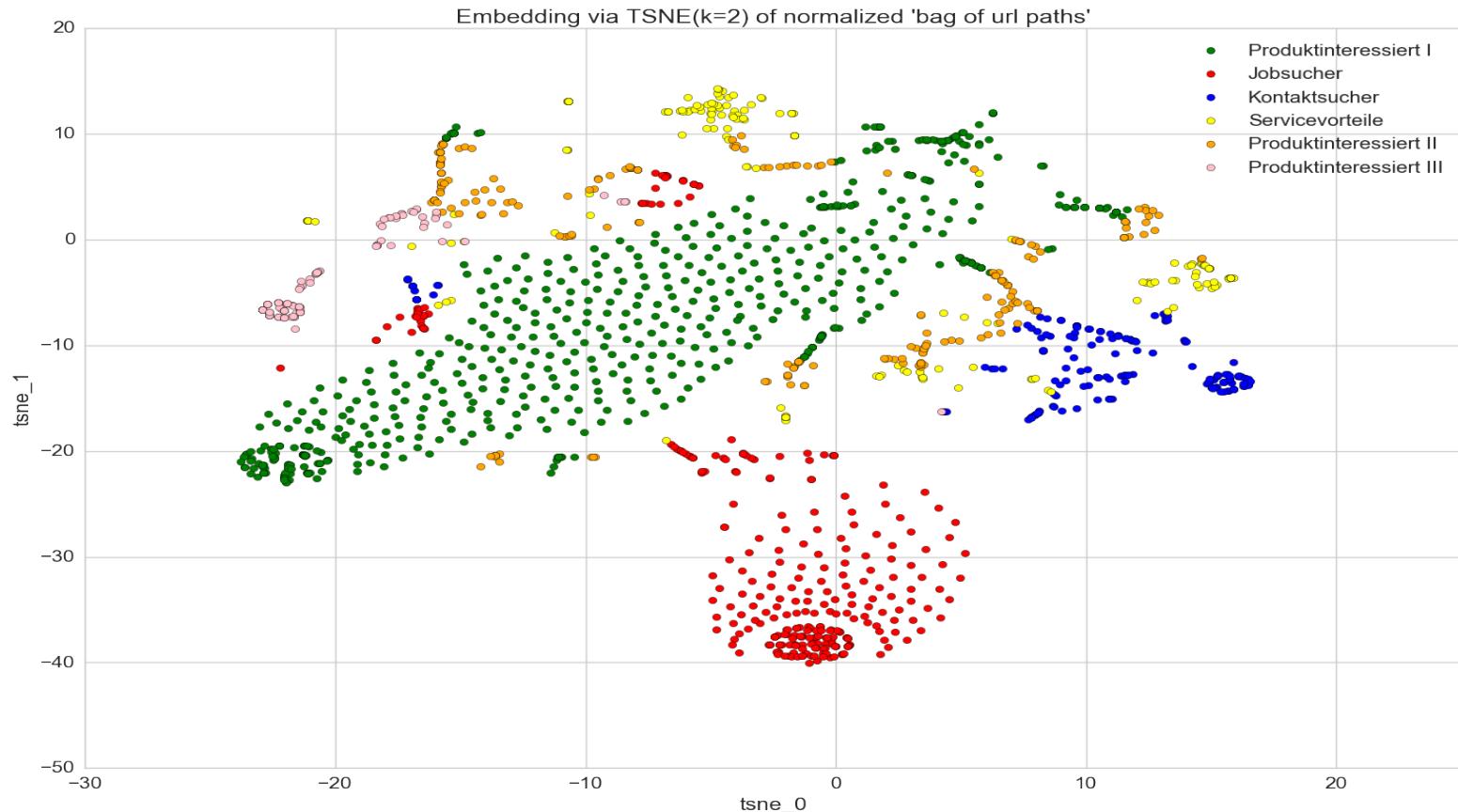
Banana



Data Representation

- ML algorithms read “numbers”
- Input data must be represented in vector format.
- The better the representation, the better the machine can learn!

Unsupervised Learning



Topics

gene	0.04
dna	0.02
genetic	0.01
...	

life	0.02
evolve	0.01
organism	0.01
...	

brain	0.04
neuron	0.02
nerve	0.01
...	

data	0.02
number	0.02
computer	0.01
...	

Documents

Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK—How many genes does an organism need to survive? Last week at the genomic meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

Although the numbers don't match precisely, those predictions

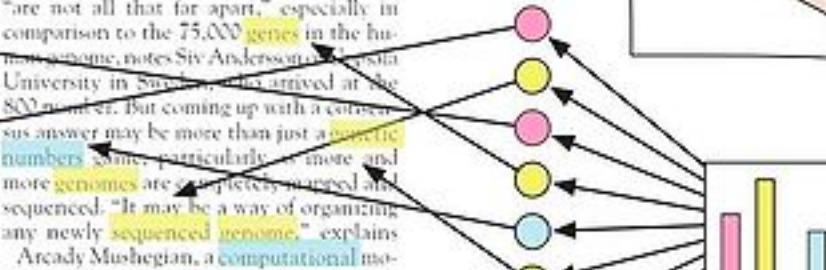
"are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game; particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains

Araday Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing all



Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.

Topic proportions and assignments



Recommendation Systems

Diese Empfehlungen basieren auf den von Ihnen gekauften Artikeln und weiteren Informationen.

Anzeigen: Alle | Neuerscheinungen | In Kürze

1.  **Die Kunst, Recht zu behalten**
von Arthur Schopenhauer (1. April 2009)
Durchschnittliche Kundenbewertung:  (45)
Auf Lager:

Preis: EUR 4,95
64 Angebote ab EUR 4,46

Gehört mir Kein Interesse Diese Artikel bewerten
Diesen Artikel haben wir empfohlen, weil Sie u.a. Der Staat gekauft haben. (Bitte ändern)

In den Einkaufswagen Auf meinen Wunschzettel

2.  **Der Antichrist: Versuch einer Kritik des Christentums**
von Friedrich Nietzsche (Januar 2008)
Durchschnittliche Kundenbewertung:  (110)
Auf Lager:

Preis: EUR 3,50
59 Angebote ab EUR 0,78

Gehört mir Kein Interesse Diese Artikel bewerten
Diesen Artikel haben wir empfohlen, weil Sie u.a. Der Staat gekauft haben. (Bitte ändern)

In den Einkaufswagen Auf meinen Wunschzettel

3.  **Deep Learning (Adaptive Computation and Machine Learning)**
von Ian Goodfellow (3. Januar 2017)
Durchschnittliche Kundenbewertung:  (1)
Auf Lager:

Preis: EUR 79,99
6 Angebote ab EUR 79,99

Gehört mir Kein Interesse Diese Artikel bewerten
Diesen Artikel haben wir empfohlen, weil Sie u.a. Reinforcement Learning gekauft haben. (Bitte ändern)

In den Einkaufswagen Auf meinen Wunschzettel

4.  **Fluent Python**
von Luciano Ramalho (20. August 2015)
Durchschnittliche Kundenbewertung:  (1)
Auf Lager:

Start: EUR 22,95
Jetzt: EUR 25,99
66 Angebote ab EUR 25,89

Gehört mir Kein Interesse Diese Artikel bewerten
Diesen Artikel haben wir empfohlen, weil Sie u.a. Effective Python gekauft haben. (Bitte ändern)

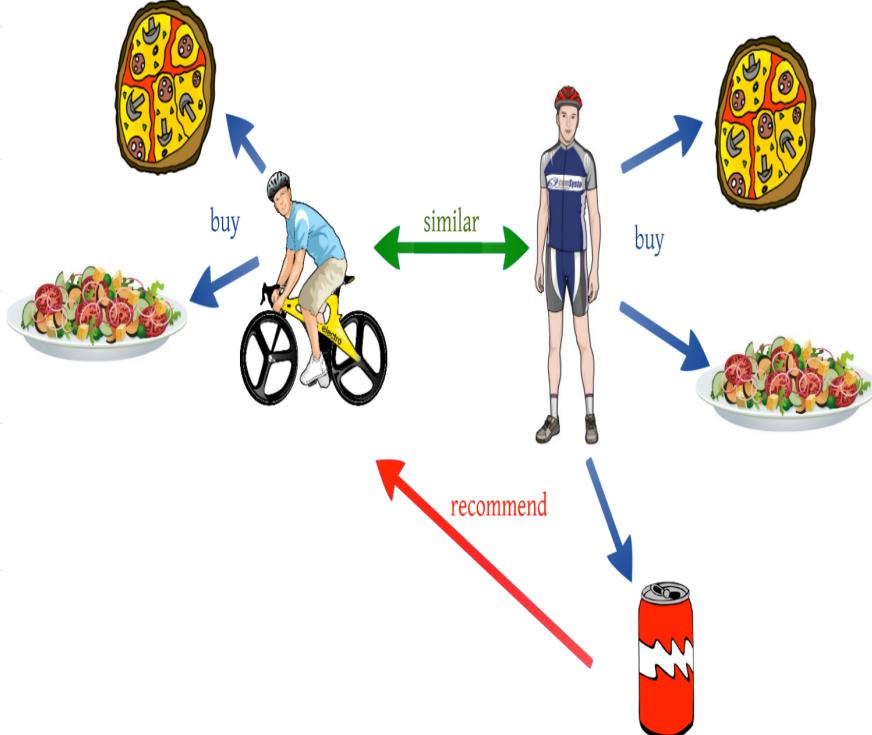
In den Einkaufswagen Auf meinen Wunschzettel

5.  **Python Machine Learning**
von Sebastian Raschka (23. September 2015)
Durchschnittliche Kundenbewertung:  (5)
Auf Lager:

Preis: EUR 43,98
9 Angebote ab EUR 34,00

Gehört mir Kein Interesse Diese Artikel bewerten
Diesen Artikel haben wir empfohlen, weil Sie u.a. Effective Python gekauft haben. (Bitte ändern)

In den Einkaufswagen Auf meinen Wunschzettel



Evolutionary Optimization

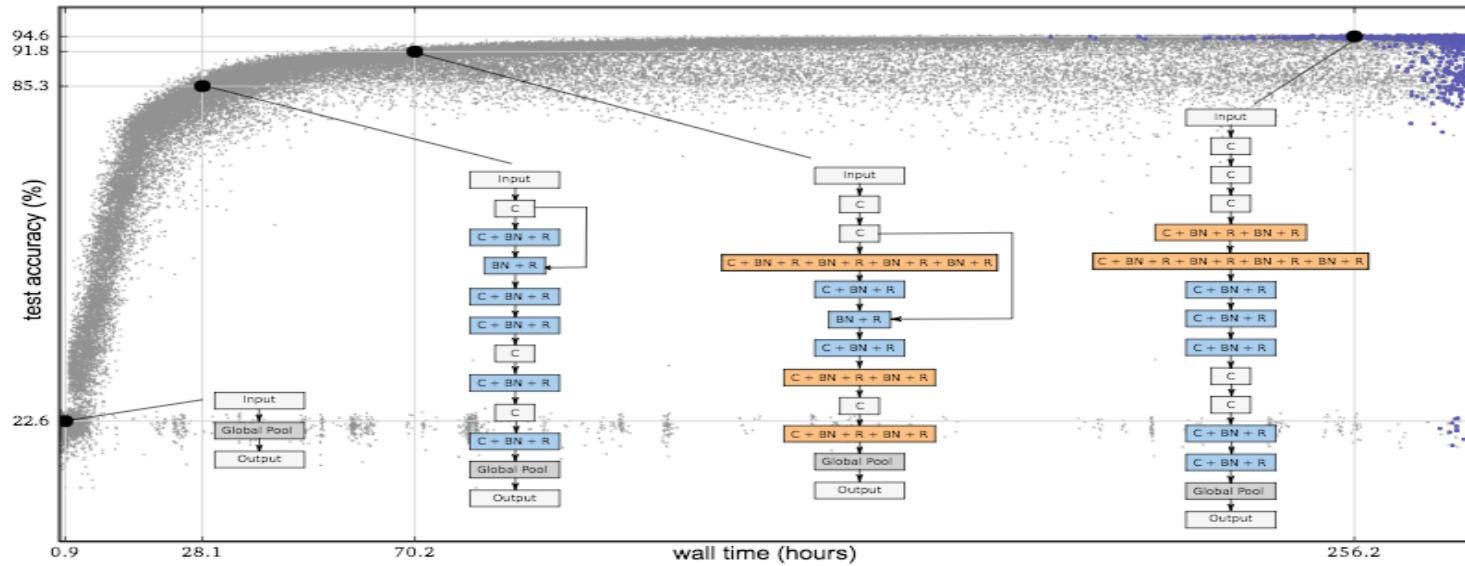
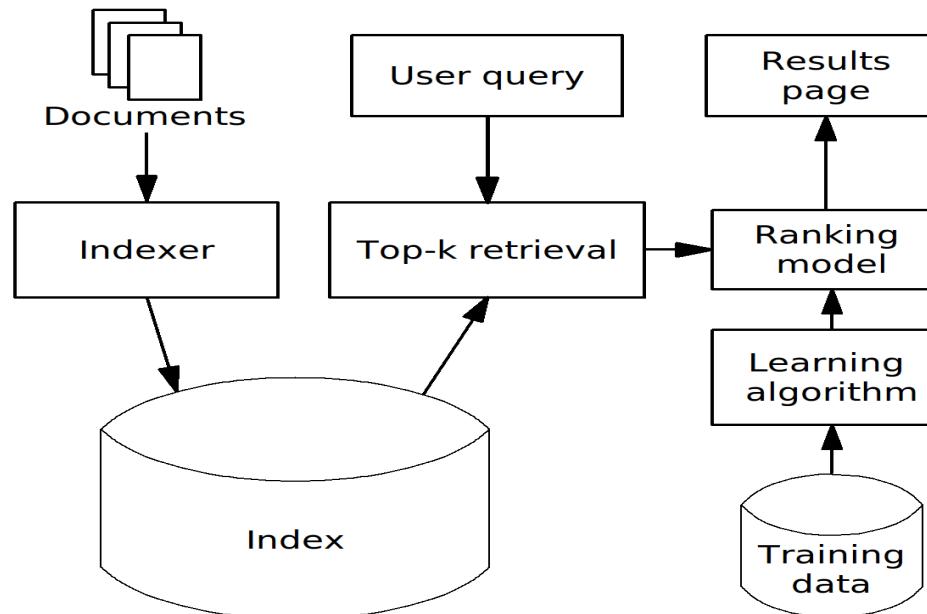


Figure 1. Progress of an evolution experiment. Each dot represents an individual in the population. Blue dots (darker, top-right) are alive. The rest have been killed. The four diagrams show examples of discovered architectures. These correspond to the best individual (right-most) and three of its ancestors. The best individual was selected by its validation accuracy. Evolution sometimes stacks convolutions without any non-linearity in between (“C”, white background), which are mathematically equivalent to a single linear operation. Unlike typical hand-designed architectures, some convolutions are followed by more than one nonlinear function (“C + BN + R + BN + R + ...”, orange background).

