An Overview of Statistical Machine Learning

Dino Sejdinovic

Department of Statistics
University of Oxford

Oxford Prospects Programme, 24/01/2020

Statistical Machine Learning

Statistical Machine Learning refers to a vast set of tools for extracting information from (typically large quantities of) data, and is closely related to Data Science and Artifical Intelligence.

Statistical Machine Learning

Statistical Machine Learning refers to a vast set of tools for extracting information from (typically large quantities of) data, and is closely related to Data Science and Artifical Intelligence.

Massive amounts of data are being collected in many different fields: financial institutions, businesses, governments, healthcare organisations, and universities are all interested in utilizing and making sense of data they collect.

Statistical Machine Learning

Statistical Machine Learning refers to a vast set of tools for extracting information from (typically large quantities of) data, and is closely related to Data Science and Artifical Intelligence.

Massive amounts of data are being collected in many different fields: financial institutions, businesses, governments, healthcare organisations, and universities are all interested in utilizing and making sense of data they collect.

The majority of mathematics and engineering students will go on to work in careers that involve carrying out or interpreting analysis of data.

What is Machine Learning?

Arthur Samuel, 1959

Field of study that gives computers the ability to **learn** without being explicitly programmed.

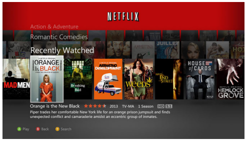
Tom Mitchell, 1997

Any computer program that **improves its performance** at some task **through experience**.

Kevin Murphy, 2012

To develop methods that can **automatically** detect **patterns in data**, and then to use the uncovered patterns to **predict** future data or other outcomes of interest.

What is Machine Learning?



recommender systems

machine translation

self-driving cars

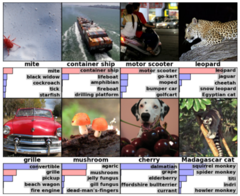
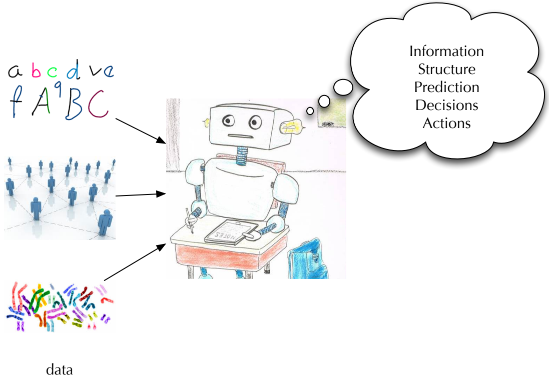


image recognition

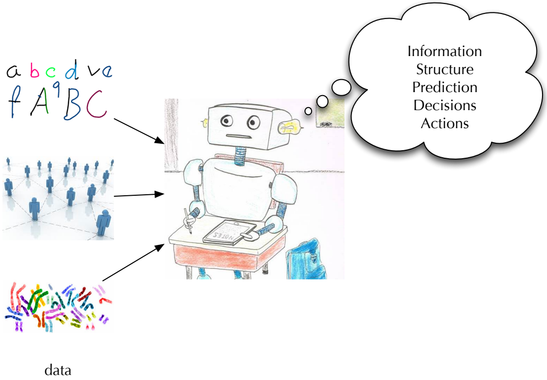
DQN Atari games

AlphaGo

Machine Learning



Machine Learning



"...procedures for analyzing data, techniques for interpreting the results of such procedures, ways of planning the gathering of data to make its analysis easier, more precise or more accurate, and all the machinery and results of (mathematical) statistics which apply to analyzing data..."
John Tukey, *The Future of Data Analysis*, 1962.

Statistical Machine Learning

Examples of recent advances in AI which make use of machine learning models:

- learning game strategies from sensory input,
- computer vision,
- machine translation,
- AlphaGO.

Types of Machine Learning

Supervised learning

- Data contains "labels": every example is an input-output pair
- classification, regression
- Goal: *prediction on new examples*

Unsupervised learning

- Extract key features of the "unlabelled" data
- clustering, signal separation, density estimation
- Goal: *representation, hypothesis generation, visualization*

Types of Machine Learning

Semi-supervised Learning

A database of examples, only a small subset of which are labelled.

Multi-task Learning

A database of examples, each of which has multiple labels corresponding to different prediction tasks.

Reinforcement Learning

An agent acting in an environment, given rewards for performing appropriate actions, learns to maximize their reward.

Software

- Python: scikit-learn, mlpy, Theano
- TensorFlow, Torch, Keras, Shogun, Weka.
- R
- Matlab/Octave

Overview of Unsupervised learning

In **unsupervised learning** we just have observations on the a set of variables $X_1, \ldots, X_p$, measured on n observations.

Overview of Unsupervised learning

In **unsupervised learning** we just have observations on the a set of variables $X_1, \ldots, X_p$, measured on n observations.

Interest lies in looking for patterns and structure in the data, which is often large and high-dimensional.

Overview of Unsupervised learning

In **unsupervised learning** we just have observations on the a set of variables $X_1, \ldots, X_p$, measured on n observations.

Interest lies in looking for patterns and structure in the data, which is often large and high-dimensional.

Relevant questions include

1 Can we find a way to visualize the data that is informative?

Overview of Unsupervised learning

In **unsupervised learning** we just have observations on the a set of variables $X_1, \ldots, X_p$, measured on n observations.

Interest lies in looking for patterns and structure in the data, which is often large and high-dimensional.

Relevant questions include

1. Can we find a way to visualize the data that is informative?
2. Can we compress the dataset without losing any relevant information?

Overview of Unsupervised learning

In **unsupervised learning** we just have observations on the a set of variables $X_1, \ldots, X_p$, measured on n observations.

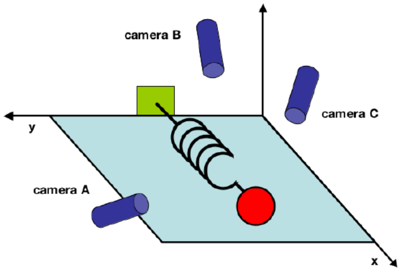
Interest lies in looking for patterns and structure in the data, which is often large and high-dimensional.

Relevant questions include

1. Can we find a way to visualize the data that is informative?
2. Can we compress the dataset without losing any relevant information?
3. Can we find separate subgroups (or clusters) of observations that describe the structure of the dataset?

Dimensionality reduction

- deceptively many variables to measure, many of them redundant / correlated to each other (large p)
- often, there is a simple but unknown underlying relationship hiding
- example: ball on a frictionless spring recorded by three different cameras
  - our imperfect measurements obfuscate the true underlying dynamics
  - are our coordinates meaningful or do they simply reflect the method of data gathering?



J. Shlens, A Tutorial on Principal Component Analysis, 2005

Principal Components Analysis (PCA)

- PCA considers interesting directions to be those with greatest *variance*.
- A *linear* dimensionality reduction technique: looks for a *new basis* to represent a noisy dataset.
- Workhorse for many different types of data analysis (often used for data preprocessing before supervised techniques are applied).
- Often the first thing to run on high-dimensional data.

Data Matrix notation

Notation

- Data consists of p variables (features/attributes/dimensions) on n examples (items/observations).
- $\mathbf{X} = (x_{ij})$ is a $n \times p$-matrix with $x_{ij} :=$ the j-th variable for the i-th example

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1j} & \dots & x_{1p} \\ x_{21} & x_{22} & \dots & x_{2j} & \dots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{i1} & x_{i2} & \dots & x_{ij} & \dots & x_{ip} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{nj} & \dots & x_{np} \end{bmatrix}.$$

- Denote the i-th data item by $x_i \in \mathbb{R}^p$ (we will treat it as a column vector: it is the transpose of the i-th row of $\mathbf{X}$).
- Assume $x_1, \dots, x_n$ are *independently and identically distributed* samples of a *random vector* X over $\mathbb{R}^p$. The j-th dimension of X will be denoted $X^{(j)}$.

PCA

PCA

Find an orthogonal basis $\{v_1, v_2, \dots, v_p\}$ for the data space such that:

- The first principal component (PC) v_1 is the *direction of greatest variance* of data.
- The j-th PC v_j is the *direction orthogonal to* $v_1, v_2, \dots, v_{j-1}$ *of greatest variance*, for $j = 2, \dots, p$.

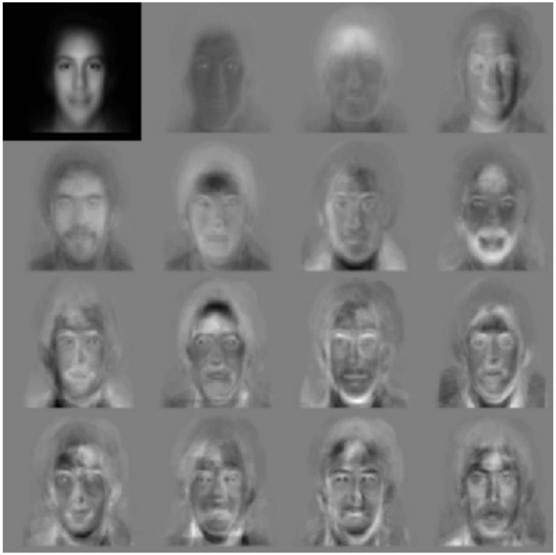
- Eigendecomposition of the sample covariance matrix $S = \frac{1}{n-1}\sum_{i=1}^{n} x_i x_i^\top$ (data is assumed centred).

$$S = \frac{1}{n-1}\mathbf{X}^\top \mathbf{X} = V \Lambda V^\top.$$

  - Λ is a diagonal matrix with eigenvalues (variances along each principal component) $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p \geq 0$
  - V (*loadings matrix*) is a $p \times p$ orthogonal matrix whose columns are the p eigenvectors of S, i.e. the principal components $v_1, \dots, v_p$
- Dimensionality reduction by projecting $x_i \in \mathbb{R}^p$ onto first k principal components, to obtain *scores matrix*:

$$z_i = \left[v_1^\top x_i, \dots, v_k^\top x_i\right]^\top \in \mathbb{R}^k.$$

PCA on Face Images: Eigenfaces



Turk and Pentland, CVPR 1995

PCA on European Genetic Variation

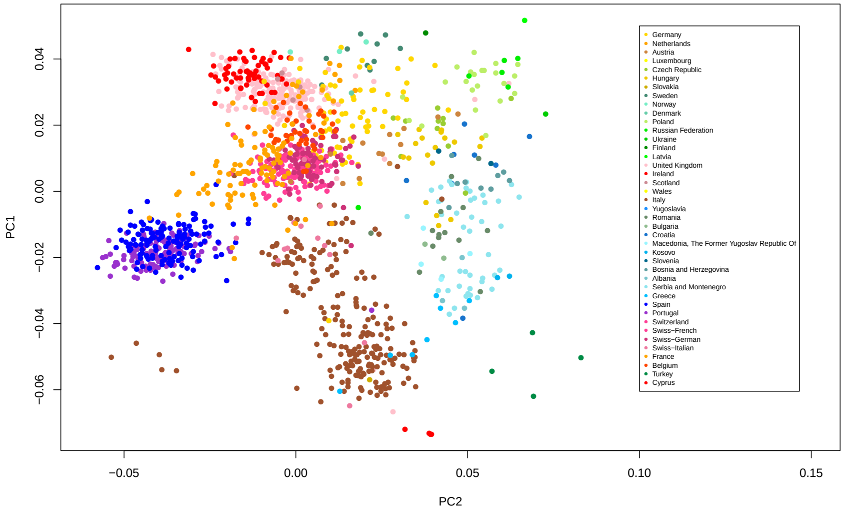
3,000 individuals from different European countries, each with measurements at $\sim$500,000 genes.