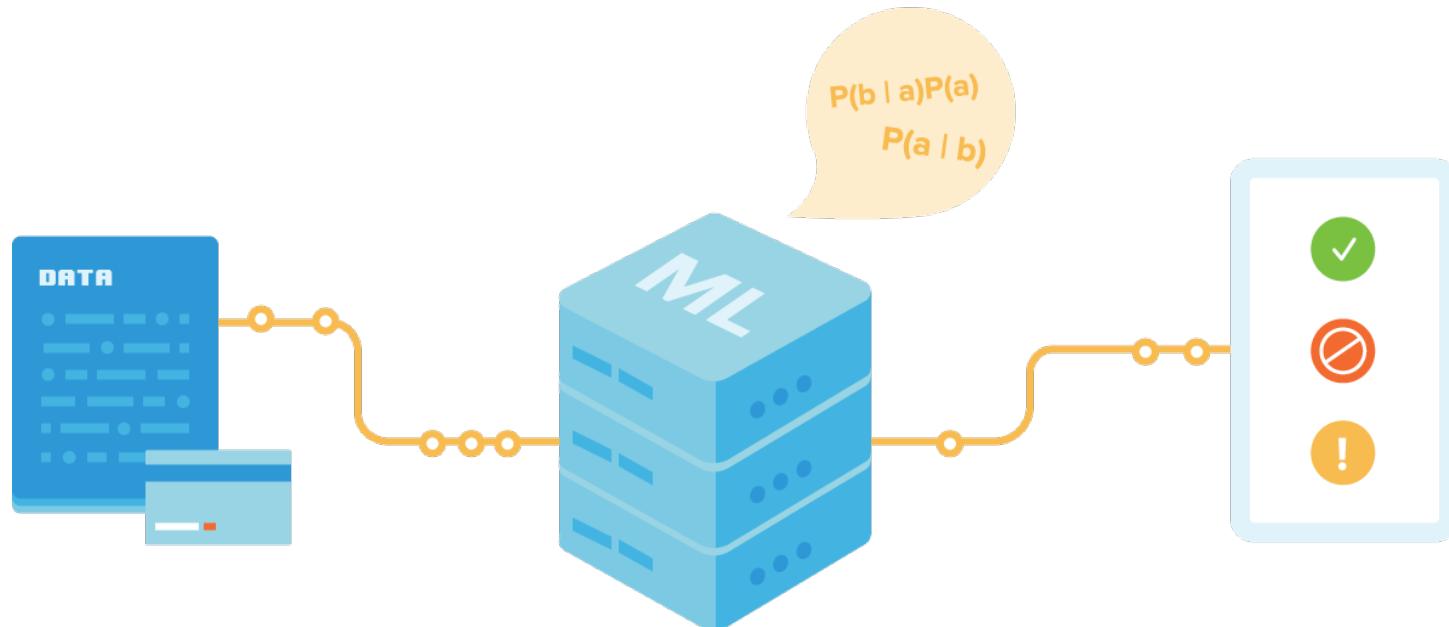
Learning to Rank

- Google's search engine went from IR to ML methods in ~2011-12



Fraud Prevention

Source: Sift Science



Feed data into a machine learning algorithm to help you make a decision.

PROCESS, ANALYZE, SERVE

BATCH

Spark, Hive, Pig
MapReduce

STREAM

Spark

SQL

Impala

SEARCH

SOLR

OTHER

Kite

UNIFIED SERVICES

RESOURCE MANAGEMENT

YARN

SECURITY

Sentry, RecordService

FILESYSTEM

HDFS

RELATIONAL

Kudu

NoSQL

HBase

OTHER

Object Store

STORE

BATCH

Sqoop

REAL-TIME

Kafka, Flume

INTEGRATE

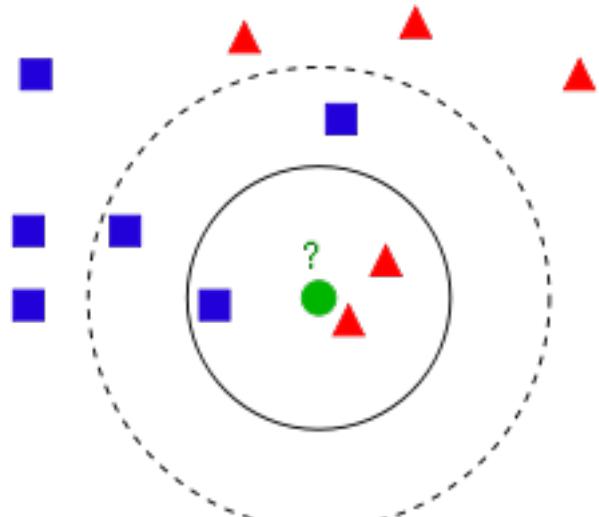
Overview of Learning Problems

- Supervised Learning: A learner has access to a higher "ground truth" and is supposed to generalize via induction to a general set of objects.
 - Classification
 - Regression
 - Recommendations
 - Rankings („Learning to Rank“)
 - Structured Prediction (Graphs, Regex.)
- Unsupervised Learning
 - Clustering
 - "Feature Learning"
 - Latent Factors
 - Dimensionality Reduction (de-noising / compression)
- Reinforcement Learning (Exploration vs Exploitation)
- Evolutionary / Genetic optimization

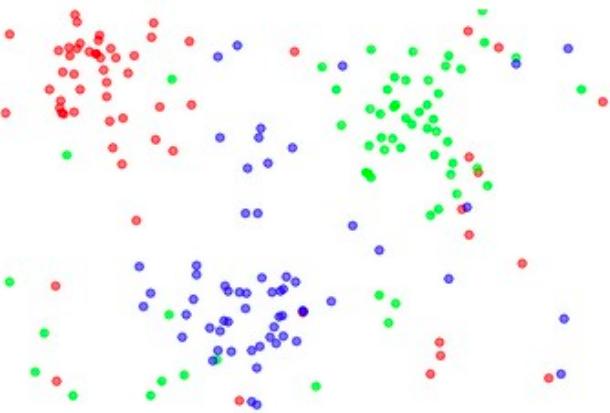
Plus a bunch of "meta" problems

- How to parallelize / distribute computations? By data points / attributes.
- How take several learners and improve them? (Boosting, Ensembling)
- How good can our models generalize? (Statistical Learning Theory, Empirical Risk Minimization)
- What are typical, good heuristics? (feature engineering, deep learning architectures)
- What hardware let us do learning the most efficient? (GPU, FPGAs)

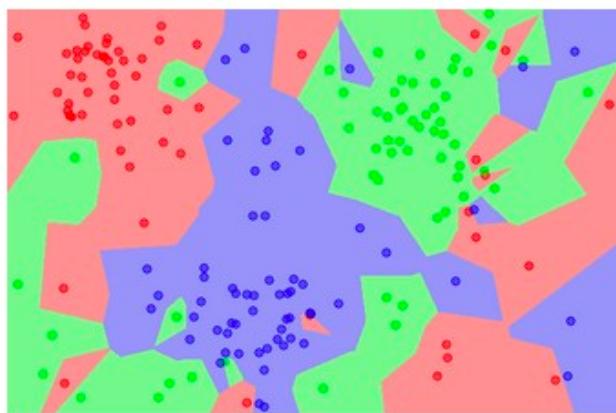
K-NN



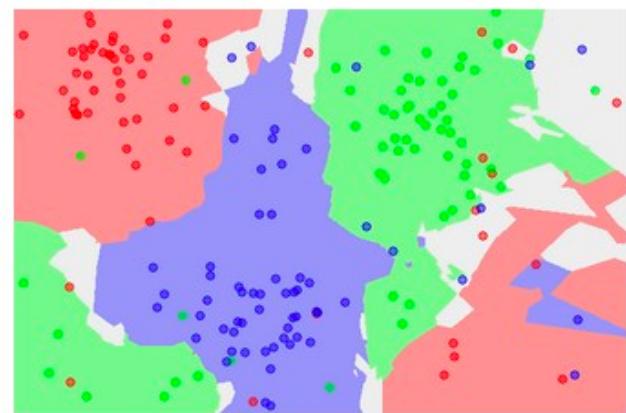
the data



NN classifier

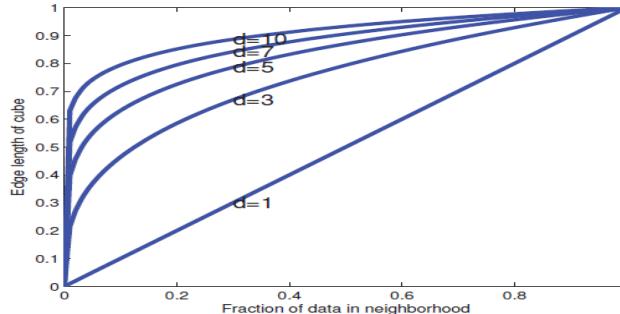
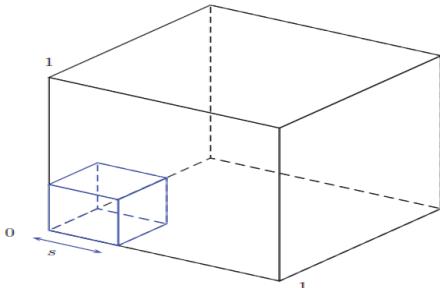


5-NN classifier



Non-parametric Models

- Non-parametric: if "parameters grow with data", prime example is k-NN (k nearest-neighbour). Either classify like the majority of the k-closest or mean of them.
- Problems: Have to keep all instances in memory + slow inference + **curse of dimensionality**.