From the paper by Novembre et al. (2008) Nature 456:98-101

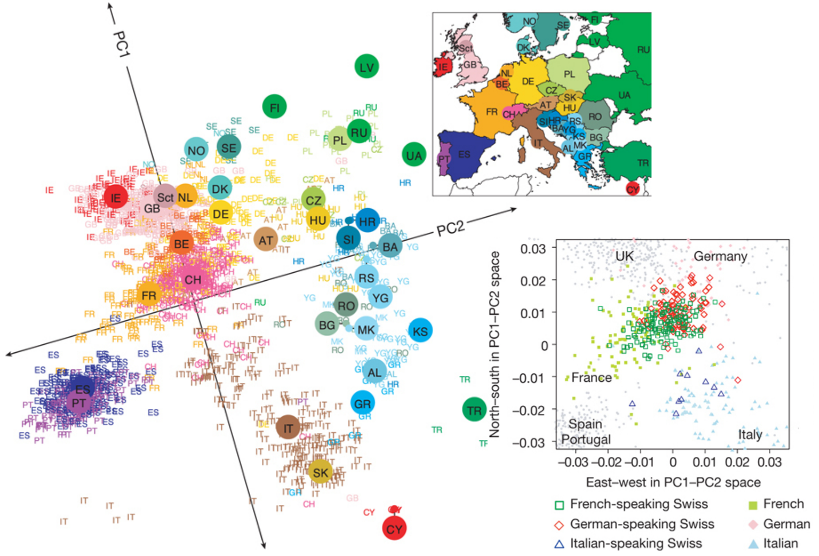
Scientific question

“not clear to what extent populations within continental regions exist as discrete genetic clusters versus as a genetic continuum, nor how precisely one can assign an individual to a geographic location on the basis of their genetic information alone."

PCA on European Genetic Variation



PCA on European Genetic Variation



Genes mirror geography within Europe, Nature 2008

Overview of Supervised Learning

Unsupervised learning:

- To "extract structure" and postulate hypotheses about data generating process from "unlabelled" observations of some data objects X.
- Visualize, summarize and compress data.

Supervised learning:

- In addition to the observations of X, we have access to their response variables / labels $Y \in \mathcal{Y}$: we observe $\{(x_i, y_i)\}_{i=1}^n$.
- Types of supervised learning:
  - Classification: discrete responses, e.g. $\mathcal{Y} = \{+1, -1\}$ or $\{1, \dots, K\}$.
  - Regression: a numerical value is observed and $\mathcal{Y} = \mathbb{R}$.

The goal is to accurately predict the response Y on new observations of X, i.e., to *learn a function* $f : \mathbb{R}^p \to \mathcal{Y}$, such that $f(X)$ will be close to the true response Y.

Loss function

- Suppose we made a prediction $\hat{Y} = f(X) \in \mathcal{Y}$ based on observation of X.
- How good is the prediction? We can use a *loss function* $L : \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R}^+$ to formalize the quality of the prediction.
- Typical loss functions:
  - *Misclassification loss* (or *0-1 loss*) for classification

$$L(y, f(x)) = \begin{cases} 0 & f(x) = y \\ 1 & f(x) \neq y \end{cases} .$$

  - *Squared loss* for regression

$$L(y, f(x)) = (f(x) - y)^2 .$$

- Many other choices are possible.

Risk

- paired observations $\{(x_i, y_i)\}_{i=1}^n$ viewed as i.i.d. realizations of a random variable (X, Y) on $\mathcal{X} \times \mathcal{Y}$ with joint distribution P_{XY}

Risk

For a given loss function L, the *risk* R of a learned function f is given by the expected loss

$$R(f) = \mathbb{E}_{P_{XY}}\left[L(Y, f(X))\right],$$

where the expectation is with respect to the true (unknown) joint distribution of (X, Y).

- The risk is unknown, but we can compute the *empirical risk*:

$$R_n(f) = \frac{1}{n}\sum_{i=1}^{n} L(y_i, f(x_i)).$$

Hypothesis Space and Empirical Risk Minimization

- The goal of learning is to find the function in *hypothesis space* $\mathcal{H}$ which minimises the risk:

$$f_\star = \operatorname*{argmin}_{f \in \mathcal{H}} \mathbb{E}_{X,Y}[L(Y, f(X))]$$

- *Empirical Risk Minimization* (ERM): minimize the empirical risk instead, since we typically do not know $P_{X,Y}$.

$$\hat{f} = \operatorname*{argmin}_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(x_i))$$

- Hypothesis space $\mathcal{H}$ is the space of functions f under consideration.
- How complex should we allow functions f to be? If hypothesis space $\mathcal{H}$ is "too large", ERM can lead to *overfitting*.

$$\hat{f}(x) = \begin{cases} y_i & \text{if } x = x_i, \\ 0 & \text{otherwise} \end{cases}$$

will have zero empirical risk, but is useless for generalization, since it has simply "memorized" the dataset.

Examples of Hypothesis Spaces

Say $\mathcal{X} \subseteq \mathbb{R}^p$.

- all linear functions $f(x) = w^\top x + b$, parametrized by $w \in \mathbb{R}^p$ and $b \in \mathbb{R}$

One-Dimensional Inputs

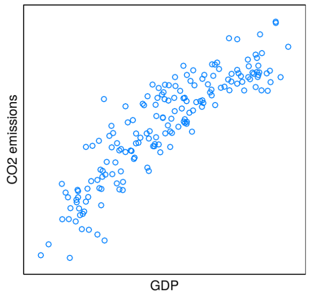


Figure 6.1. Scatterplot of measures of CO_2 emissions per head against GDP per head, for 178 countries.

A simple model for the dependence of y_i on x_i is:

$$y_i = b + wx_i + \text{“error”}$$

The errors represent a random scatter of the points (x_i, y_i) about the unknown line $y = b + wx$, which we wish to infer, i.e., the parameters are $\theta = (b, w)$.

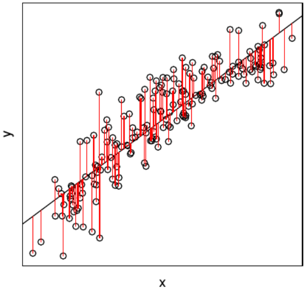
Estimating the parameters $\theta = (b, w)$

- In order to complete the model, we need to specify the *noise distribution*.
- The simplest choice is to use noise $\epsilon_i = y_i - b - wx_i \sim \mathcal{N}(0, \sigma^2)$ - we regard σ^2 as being known but it can also be inferred from the data.
- Thus, we can write down the likelihood

$$\begin{aligned} L(w,b) &= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}\left(y_i - b - wx_i\right)^2\right) \\ &= \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\left(y_i - b - wx_i\right)^2\right) \end{aligned}$$

Maximizing the likelihood is equivalent to minimizing the *sum of squares* $S(w,b) = \sum_{i=1}^{n}(y_i - b - wx_i)^2$ (*least-squares estimation*), i.e. empirical risk with respect to the squared loss.

Estimating the parameters $\theta = (b, w)$



Vertical distances from the points (x_i, y_i) to the line $y = b + wx$ represent the errors. Least-squares estimates of b and w minimize the sum of these squared distances.

To find estimates, we calculate and set to zero:

$$\frac{\partial S}{\partial b} = -2\sum_{i=1}^{n}(y_i - b - wx_i) \qquad \frac{\partial S}{\partial w} = -2\sum_{i=1}^{n} x_i(y_i - b - wx_i)$$

Assuming *centred inputs*, i.e. $\sum_{i=1}^{n} x_i = 0$, we obtain:

$$\hat{b} = \frac{1}{n}\sum_{i=1}^{n} y_i \quad \hat{w} = \frac{\sum_{i=1}^{n} x_i y_i}{\sum_{i=1}^{n} x_i^2}.$$

Multiple Linear Regression

Now we have p explanatory variables and

$$y_i = b + w_1 x_{i1} + w_2 x_{i2} + \cdots + w_p x_{ip} + \epsilon_i = b + w^\top x_i + \epsilon_i$$

and we wish to minimize the empirical risk:

$$\min_{w,b} \frac{1}{n} \sum_{i=1}^{n} (y_i - w^\top x_i - b)^2$$

Intercept b can be estimated by $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$. For w by differentiating and setting to zero there is a closed form solution

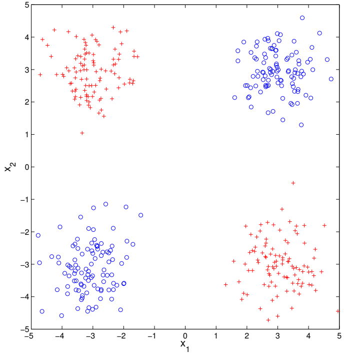
$$\hat{w} = \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \mathbf{X}^\top \mathbf{y}.$$

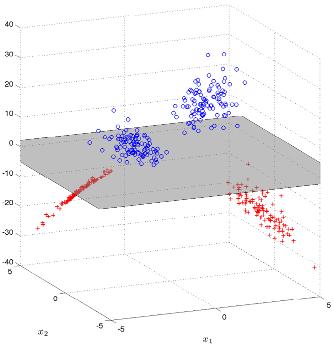
Examples of Hypothesis Spaces

Say $\mathcal{X} \subseteq \mathbb{R}^p$.

- all linear functions $f(x) = w^\top x + b$, parametrized by $w \in \mathbb{R}^p$ and $b \in \mathbb{R}$
- consider a specific *nonlinear feature expansion* $\varphi : \mathcal{X} \to \mathbb{R}^D$, with $D > p$ and use functions linear in those features: $f(x) = w^\top \varphi(x) + b$, but nonlinear in the original inputs $\mathcal{X}$, parametrized by $w \in \mathbb{R}^D$ and $b \in \mathbb{R}$. For example, starting with $\mathcal{X} = \mathbb{R}^2$, we can consider $\varphi\left(\begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix}\right) = [x_{i1}, x_{i2}, x_{i1}^2, \sqrt{2}x_{i1}x_{i2}, x_{i2}^2]^\top$, such that the resulting function can depend on quadratic and interaction terms as well.
- This has an interpretation of simply adding more features and thereby also increasing the number of parameters - but can be solved in exactly the same way!