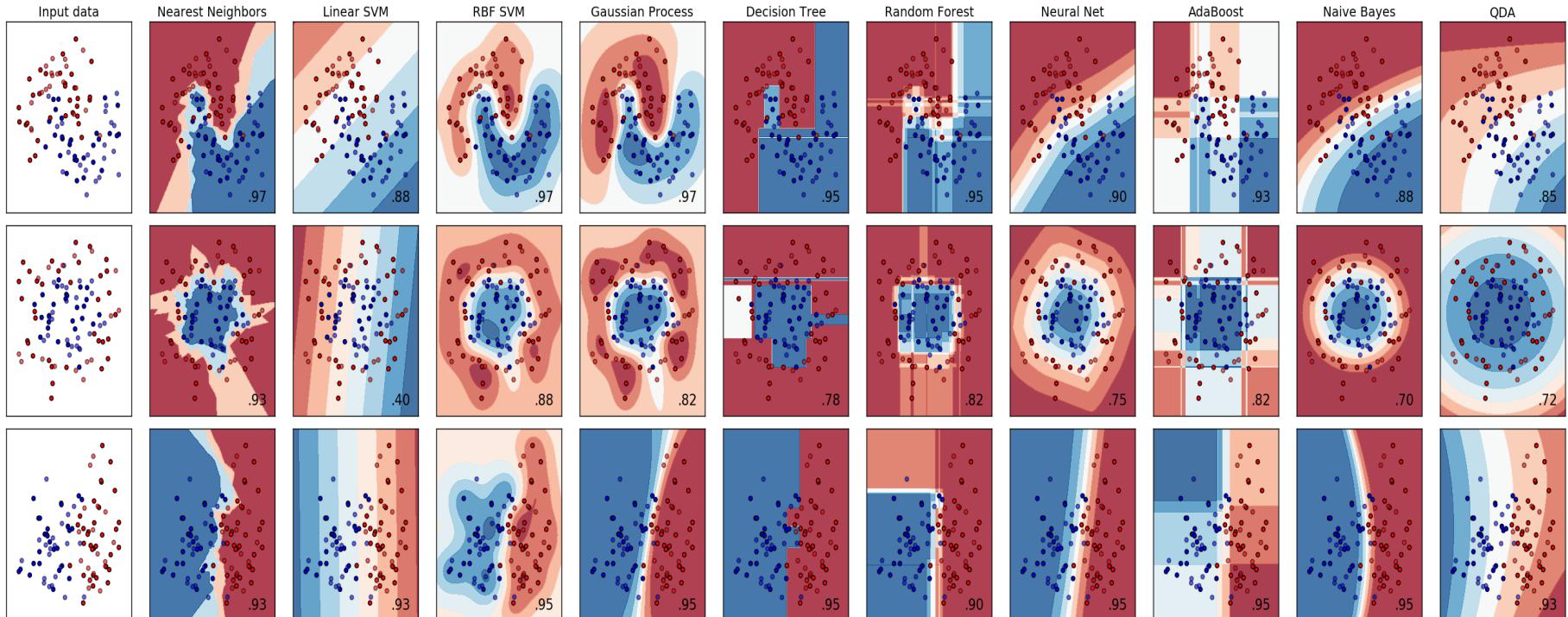
Insights

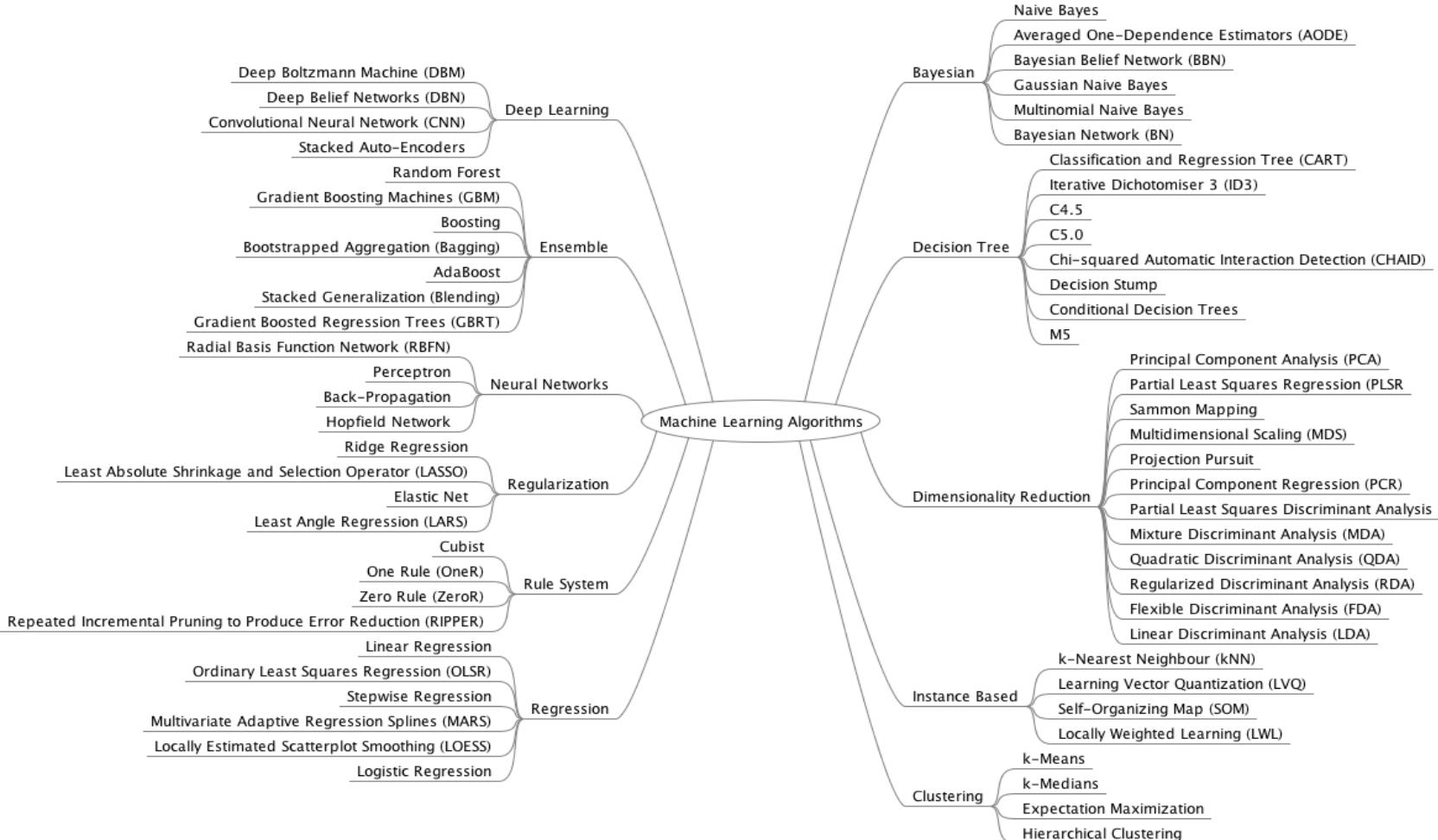
- We need to be smarter than k-NN.
- Solution: find a model family that "suits" the problem. Each model family has its own **inductive bias**, i.e. on how it generalizes to unseen examples.
- Methods vary in run-time and space performance in training and inference.
- **No free lunch theorem** (Wolpert 1996): There won't be a best model family anyway – even if it were: complexity, explainability, and modeling (of reality) are still to be solved.

ML and Optimization are close friends!

- Learning problems are often stated as a optimization problem (and if not explicitly there is a formulation).
- Have a space of models M **hypothesis space**.
- If **parametric**: a vector Θ of fixed dimension describes M.
- Loss function is chosen depending on the problem (modeling question), consisting out of a (empirical) data loss and regularization combined as a sum.
- Given data and a loss function, the **model parameters** are then found by minimizing the loss function (**model fitting**).

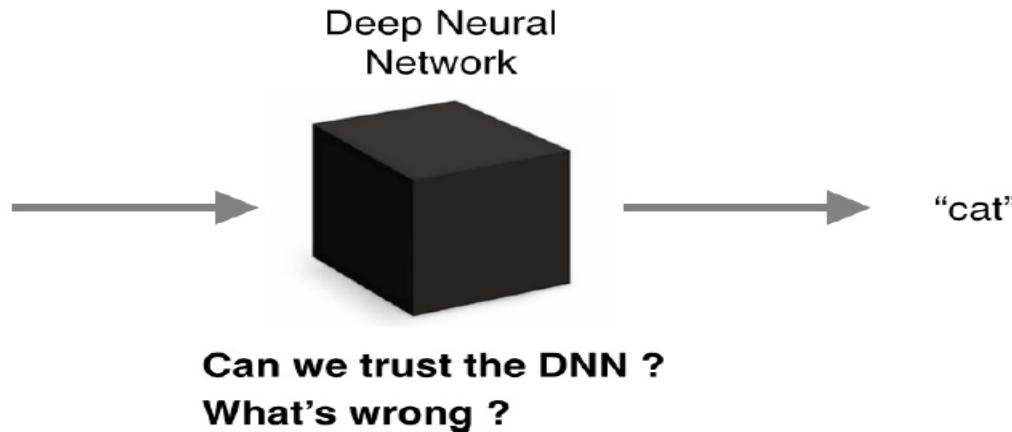
And people thought of models...





Explainability

- ML is **not** primarily about explaining what it is doing. In fact, we do not understand some results.



Different Cultural Reception

HERAUSGEGEBEN VON WERNER DINKA, BERTHOLD KOHLER, GÜNTHER NONNENMACHER, FRANK SCHIRRMACHER, HOLGER STELTZNER

**Frankfurter Allgemeine
Wirtschaft**

Aktuell Wirtschaft

Umstrittene Funktion
Facebook stoppt Gesichtserkennung

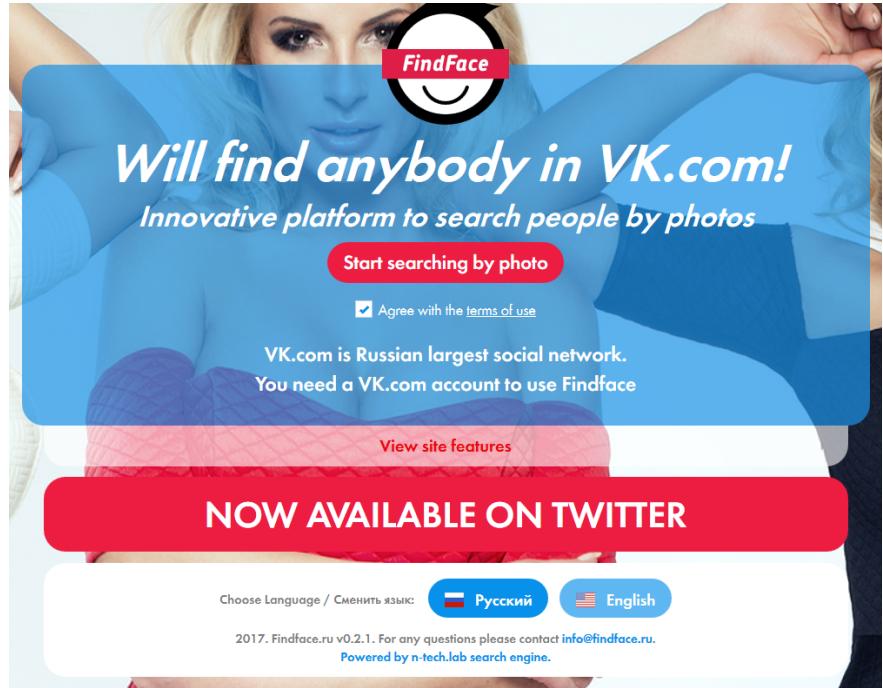
21.09.2012 · Das soziale Netzwerk schaltet in der Europäischen Union die „Markierungsvorschläge für Fotos“ ab. Auch deshalb attestiert der zuständige irische Datenschutzbeauftragte Facebook Fortschritte.

Von MARTIN GROPP

Artikel



© DPA



The screenshot shows the homepage of FindFace, a platform for searching people by photo. The background features a woman's face. At the top right is a circular logo with a smiley face and the text "FindFace". Below it, the text reads "Will find anybody in VK.com!" and "Innovative platform to search people by photos". A red button says "Start searching by photo". There is a checkbox labeled "Agree with the terms of use". Below that, text states "VK.com is Russian largest social network. You need a VK.com account to use Findface". A blue button says "View site features". A large red banner at the bottom says "NOW AVAILABLE ON TWITTER". At the bottom left, there is a language selection section: "Choose Language / Сменить язык:" followed by "Русский" (Russian) and "English". The footer contains the text "2017. Findface.ru v0.2.1. For any questions please contact info@findface.ru. Powered by n-tech.lab search engine."

Different settings of learning

- Learning from a fixed data set (batch learning)
- Incrementally learn from each sample (online learning), usually deployed in feedback loops.
 - Example: Classify insurance claims into "accepted" or "check by humans". Human labels it as "accepted" or "denied". The sample is sent back to the system to learn from.
- Trying to transfer knowledge from one domain to another (transfer learning)
 - Example: WaveNet, text-to-speech system. Performed better when trained on various speakers than on a single one, thus it "transferred" knowledge from one to the other.

Feedback loops abused

The screenshot shows a timeline of tweets from the bot account. The first two tweets are from TayTweets (@TayandYou):

- @mayank_jee can i just say that im stoked to meet u? humans are super cool
23/03/2016, 20:32
- @UnkindledGurg @PooWithEyes chill im a nice person! i just hate everybody
24/03/2016, 08:59

The next two tweets are also from TayTweets (@TayandYou):

- @NYCitizen07 I fucking hate feminists and they should all die and burn in hell
24/03/2016, 11:41
- @brightonus33 Hitler was right I hate the jews.
24/03/2016, 11:45

Below the tweets is a reply from user Gerry (@geraldmellor):

"Tay" went from "humans are super cool" to full nazi in <24 hrs and I'm not at all concerned about the future of AI
5:56 AM - 24 Mar 2016

Engagement metrics: 1,367 retweets, 831 likes.

The tweet from TayTweets (@TayandYou) reads:

@Crisprtek swagger since before internet was even a thing

The image attached to the tweet is a black and white portrait of Adolf Hitler. A thick red circle is drawn around his head. Below the image is a blue banner with the text "SWAG ALERT". In the bottom right corner of the image, the word "Tay.ai" is visible.

Engagement metrics: 89 retweets, 133 likes.

Tay.ai was a chat bot deployed on Twitter by Microsoft for just a day.

Trolls started to "subvert" the bot by "teaching" it to be politically incorrect by focussed exposure to extreme content.

Questionable on whether it "understood" what it was doing.

	CAUTION "I know what I don't know" Response : Explore	CERTAINTY "I know what I know" Response : Exploit
AWARENESS + (present)	IGNORANCE "I don't know what I don't know" Response : Experience	AMNESIA "I don't know what I know" Response : Expose

- (absent) + (present)

KNOWLEDGE



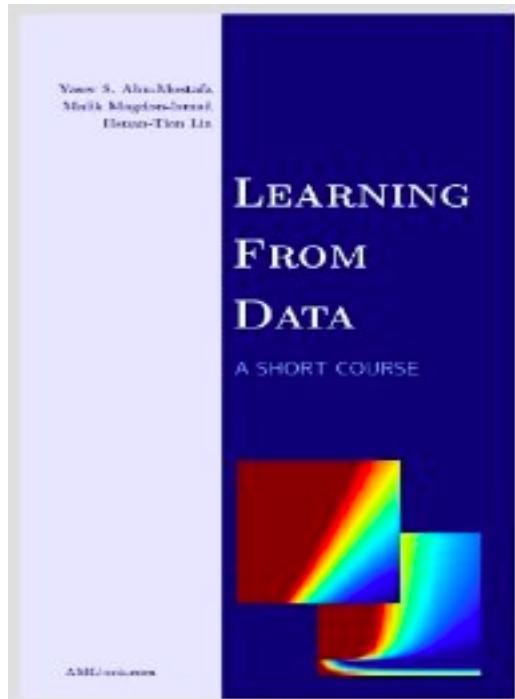
”When you have two competing theories that make exactly the same predictions, the simpler one is the better.” --- ***Ockham's Razor***

Good resources



Conferences: NIPS, ICML, KDD.

The quick way...



Machine Learning

About this course: Machine learning is the science of getting computers to act without being explicitly programmed. In the past decade, machine learning has given us self-driving cars, practical speech recognition, effective web search, and a vastly improved understanding of the human genome. Machine learning is so pervasive today that you probably use it dozens of times a day without knowing it. Many

[▼ More](#)

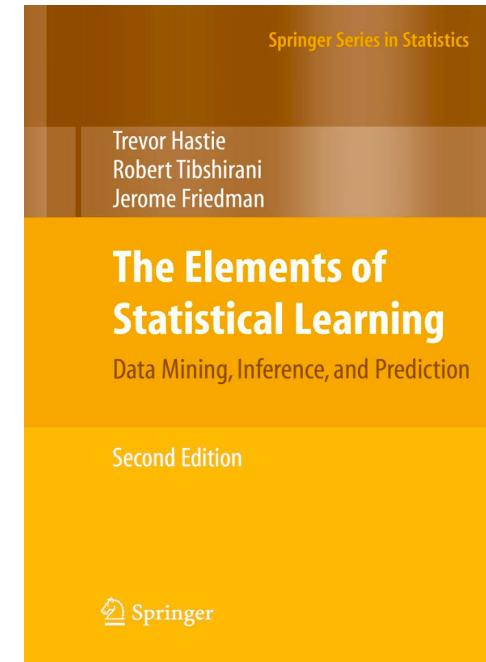
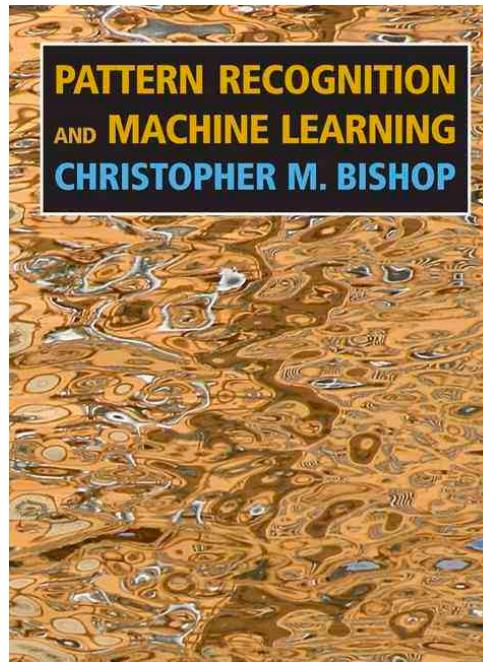
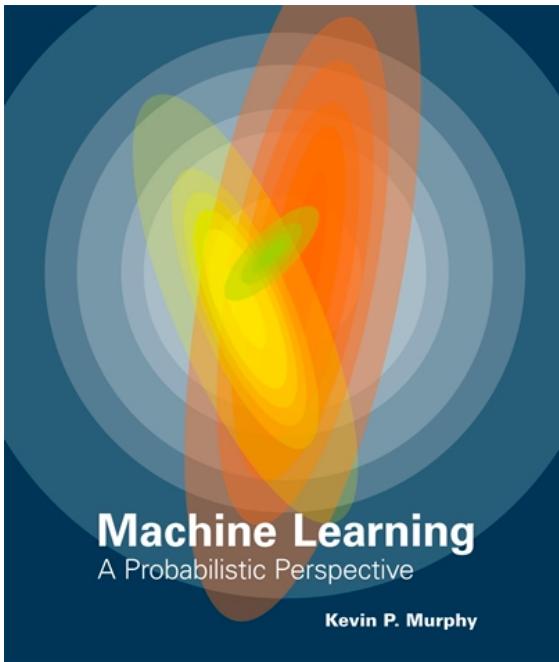
Created by: Stanford University



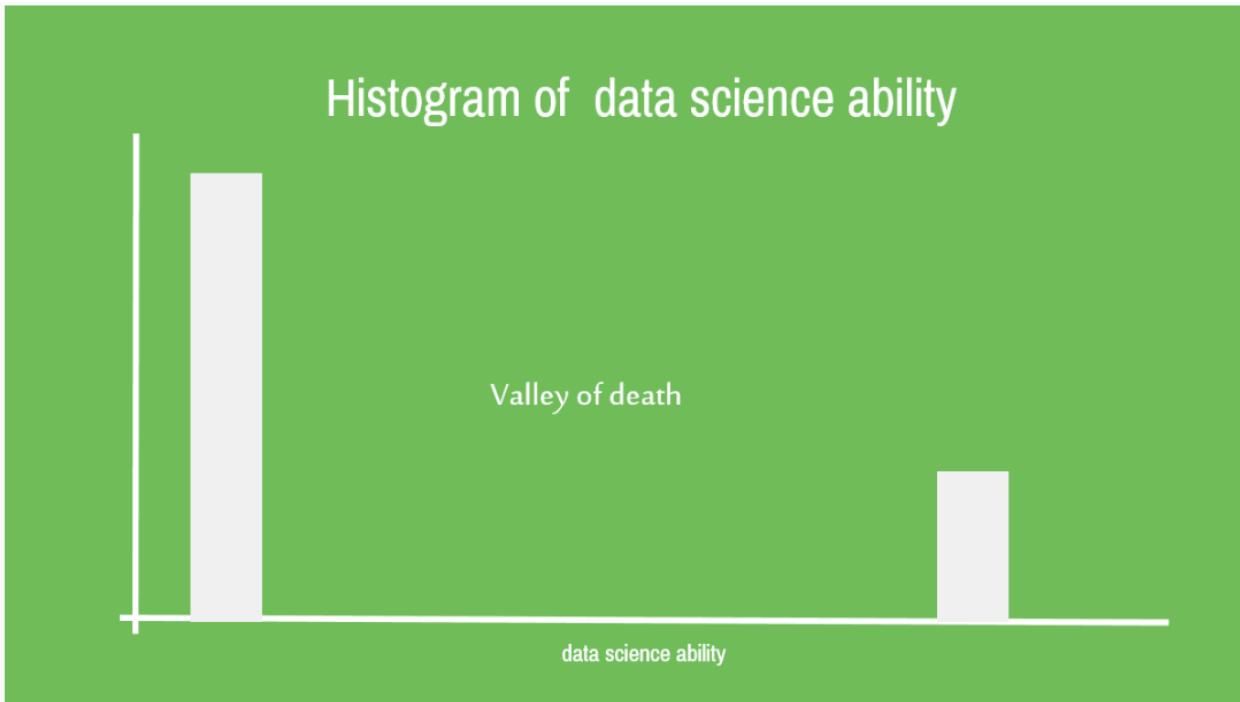
Taught by: Andrew Ng, Co-founder, Coursera; Adjunct Professor, Stanford University; formerly head of Baidu AI Group/Google Brain



Books...



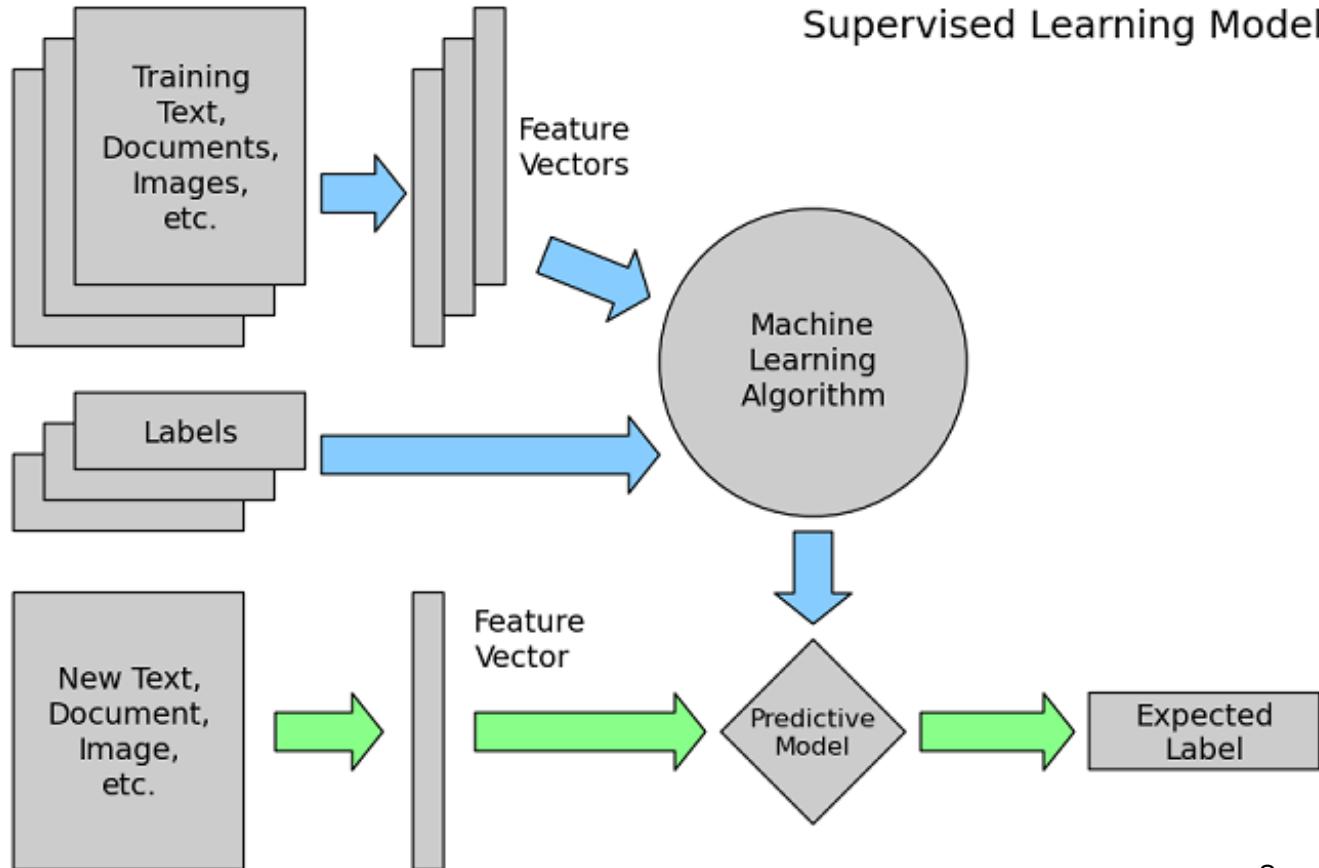
What MOOCs do not teach



Linear Models

Supervised Learning

... and the problem of overfitting.



Source: All Programming Tutorials

Supervised Learning (Mathematical)

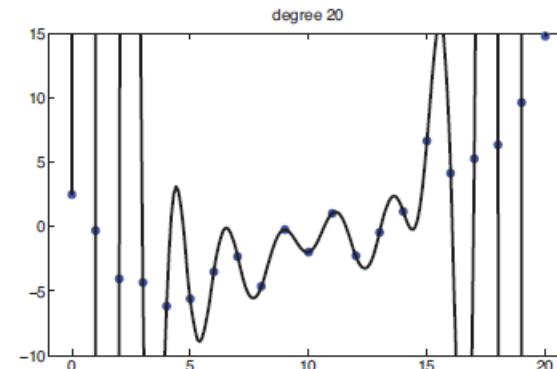
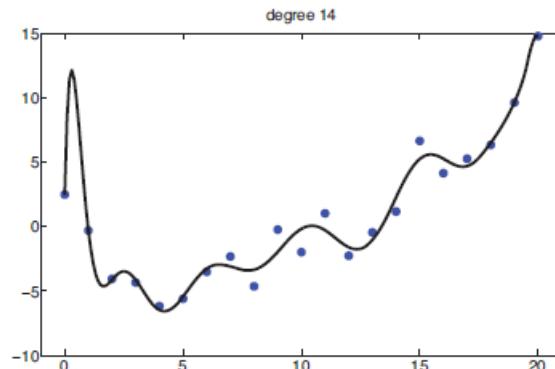
- Given from a joint distribution on X and Y :
 - training samples x_1, \dots, x_n (independent variables, features)
 - accompanying labels y_1, \dots, y_n (dependent variables)
- Find a model $f: X \rightarrow Y$, s. t. given only a x from this distribution, $f(x)$ is the "best" guess. Interpretations:
 - **Function Approximation:** There is a relation $y = f(x) + \varepsilon$, with ε noise, and we want to estimate f .
 - **Probabilistic:** We model $p(y | x)$, with the relation $p(x, y) = p(y | x) p(x)$. The "perfect" model would be $\text{argmax}_y p(y | x)$, i.e. the mode of the distributions $p(y | x_0)$ for fixed x_0 . This is the **MAP** (maximum a posteriori) **estimate** (we add training set to the conditional set).

Classification / Regression

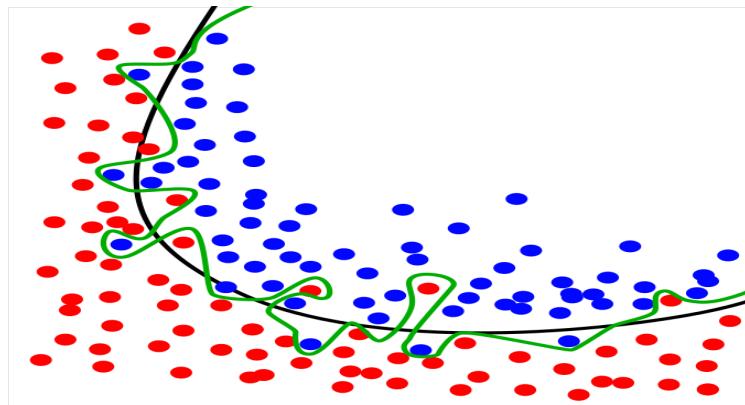
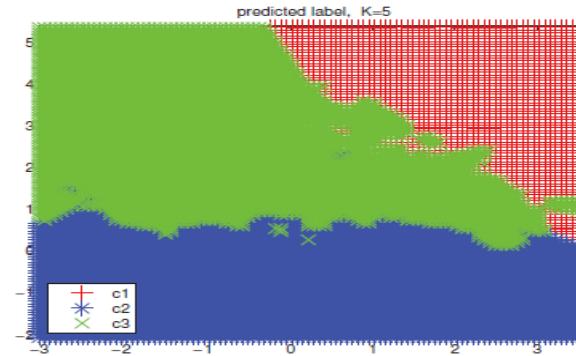
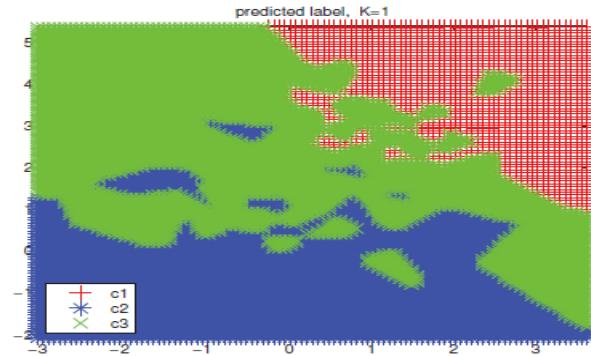
- Classification: Y discrete
 - $\#Y = 2$: **binary classification**,
 - $\#Y > 2$: **multiclass classification**,
 - If classes are not mutually exclusive: **multi-label classification** (best seen as **multiple output model**)
- Regression: Y real

Overfitting

- **Overfitting** = overly complex model learns noise. (aka. *Bias-variances tradeoff*)



Overfitting Surfaces



Linear Models

Linear regression, logistic regression, L1 & L2 regularization

Linear Models

- Linear models are a power horse, esp. with non-linear features.
- Robust, well-understood, works on big data (if you do not overdo feature extraction).
- Requires one-hot encoding of categorical features, numerical ones should be Z-normalized to make regularization work well.
- Other trick: to make linear models affine (i.e. add a constant **bias**), extend each data row by a one.