Beyond Linear Classifiers





- No linear classifier separates red from blue.
- Linear separation after mapping to a **higher dimensional feature space**:

$$\mathbb{R}^2 \ni \begin{pmatrix} x^{(1)} & x^{(2)} \end{pmatrix}^\top = x \;\mapsto\; \varphi(x) = \begin{pmatrix} x^{(1)} & x^{(2)} & x^{(1)}x^{(2)} \end{pmatrix}^\top \in \mathbb{R}^3$$

Kernel trick

- Suppose we have 2-dimensional inputs x, and we would like to introduce quadratic non-linearities,

$$\varphi(x) = \left(1, \sqrt{2}x^{(1)}, \sqrt{2}x^{(2)}, \sqrt{2}x^{(1)}x^{(2)}, \left(x^{(1)}\right)^2, \left(x^{(2)}\right)^2\right)^\top.$$

  Then

$$\begin{aligned}\varphi(x_i)^\top\varphi(x_j) &= 1 + 2x_i^{(1)}x_j^{(1)} + 2x_i^{(2)}x_j^{(2)} + 2x_i^{(1)}x_i^{(2)}x_j^{(1)}x_j^{(2)} \\ &\quad + \left(x_i^{(1)}\right)^2\left(x_j^{(1)}\right)^2 + \left(x_i^{(2)}\right)^2\left(x_j^{(2)}\right)^2 = (1 + x_i^\top x_j)^2\end{aligned}$$

- Since only inner products are needed, non-linear transform need not be computed explicitly - inner product between features can be a simple function (*kernel*) of x_i and x_j: $k(x_i, x_j) = \varphi(x_i)^\top\varphi(x_j) = (1 + x_i^\top x_j)^2$
- d-order interactions can be implemented by $k(x_i, x_j) = (1 + x_i^\top x_j)^d$ (*polynomial kernel*). Never need to compute explicit feature expansion of dimension $\binom{p+d}{d}$ where this inner product happens!
- kernel functions can correspond to inner products of features in infinite-dimensional spaces: kernel trick.

Kernel methods

- In a learning algorithm, if only inner products $x_i^\top x_j$ are explicitly used, rather than data items x_i, x_j directly, we can replace them with a kernel function $k(x_i, x_j) = \langle \varphi(x_i), \varphi(x_j) \rangle$, where $\varphi(x)$ could be *nonlinear, high- and potentially infinite-dimensional* features of the original data.
  - Kernel ridge regression
  - Kernel logistic regression
  - Kernel PCA, CCA, ICA
  - Kernel K-means

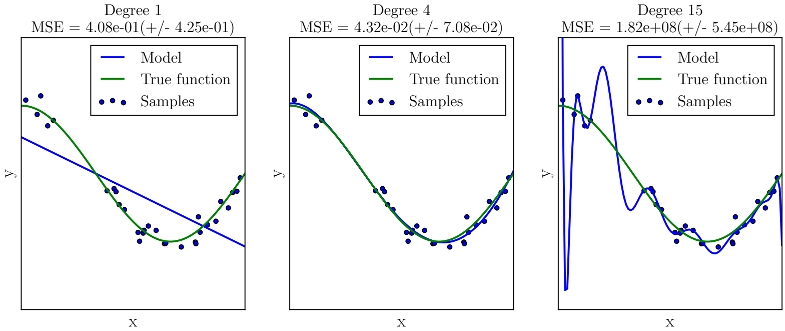


Figure: Underfitting and Overfitting

Training and Test Performance

- **Training error** is the empirical risk

$$\hat{R}_{\mathsf{tr}}(\hat{f}) = \frac{1}{n}\sum_{i=1}^{n} L(y_i, \hat{f}(x_i))$$

of the *learned function* $\hat{f}$. For example, for 0-1 loss in classification, this is the number of misclassified training examples **which were used in learning** $\hat{f}$. Note that

$$\mathbb{E}_{P_{XY}}\hat{R}_{\mathsf{tr}}(\hat{f}) \neq R(f).$$

- **Test error** is the empirical risk on **new, previously unseen** observations $\{\tilde{x}_i, \tilde{y}_i\}_{i=1}^{m}$

$$\hat{R}_{\mathsf{tst}}(\hat{f}) = \frac{1}{m}\sum_{i=1}^{m} L(\tilde{y}_i, \hat{f}(\tilde{x}_i))$$

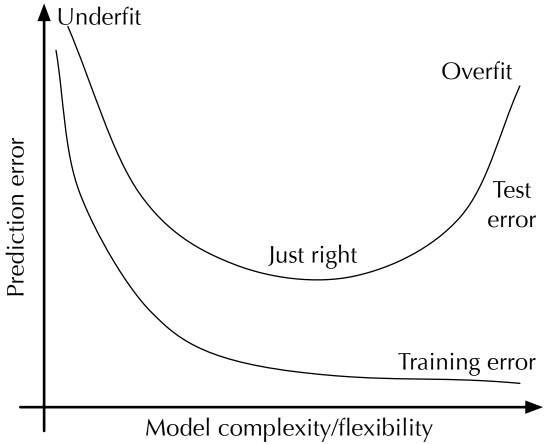
which were NOT used in learning f.

- Test error tells us how well the learned function *generalizes* to new data ($\mathbb{E}_{P_{XY}}\hat{R}_{\mathsf{tst}}(\hat{f}) \neq R(f)$) and is in general larger than the training error.

Generalization

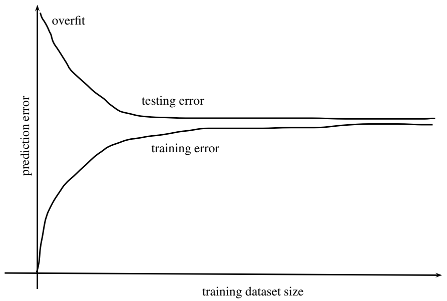
- Generalization ability: what is the out-of-sample error of learner f?
- training error $\neq$ testing error.
- We learn f by minimizing some variant of empirical risk $R^{\text{emp}}(f)$- what can we say about the true risk $R(f)$?
- Two important factors determining generalization ability:
  - Model complexity
  - Training data size
- To control overfitting, we need to reduce the model complexity to an appropriate level, i.e. *regularize learning*.

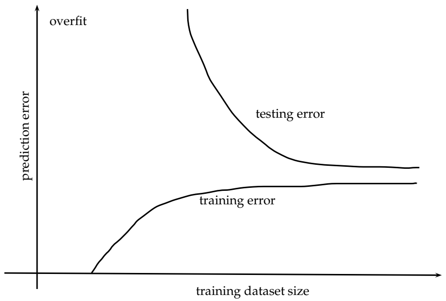
Learning Curves



The relationship between training and test error as a function of model complexity.

Learning Curves





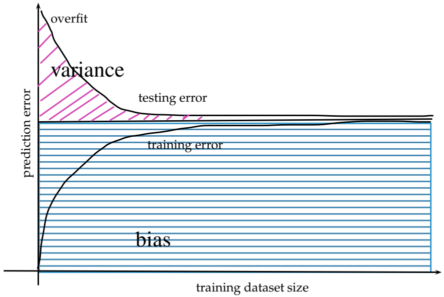
Fixed model complexity, varying dataset size.
Two models: one simple, one complex. Which is which?

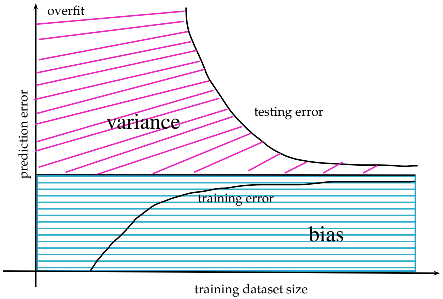
Bias-Variance Tradeoff

Where does the prediction error come from?

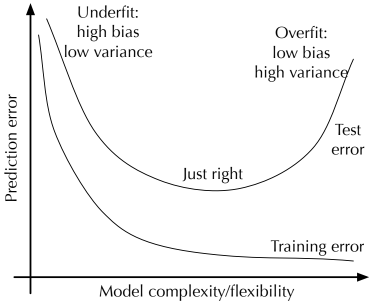
- *Noise*: Intrinsic difficulty of regression problem.
- *Bias*: How far away is the best learner in the model (average learner over all possible datasets) from the optimal one?
- *Variance*: How variable is our learning method if given different datasets?

Learning Curves





Building models to trade bias with variance



- Building a machine learning model involves trading between its bias and variance.
    - Bias reduction at the expense of a variance increase: building more complex models, e.g. adding nonlinear features and additional parameters, increasing the number of hidden units in neural nets, using decision trees with larger depth.
    - Variance reduction at the expense of a bias increase: increasing the regularization parameter, early stopping, using k-nearest neighbours with larger k.

Regularisation

- Flexible models for high-dimensional problems require many parameters.
- With many parameters, learners can easily overfit.
- *regularisation*: Limit flexibility of model to prevent overfitting.
- Add term *penalizing large values of parameters* θ.

$$\min_\theta \hat{R}(f_\theta) + \lambda \|\theta\|_\rho^\rho = \min_\theta \frac{1}{n} \sum_{i=1}^{n} L(y_i, f_\theta(x_i)) + \lambda \|\theta\|_\rho^\rho$$

  where $\rho \geq 1$, and $\|\theta\|_\rho = (\sum_{j=1}^{p} |\theta_j|^\rho)^{1/\rho}$ is the L_ρ norm of θ (also of interest when $\rho \in [0, 1)$, but is no longer a norm).
- Also known as *shrinkage* methods—parameters are shrunk towards 0.
- λ is a *tuning parameter* (or *hyperparameter*) and controls the amount of regularisation, and resulting complexity of the model.

Types of regularisation

- *Ridge regression / Tikhonov regularisation*: $\rho = 2$ (Euclidean norm)
- *LASSO*: $\rho = 1$ (Manhattan norm)
- *Sparsity-inducing* regularisation: $\rho \leq 1$ (nonconvex for $\rho < 1$)
- *Elastic net* regularisation: mixed L_1/L_2 penalty:

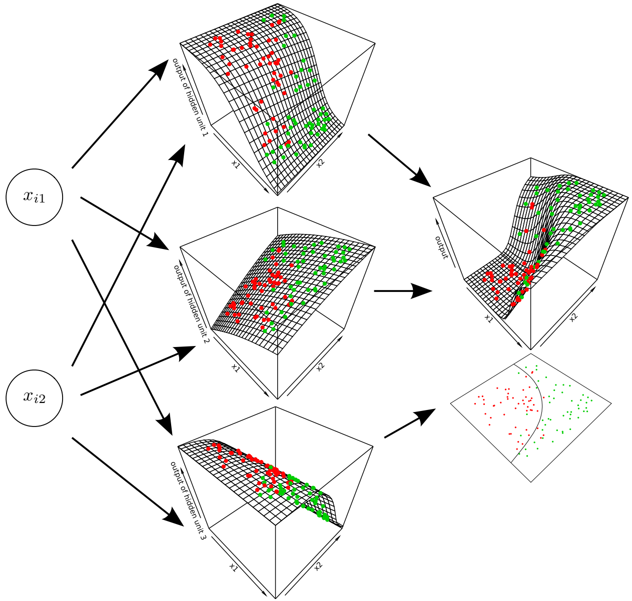
$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f_\theta(x_i)) + \lambda \left[(1-\alpha)\|\theta\|_2^2 + \alpha\|\theta\|_1\right]$$