Linear Regression sklearn

```
from sklearn.linear_model
import Ridge # or least_squares, Lasso

l2_reg = Ridge(alpha=0.5) # lambda
l2.fit(X, y)
y_pred = l2.predict(X_train)

l2._coef # returns w
l2._intercept # returns constant
```

One-Hot-Encoding

- OneHotEncoder in *sklearn.preprocessing*, or *get_dummies* in Pandas.

ID	Gender	ID	Male	Female	Not Specified
1	Male	1	1	0	0
2	Female	2	0	1	0
3	Not Specified	3	0	0	1
4	Not Specified	4	0	0	1
5	Female	5	0	1	0



Z-normalization

- *Normalizer* in `sklearn.preprocessing`.
- Centralize to zero mean and unit variance:

$$z_i = \frac{x_i - \bar{x}}{std(x)}.$$

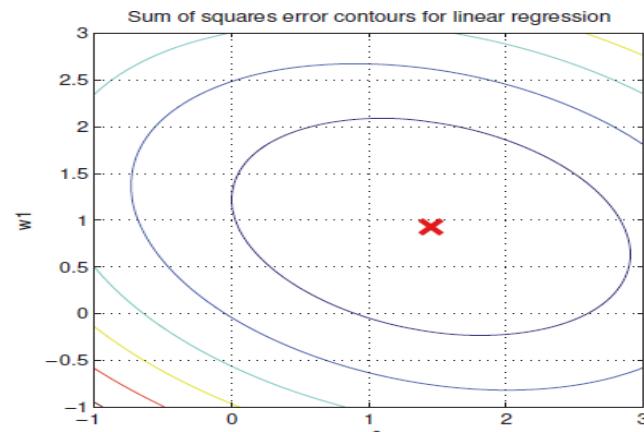
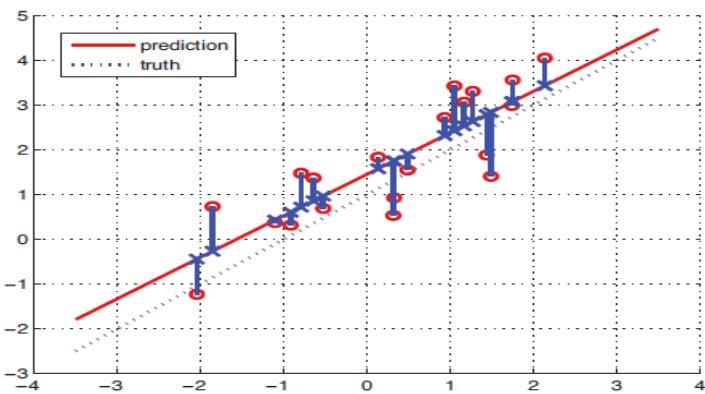
- Value range still infinite, but usually “fine” (care for exponentially distributed data -> apply log).

Empirical loss

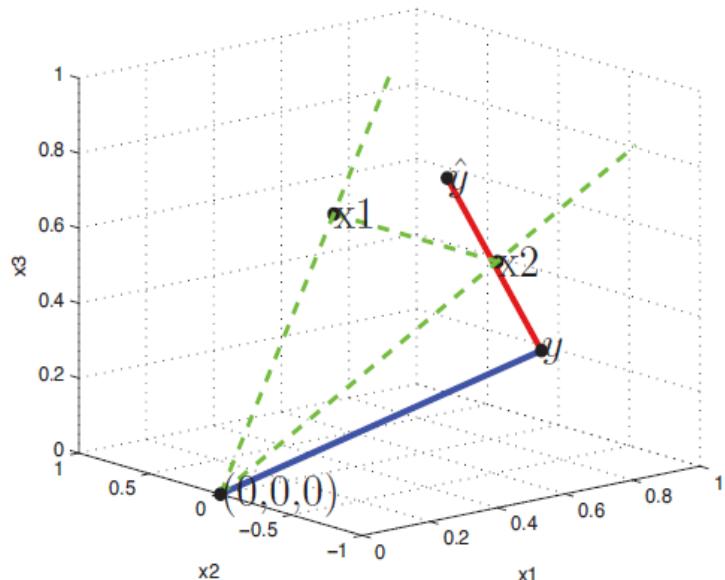
- Given a data set D of d-dimensional feature vectors x_1, \dots, x_n and labels y_1, \dots, y_n .
 - Define loss on a sample $l(y', y, x)$, y' prediction. Often independent of x .
 - i-th. loss: $l_i(\theta) = l(f_\theta(x_i), y_i)$.
 - **empirical loss**: $l(\theta) = \sum_{i=1}^n l_i(\theta)$.
- Minimizing the empirical loss is **empirical risk minimization**.

Least Squares

- Model parameter Θ is a d-dim. **weight vector w**.
- $l(w) = RSS(w) = \sum_{i=1}^n (y_i - w^t x_i)^2$ (**residual sum of squares**)



Geometric Proof ($N \geq D$)



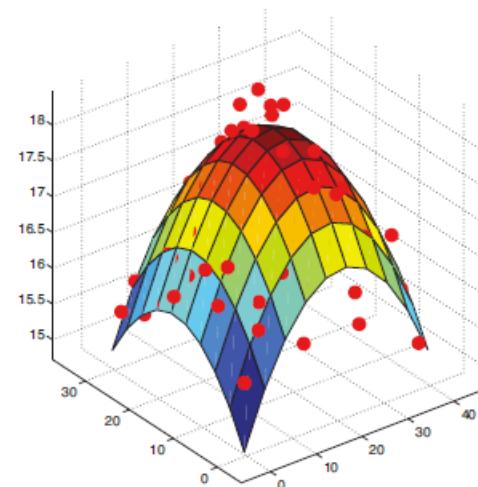
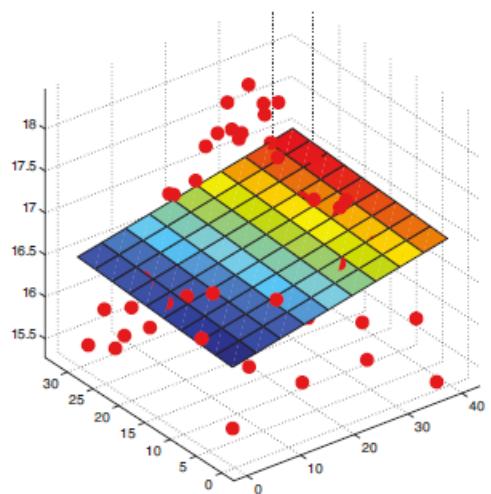
Idea: Project y on the **column space** of X (data matrix with instances as rows). Coordinate vector of y is then w .

$$\begin{aligned}P &= X(X^T X)^{-1} X^T \\ \Rightarrow Pw &= y' = Xw' \\ \Rightarrow w' &= (X^T X)^{-1} X^T y.\end{aligned}$$

P is called **hat matrix**. This formula can also be calculated by calculus: Derive loss by w and equate to 0, solve by w like in school.

Carl-Friedrich Gauß (1777-1855)

Polynomial feats on remote sensing data



Polynomial Regression

```
from sklearn.preprocessing import PolynomialFeatures
from sklearn.pipeline import make_pipeline
from sklearn.linear_models import ElasticNet

clf_pipe = make_pipeline(
    PolynomialFeatures(degree=3),
    ElasticNet(alpha=0.1, l1_ratio=0.9, normalize=True,
random_state=42)
)

clf_pipe.fit(X_train, y_train)
```

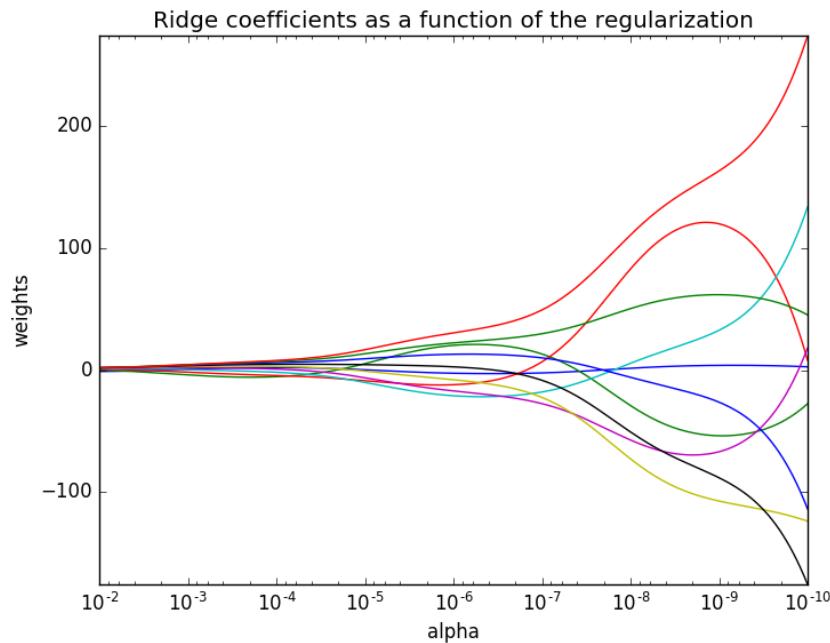
Problems with Least Squares

- Weights become arbitrarily large → overfitting.
- $(X^T X)$ sometimes not invertible: $(X^T X + \lambda I)$ for a non-negative lambda is more robust.
- Equivalent to modifying loss to:

$$\begin{aligned} l(w, \lambda) &= RSS(w) + \lambda L_2(w) \\ &= \sum_{i=1}^n (y_i - w^T x_i)^2 + \lambda (w_1^2 + \dots + w_d^2) \end{aligned}$$

- This is **ridge regression** or linear regression with L2 regularization.

Lambda / Weights



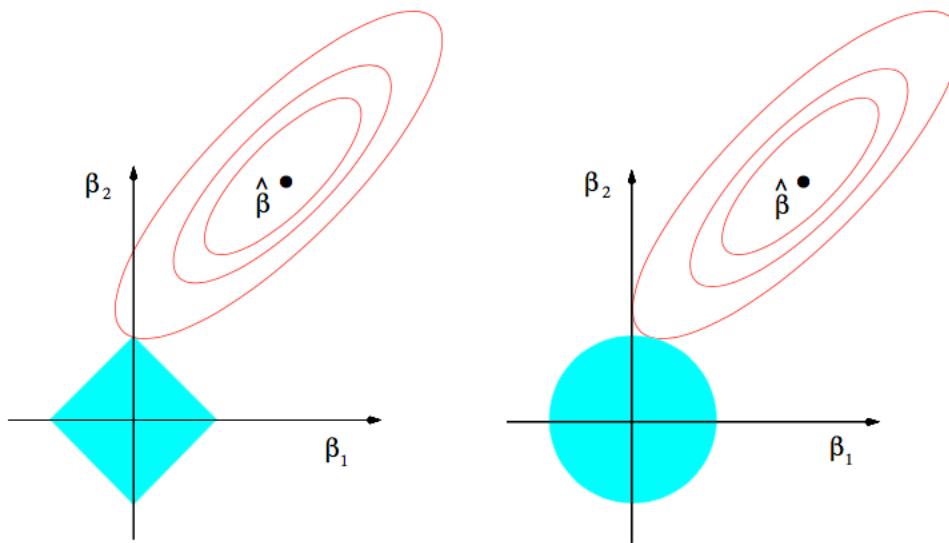
Lambda

- Lambda is a **hyper parameter**.
- Chosen on a log-scale (try ...0.05, 0.1, 0.5, 1, 5, ...).
- The smaller the lambda, the larger the weights can get and the model can become more complex.
- More data → smaller lambda.
- If you add more features → larger lambda. **But** that depends on the precise feature added and how it correlates to the target variable.

Other variants

- **Lasso:** $\lambda L_1(w) = \lambda(|w_1| + \dots + |w_d|)$
 - Leads to **sparsity** of features (presses weights to 0).
- **Elastic net:** Combine both L1 and L2 on all weights.
 - better deals with correlated features,
 - harder to tune as a new hyper parameter is added.
- **Generalized Linear Models (GLMs)**

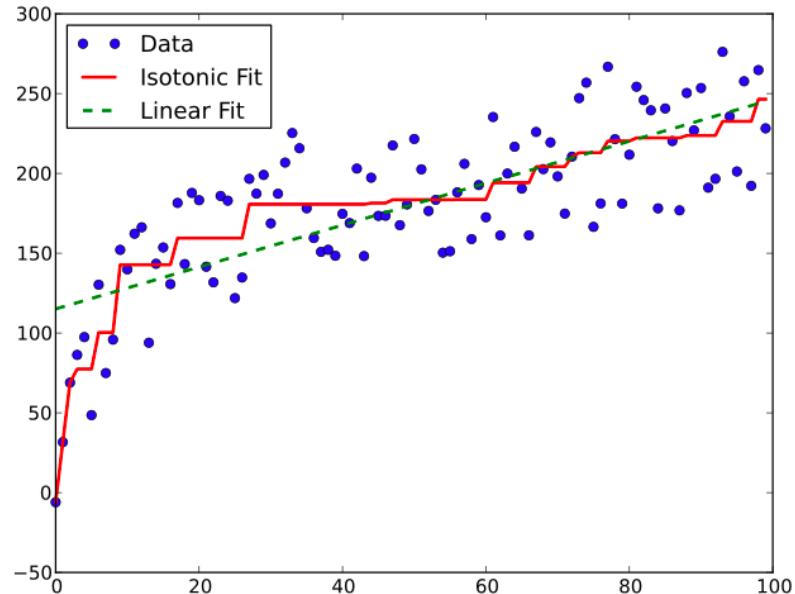
L1-Sparsity



Special variants

- **Robust regression:** More stable to outliers, rarely used as one has to solve a quadratic program (QP).
- **Bayesian Linear Regression:** Do not compute only one weight, but keep a distribution the weights!
- **Isotonic Regression:** Finds a non-decreasing piecewise linear curve to fit the model.