- *Regularization in kernel methods*: directly penalise some notion of *smoothness* of function f, e.g. for $\mathcal{X} = \mathbb{R}$, the regularisation term can consist of the *Sobolev norm*

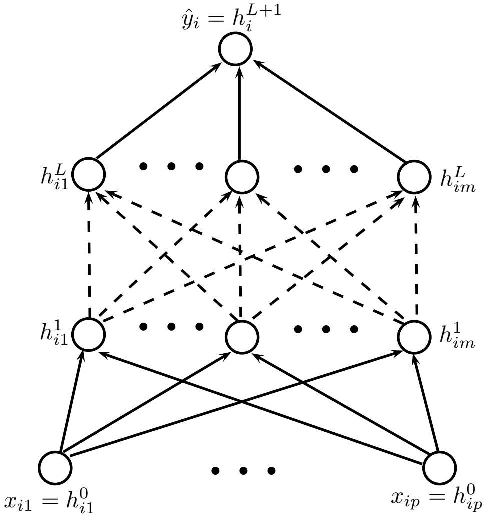
$$\|f\|_{W^1}^2 = \int_{-\infty}^{+\infty} f(x)^2 dx + \int_{-\infty}^{+\infty} f'(x)^2 dx, \tag{1}$$

which penalises functions with large derivative values.

Neural Networks and Adaptive Feature Maps



Deep Networks



$$h_i^{\ell+1} = \underline{s}\left(W^{\ell+1} h_i^\ell\right)$$

- $W^{\ell+1} = \left(w_{jk}^\ell\right)_{jk}$: weight matrix at the $(\ell+1)$-th layer, weight w_{jk}^ℓ on the edge between $h_{ik}^{\ell-1}$ and h_{ij}^ℓ
- $\underline{s}$: entrywise nonlinear transfer function, e.g. ReLU: $s(z) = \max(0, z)$

$$\hat{y}_i = \underline{s}\left(W^{L+1}\underline{s}\left(W^L\left(\cdots\underline{s}\left(W^1 x_i\right)\right)\right)\right)$$

- **Many** hidden layers can be used: they are usually thought of as forming a hierarchy from low-level to high-level features.

Dropout Regularization of Neural Networks

- Neural network with single layer of hidden units:
  - *Hidden unit activations*:

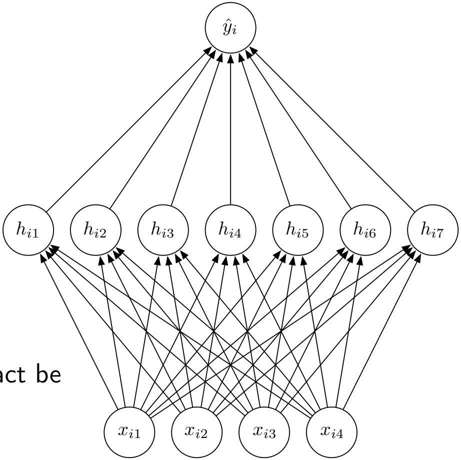
$$h_{ik} = s\left(b_k^h + \sum_{j=1}^{p} W_{jk}^h x_{ij}\right)$$

  - *Output probability*:

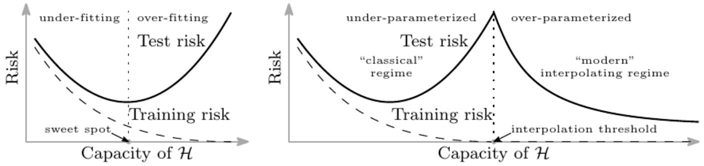
$$\hat{y}_i = s\left(b^o + \sum_{k=1}^{m} W_k^o h_{ik}\right)$$

- Large, overfitted networks often have co-adapted hidden units.
- What each hidden unit learns may in fact be useless, e.g. predicting the negation of predictions from other units.
- Can prevent co-adaptation by randomly *dropping out* units from network.

[Hinton et al (2012)]



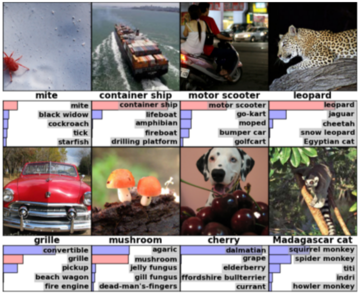
Modern “Interpolating” Regime



- **Left:** The classical U-shaped risk curve arising from the bias-variance trade-off.
- **Right:** The double descent risk curve, which incorporates the U-shaped risk curve (i.e., the classical regime) together with the observed behavior from using high capacity function classes (i.e., the modern interpolating regime). The predictors to the right of the interpolation threshold have zero training risk.

Belkin et al, Reconciling modern machine learning practice and the bias-variance trade-off

Statistical Tools for Machine Learning

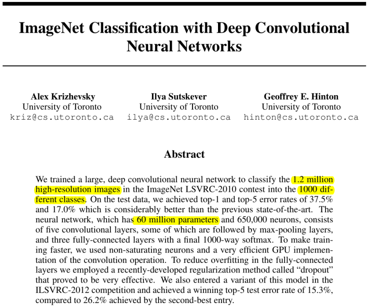


[Krizhevsky, Sutskever & Hinton, 2012]

The field of machine learning has been driven by the exponential growth in dataset sizes and computational resources, allowing to tackle difficult inference problems, which are characterized by:

- high dimensionality,
- multivariate interaction,
- complex patterns exhibiting various forms of nonlinearity and nonstationarity,
- little prior knowledge.