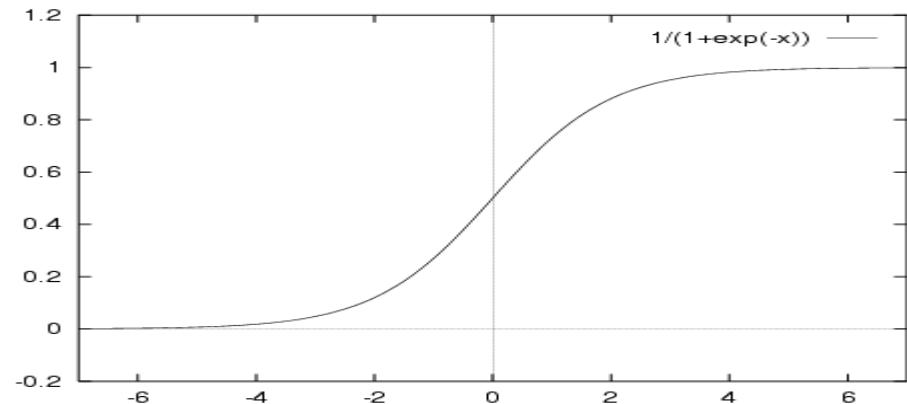
Isotonic Regression



Logistic Regression

- Logistic Regression is simply linear regression + **squashing**, with the **sigmoid** function:

$$\bullet \sigma(h) = \frac{1}{1+e^{-h}}, \quad h = w^t x$$



Logistic Regression II

- Outputs a probability in [0, 1] that outputs the likelihood of the label being 1: $P(y = 1 | x, D)$
→ binary classification problem.

$$\text{logloss}(w) = \sum_{i=1}^n \ln(1 + e^{-y_i(w^T x_i)})$$

- **multinomial logistic regression:** (w_i are model parameters)

$$P(y = c | x, w_1, \dots, w_C) = \frac{\exp(w_c^T x)}{\sum_{c'=1}^C \exp(w_{c'}^T x)}$$

Alternative: one-vs-all approach.

Logistic Regression

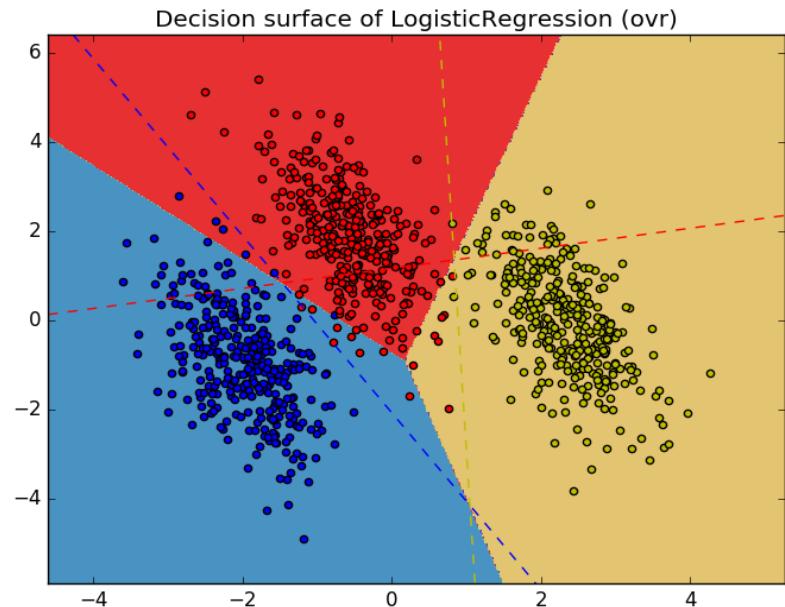
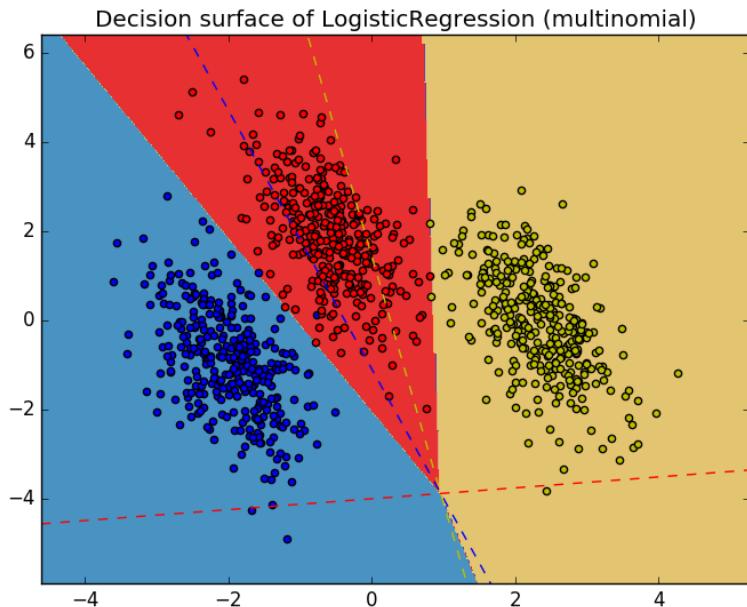
```
from sklearn.linear_model import LogisticRegression

lr_clf = LogisticRegression(
penalty="l2", # or "l1"
C=1./lamb, # capacity
fit_intercept=true, # add bias?
n_jobs=-1) # -1 = all cores

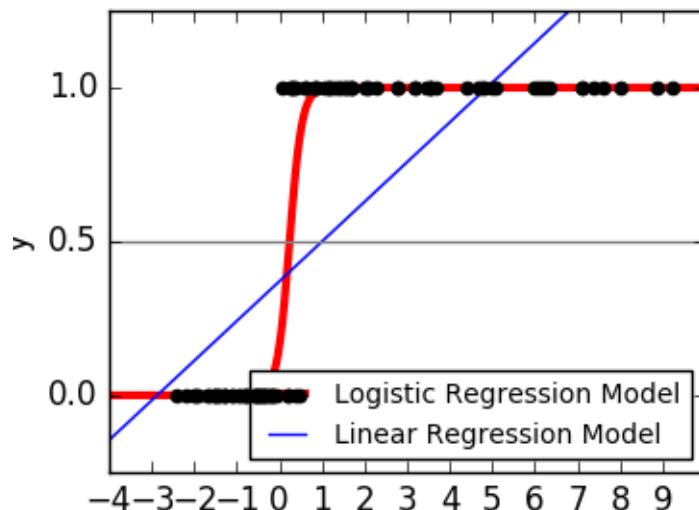
lr_clf.fit(X_train, y_train)
y_pred = lr_clf.predict_proba(X_test)
# or predict for labels

lr_clf.coef_ # the coefficients
```

Multinomial vs One-vs-rest



Logistic Regression III



Multinomial Regression

```
clf = LogisticRegression(  
    C=50. / train_samples,  
    multi_class='multinomial',  
    penalty='l1',  
    solver='saga')
```

Model Evaluation

Outlook

- Understanding Model fitting vs selection vs evaluation:
 - Resampling: cross validation (CV) and nested CV,
 - Model selection criteria,
 - Overfitting, Over-/Underselection, Bias-variance tradeoff.
- Evaluation metrics in regression and classification:
 - ROC curves.

Goal of ML

- Model is supposed to learn from shown patterns to generalize from this data.
- How do we measure generalization performance of the model?
Empirical risk minimization (ERM) framework.

”When you have two competing theories that make exactly the same predictions, the simpler one is the better.” --- ***Occam's Razor***

Empirical Risk Minimization

- Two assumptions:
 - Data follows a distribution $p(x, y)$ on $X \times Y$.
 - Learner sees data set $D = \{(x_1, y_1), \dots, (x_N, y_N)\}$ of **independent** and **identically sampled** (from p) data (*IID assumption*).
- **The IID assumption is almost always broken in practice.**

Empirical Loss

- Given a loss function l the *empirical loss/risk* is,

$$R_D[f] = \sum_{n=1}^N l(f(x_n), y_n, x_n).$$

- This is a statistical estimator if you vary the selection of D .

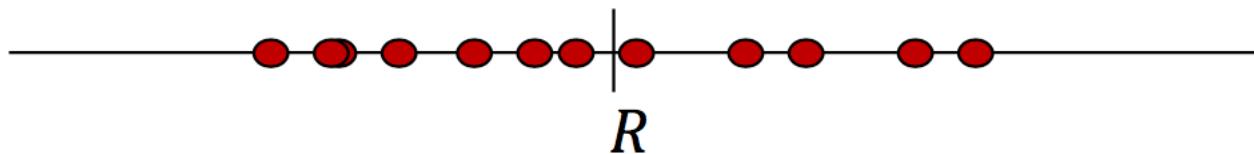
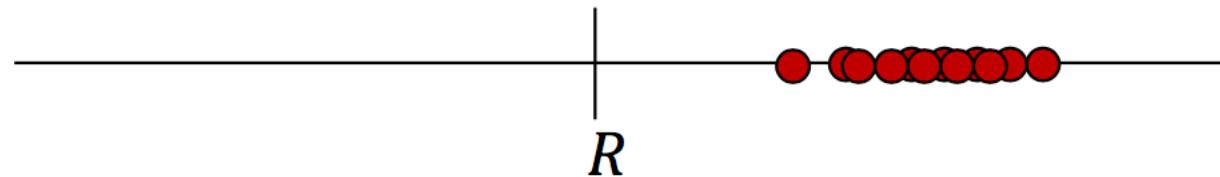
Empirical Risk Minimization II

- $R_D[f]$ is supposed to estimate the *generalization error/risk*,

$$R[f] = \int l(f(x), y, x) dp(x, y).$$

- The larger $|D|$ the more accurate.
- $R_D[f]$ has a bias (distance to $R[f]$) and a variance.

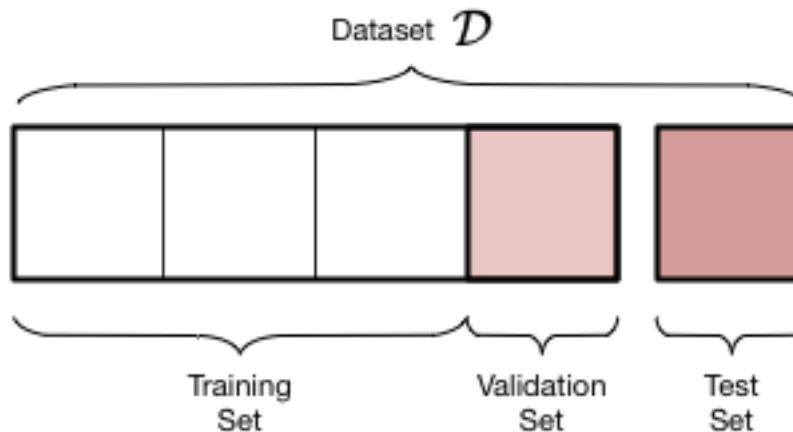
Bias / Variance



- **Model fitting:** finding the best model parameter:

$$\theta = \operatorname{argmin}_{\theta} R_D[f_{\theta}]$$

- **Model selection:** find the best fitting model family / hyperparams.
- **Model evaluation:** estimate the generalization risk.



On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation

Gavin C. Cawley

Nicola L. C. Talbot

School of Computing Sciences

University of East Anglia

Norwich, United Kingdom NR4 7TJ

GCC@CMP.UEA.AC.UK

NLCT@CMP.UEA.AC.UK

Editor: Isabelle Guyon

Abstract

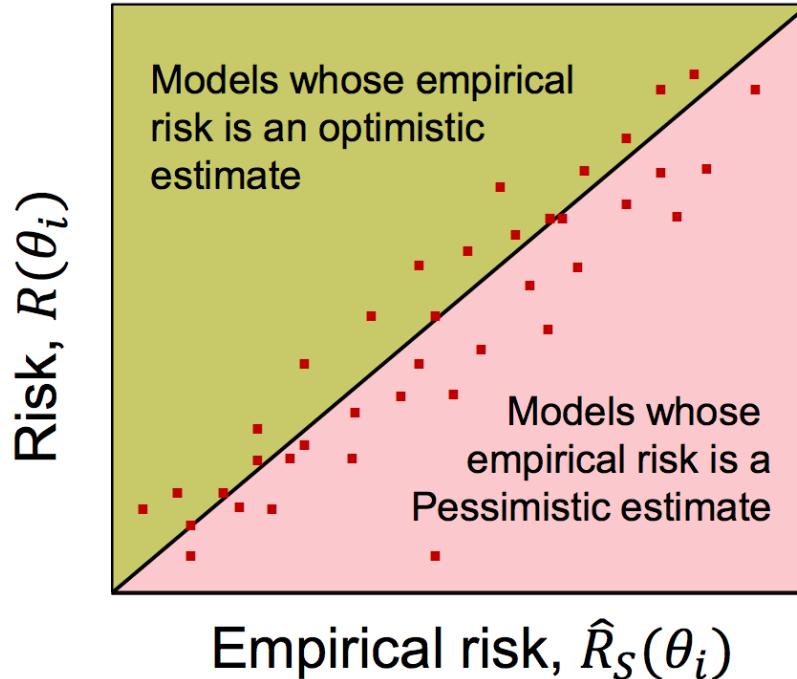
Model selection strategies for machine learning algorithms typically involve the numerical optimisation of an appropriate model selection criterion, often based on an estimator of generalisation performance, such as k -fold cross-validation. The error of such an estimator can be broken down into bias and variance components. While unbiasedness is often cited as a beneficial quality of a model selection criterion, we demonstrate that a low variance is at least as important, as a non-negligible variance introduces the potential for over-fitting in model selection as well as in training the model. While this observation is in hindsight perhaps rather obvious, the degradation in performance due to over-fitting the model selection criterion can be surprisingly large, an observation that appears to have received little attention in the machine learning literature to date. In this paper, we

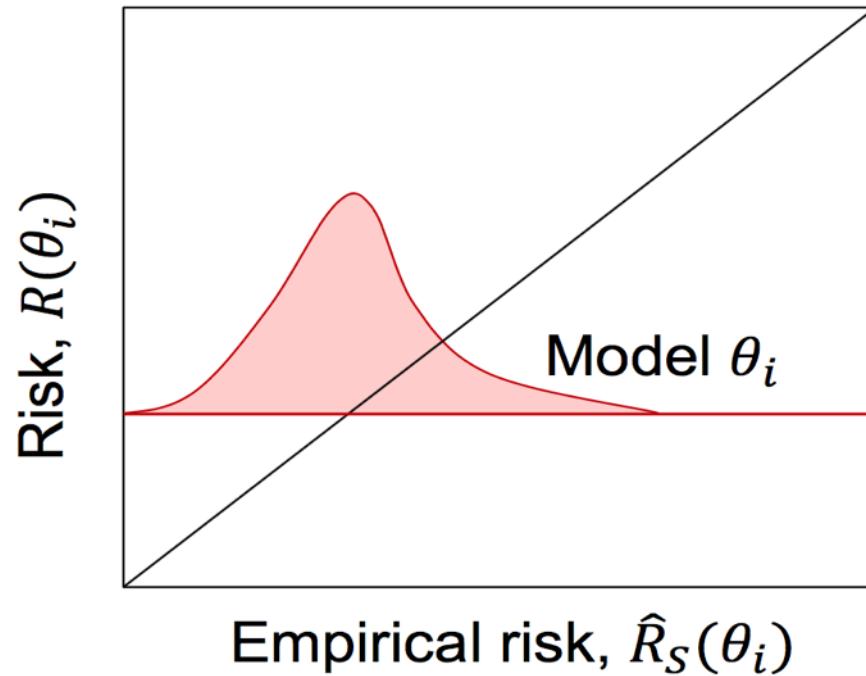
Prefer to call it “over-selection”

In “Learning with Kernels” from Smola & Schölkopf they name ex. 5.10. “overfitting on the test set”.

Paper: <http://bit.ly/2gBIR1M>

Parameter space, $\theta_i \in \Theta$



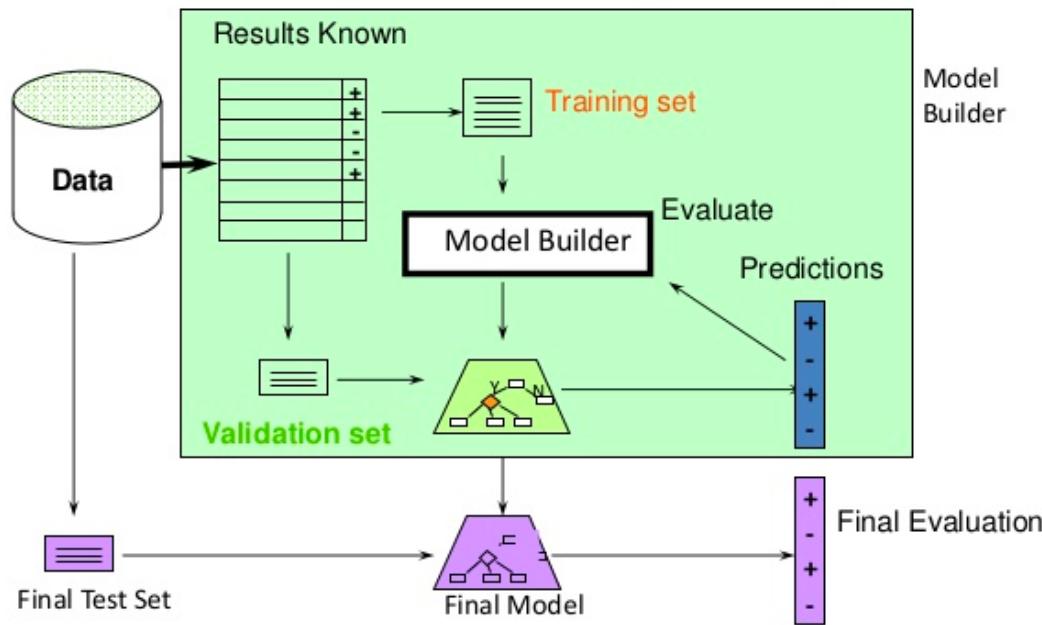


Messing up your experiments

- Data split strategy is part of experiment.
- Mainly care for:
 - Class distribution
 - Problem domain relevant issues such as time

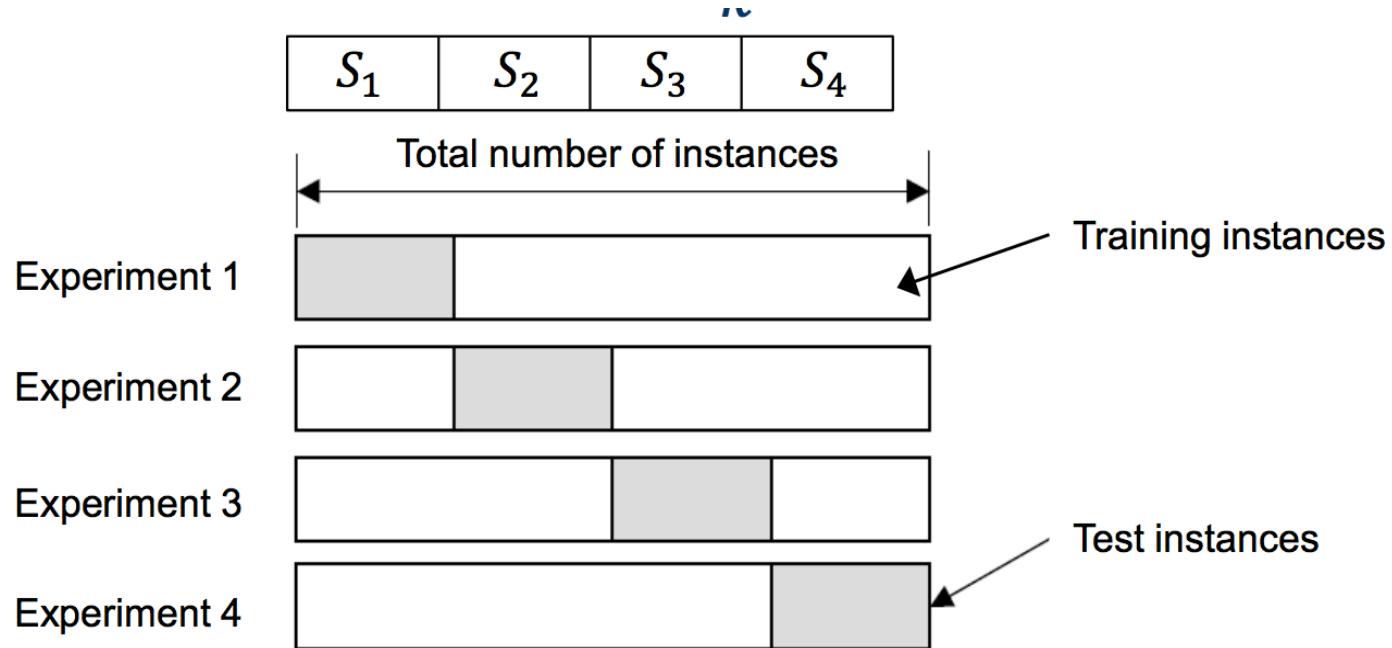
*“Validation and Test sets should model nature
and nature is not accommodating.” --- **Data
Scientist’s Proverbs***

Classification: Train, Validation, Test split

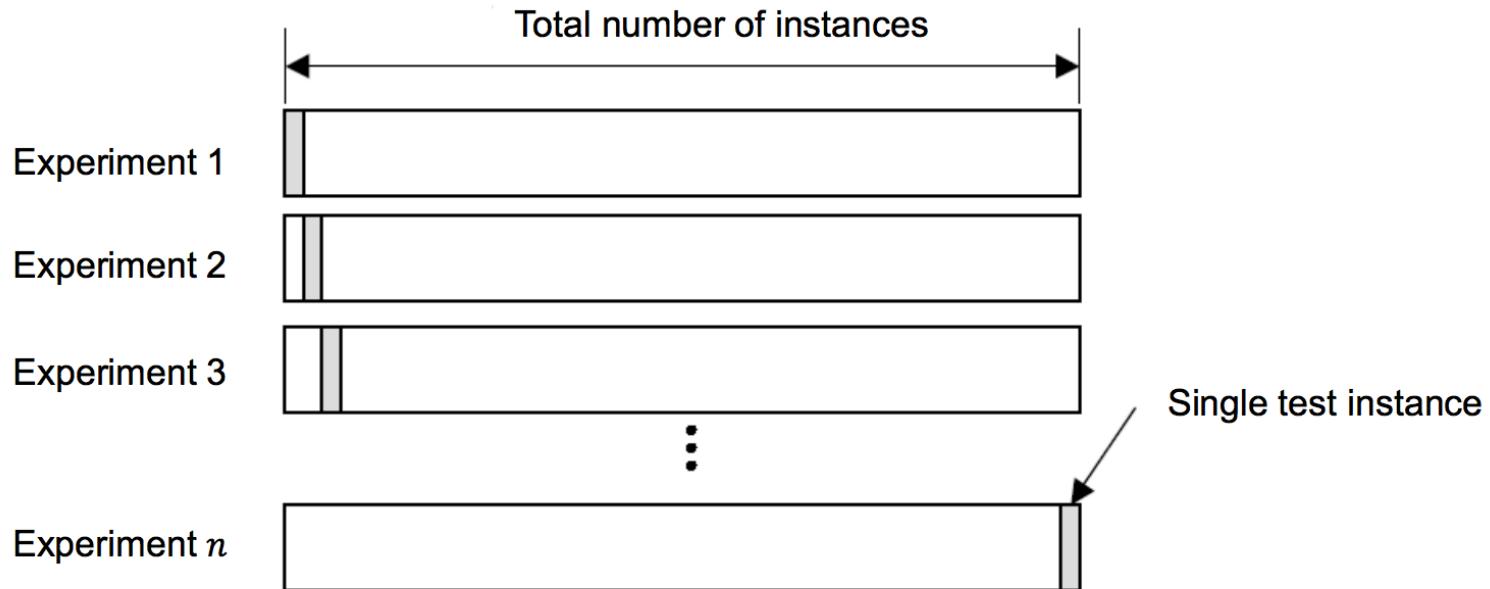


16

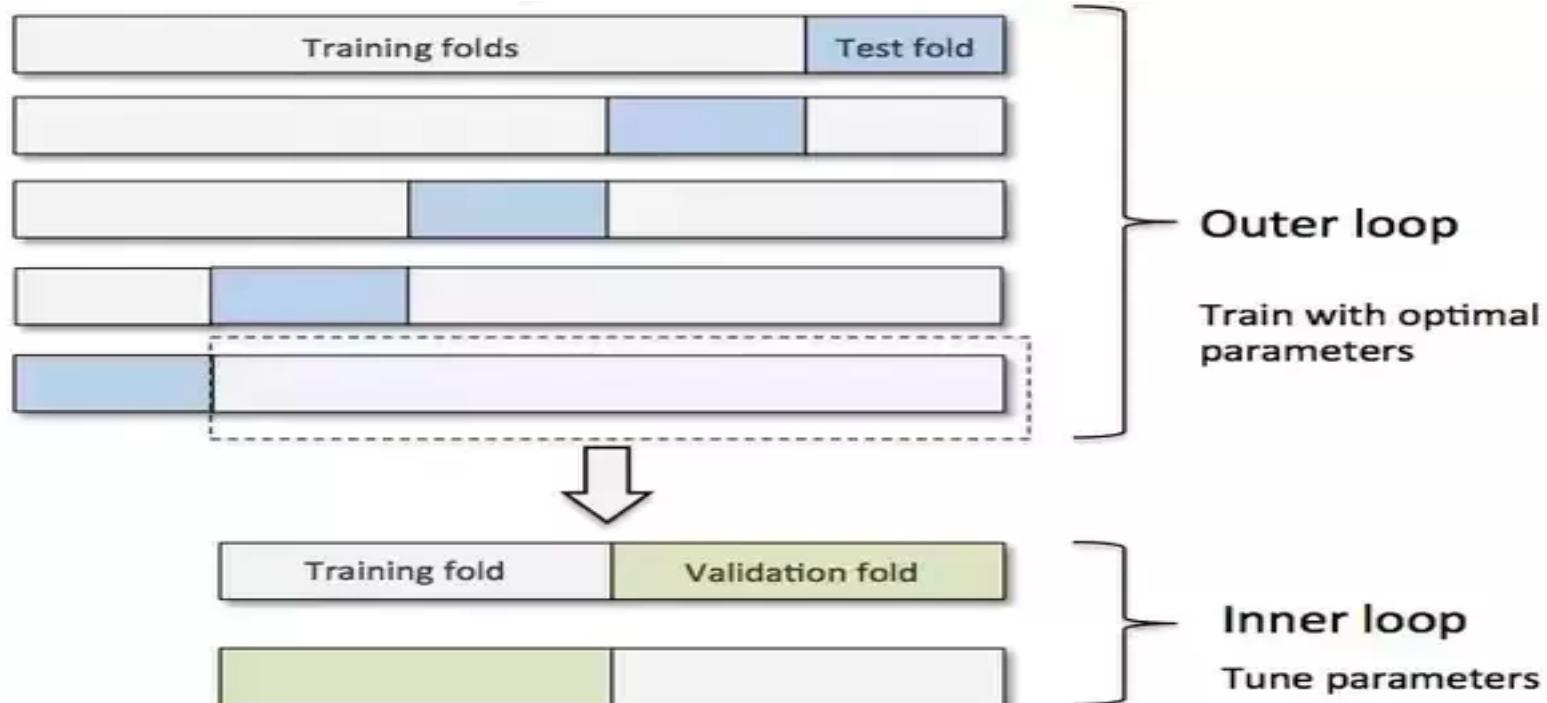
CV



LOOCV



Nested CV



```
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import ShuffleSplit
from sklearn.model_selection import GridSearchCV

base_clf = LogisticRegression(C=1, solver="saga", multi_class="multinomial")

param = {
    "C": [0.03, 0.01, 0.1, 1, 10, 100, 1000],
    "penalty": ("l1", "l2")
}

clf = GridSearchCV(base_clf, param, scoring="accuracy", cv=ShuffleSplit(3),
verbose=1)
scores = cross_val_score(clf, X, y, scoring="accuracy", cv=ShuffleSplit(5), n_jobs=-1,
verbose=1)

print("% .4f (%+.4f)" % (np.mean(scores), np.std(scores)))
```

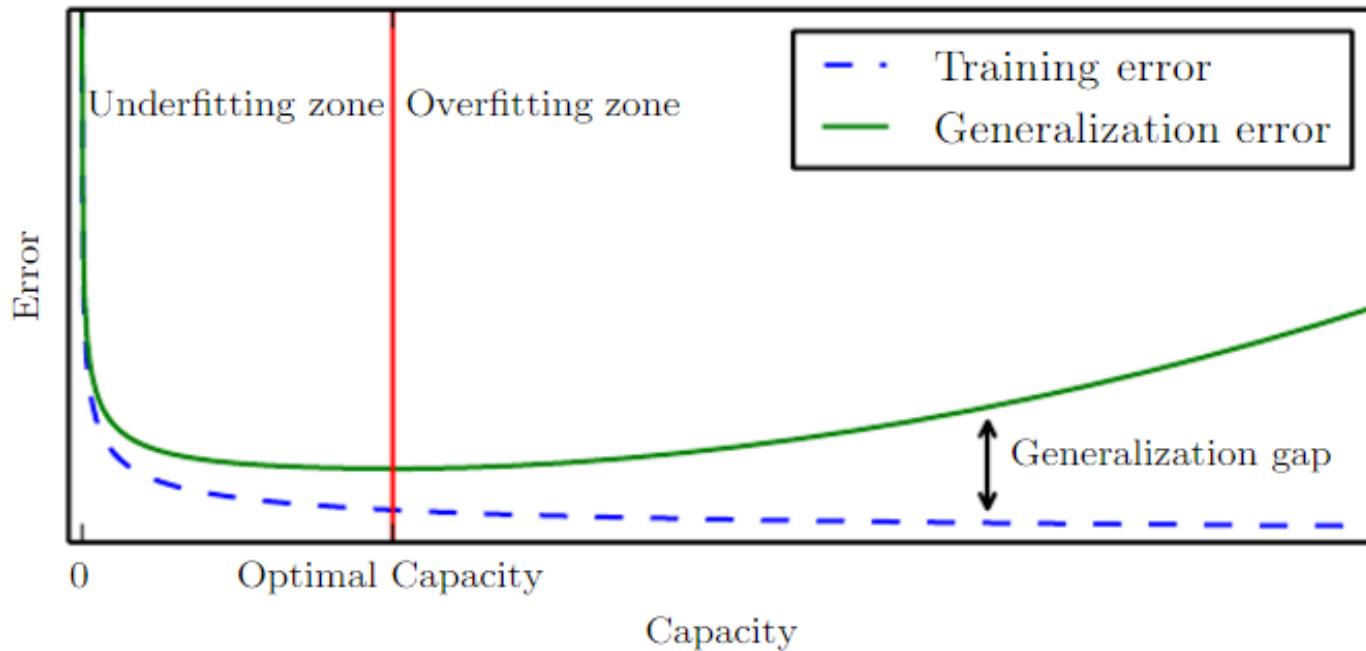
Alternative – Central Limit Theorem

- Given test size of n elements the confidence interval for the estimate e is

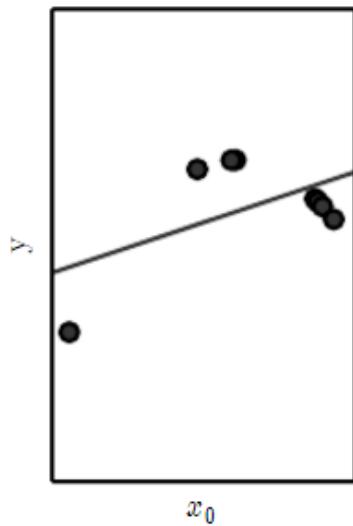
$$e \pm z \sqrt{\frac{1}{n} e(1 - e)}$$

where z is set to the $1 - \frac{\alpha}{2}$ quantile of the normal distribution.

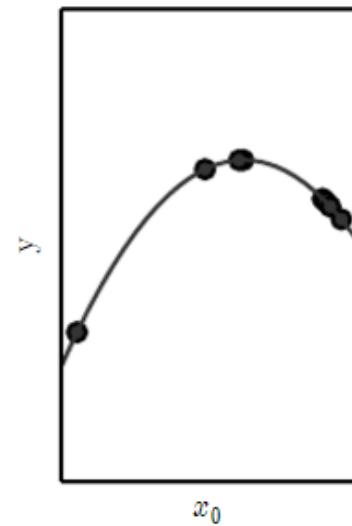
- Example: want 95% CI, $n = 20$, accuracy = 85% $\Rightarrow e = 0.85$, $\alpha = 0.05 \Rightarrow z = 1.96 \Rightarrow \text{CI} = 0.85 \pm 1.96 \sqrt{\frac{1}{20} 0.85(1 - 0.85)} = 85\% \pm 15.6\%$. For $n = 100$: $85\% \pm 7\%$.
- Models test process as binomial, then CLT.



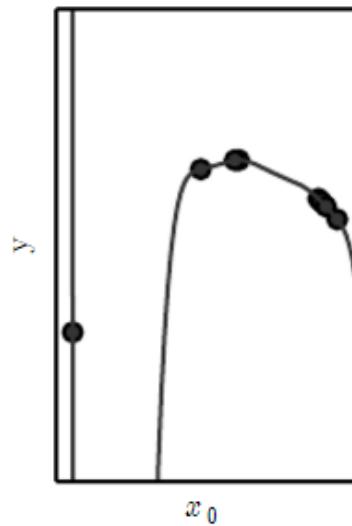
Underfitting



Appropriate capacity



Overfitting



Evaluation Metrics

Regression

- Simple approach: compare errors (eg **RMSE**)
- Or R-squared (variance explained). If the models do not have the same complexity, then use adjusted R-squared
- There is no absolute standard for a "good" value of adjusted R-squared

		Condition (as determined by "Gold standard")	
Total population		Condition positive	Condition negative
Test outcome	Test outcome positive	True positive	False positive (Type I error)
	Test outcome negative	False negative (Type II error)	True negative

https://en.wikipedia.org/wiki/Precision_and_recall

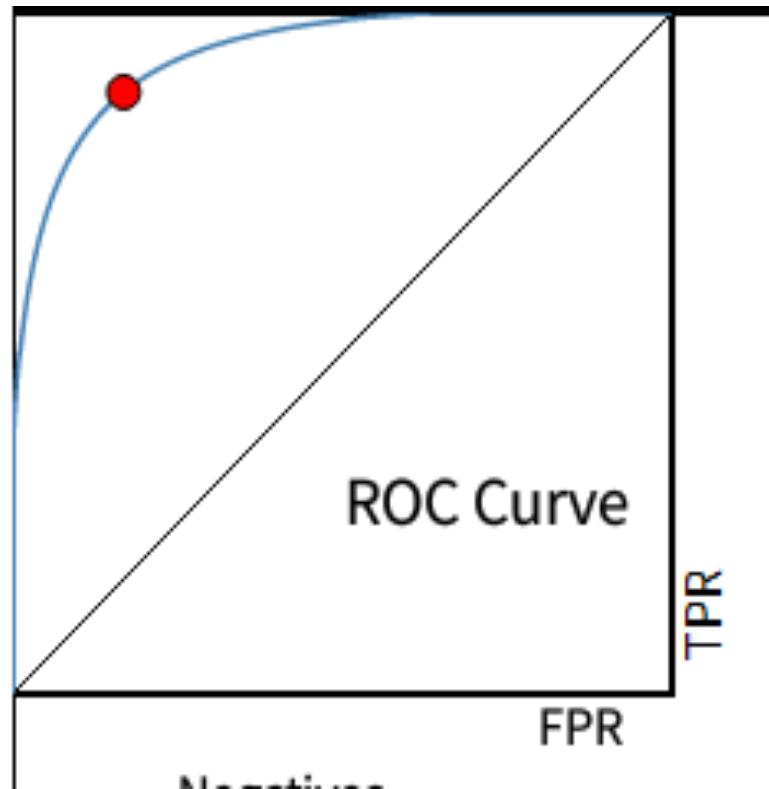
		Condition (as determined by "Gold standard")		
Total population		Condition positive	Condition negative	Prevalence = $\frac{\sum \text{Condition positive}}{\sum \text{Total population}}$
Test outcome	Test outcome positive	True positive	False positive (Type I error)	Positive predictive value (PPV), Precision $= \frac{\sum \text{True positive}}{\sum \text{Test outcome positive}}$
	Test outcome negative	False negative (Type II error)	True negative	False omission rate (FOR) $= \frac{\sum \text{False negative}}{\sum \text{Test outcome negative}}$
Accuracy (ACC) = $\frac{\sum \text{True positive} + \sum \text{True negative}}{\sum \text{Total population}}$	True positive rate (TPR), Sensitivity, Recall $= \frac{\sum \text{True positive}}{\sum \text{Condition positive}}$	False positive rate (FPR), Fall-out = $\frac{\sum \text{False positive}}{\sum \text{Condition negative}}$	Positive likelihood ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) $= \frac{\text{LR+}}{\text{LR-}}$
	False negative rate (FNR), Miss rate $= \frac{\sum \text{False negative}}{\sum \text{Condition positive}}$	True negative rate (TNR), Specificity (SPC) $= \frac{\sum \text{True negative}}{\sum \text{Condition negative}}$	Negative likelihood ratio (LR-) $= \frac{\text{FNR}}{\text{TNR}}$	

ROC curve

y -axis is true positive rate, and the
 x -axis is false positive rate

Interpretation:

Pick a random negative and a random positive example; The AUC gives you the probability that your classifier assigns a higher score to the positive example (ie, ranks the positive higher than the negative).



ROC curve

- The most common method for combining sensitivity and specificity into a single value uses the **receiver operating characteristic (ROC)** curve.
- The ROC curve is useful for determining alternate cutoffs for class probabilities

Precision and recall

- **Origins:** information retrieval
- **Precision** is the probability that a (randomly selected) retrieved document is relevant.
- **Recall** is the probability that a (randomly selected) relevant document is retrieved in a search.
- They are balanced: you can increase one at the cost of the other

		Condition (as determined by "Gold standard")			
Total population		Condition positive	Condition negative	Prevalence = $\frac{\sum \text{Condition positive}}{\sum \text{Total population}}$	
Test outcome	Test outcome positive	True positive	False positive (Type I error)	Positive predictive value (PPV), Precision $= \frac{\sum \text{True positive}}{\sum \text{Test outcome positive}}$	False discovery rate (FDR) $= \frac{\sum \text{False positive}}{\sum \text{Test outcome positive}}$
	Test outcome negative	False negative (Type II error)	True negative	False omission rate (FOR) $= \frac{\sum \text{False negative}}{\sum \text{Test outcome negative}}$	Negative predictive value (NPV) $= \frac{\sum \text{True negative}}{\sum \text{Test outcome negative}}$
Accuracy (ACC) = $\frac{\sum \text{True positive} + \sum \text{True negative}}{\sum \text{Total population}}$		True positive rate (TPR), Sensitivity, Recall $= \frac{\sum \text{True positive}}{\sum \text{Condition positive}}$	False positive rate (FPR), Fall-out $= \frac{\sum \text{False positive}}{\sum \text{Condition negative}}$	Positive likelihood ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) $= \frac{\text{LR+}}{\text{LR-}}$
		False negative rate (FNR), Miss rate $= \frac{\sum \text{False negative}}{\sum \text{Condition positive}}$	True negative rate (TNR), Specificity (SPC) $= \frac{\sum \text{True negative}}{\sum \text{Condition negative}}$	Negative likelihood ratio (LR-) $= \frac{\text{FNR}}{\text{TNR}}$	

		Condition (as determined by "Gold standard")		
Total population		Condition positive	Condition negative	Prevalence = $\frac{\sum \text{Condition positive}}{\sum \text{Total population}}$
Test outcome	Test outcome positive	True positive	False positive (Type I error)	Positive predictive value (PPV), Precision $= \frac{\sum \text{True positive}}{\sum \text{Test outcome positive}}$
	Test outcome negative	False negative (Type II error)	True negative	False omission rate (FOR) $= \frac{\sum \text{False negative}}{\sum \text{Test outcome negative}}$
	Accuracy (ACC) = $\frac{\sum \text{True positive} + \sum \text{True negative}}{\sum \text{Total population}}$	True positive rate (TPR), Sensitivity $= \frac{\sum \text{True positive}}{\sum \text{Condition positive}}$	False positive rate (FPR), Fall-out $= \frac{\sum \text{False positive}}{\sum \text{Condition negative}}$	Positive likelihood ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$
		False negative rate (FNR), Miss rate $= \frac{\sum \text{False negative}}{\sum \text{Condition positive}}$	True negative rate (TNR), Specificity (SPC) $= \frac{\sum \text{True negative}}{\sum \text{Condition negative}}$	Negative likelihood ratio (LR-) $= \frac{\text{FNR}}{\text{TNR}}$
				Diagnostic odds ratio (DOR) $= \frac{\text{LR+}}{\text{LR-}}$

ROC curve

- <http://www.navan.name/roc/>