Statistical Tools for Machine Learning



ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky University of Toronto kriz@cs.utoronto.ca
Ilya Sutskever University of Toronto ilya@cs.utoronto.ca
Geoffrey E. Hinton University of Toronto hinton@cs.utoronto.ca

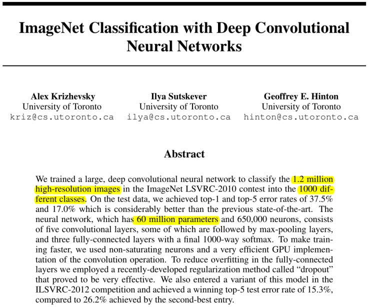
Abstract

We trained a large, deep convolutional neural network to classify the 1.2 million high-resolution images in the ImageNet LSVRC-2010 contest into the 1000 different classes. On the test data, we achieved top-1 and top-5 error rates of 37.5% and 17.0% which is considerably better than the previous state-of-the-art. The neural network, which has 60 million parameters and 650,000 neurons, consists of five convolutional layers, some of which are followed by max-pooling layers, and three fully-connected layers with a final 1000-way softmax. To make training faster, we used non-saturating neurons and a very efficient GPU implementation of the convolution operation. To reduce overfitting in the fully-connected layers we employed a recently-developed regularization method called "dropout" that proved to be very effective. We also entered a variant of this model in the ILSVRC-2012 competition and achieved a winning top-5 test error rate of 15.3%, compared to 26.2% achieved by the second-best entry.

The field of machine learning has been driven by the exponential growth in dataset sizes and computational resources, allowing to tackle difficult inference problems, which are characterized by:

- high dimensionality,
- multivariate interaction,
- complex patterns exhibiting various forms of nonlinearity and nonstationarity,
- little prior knowledge.

Statistical Tools for Machine Learning



ImageNet Classification with Deep Convolutional Neural Networks

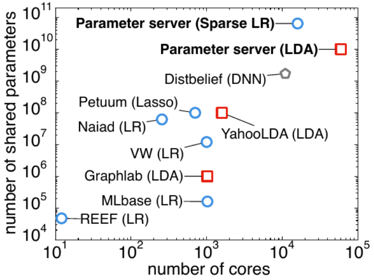
Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca

Abstract

We trained a large, deep convolutional neural network to classify the 1.2 million high-resolution images in the ImageNet LSVRC-2010 contest into the 1000 different classes. On the test data, we achieved top-1 and top-5 error rates of 37.5% and 17.0% which is considerably better than the previous state-of-the-art. The neural network, which has 60 million parameters and 650,000 neurons, consists of five convolutional layers, some of which are followed by max-pooling layers, and three fully-connected layers with a final 1000-way softmax. To make training faster, we used non-saturating neurons and a very efficient GPU implementation of the convolution operation. To reduce overfitting in the fully-connected layers we employed a recently-developed regularization method called "dropout" that proved to be very effective. We also entered a variant of this model in the ILSVRC-2012 competition and achieved a winning top-5 test error rate of 15.3%, compared to 26.2% achieved by the second-best entry.



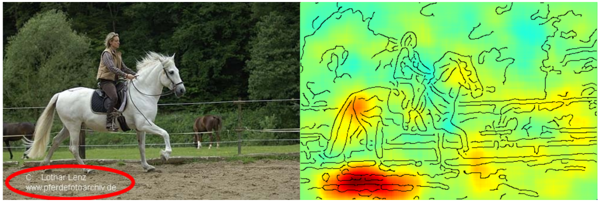
[Mu Li et al, 2014]

The field of machine learning has been driven by the exponential growth in dataset sizes and computational resources, allowing to tackle difficult inference problems, which are characterized by:

- high dimensionality,
- multivariate interaction,
- complex patterns exhibiting various forms of nonlinearity and nonstationarity,
- little prior knowledge.

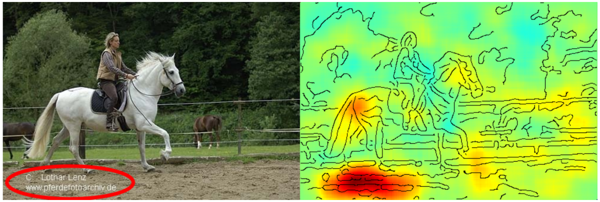
The use of complex models with massive amounts of parameters, even if they are unidentifiable and uninterpretable.

Uncertainty Calibration and Brittleness

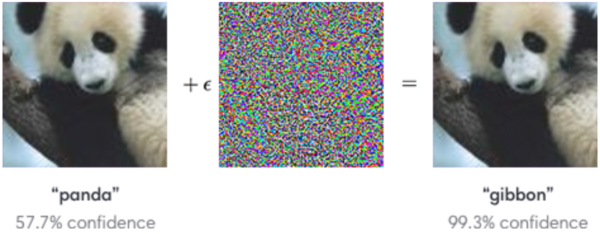


[Lapuschkin et al, 2016]

Uncertainty Calibration and Brittleness

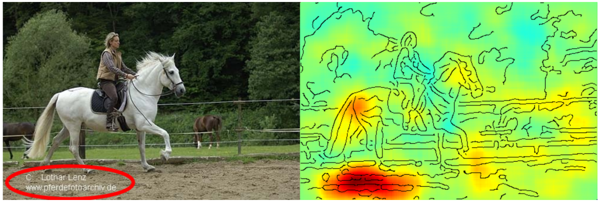


[Lapuschkin et al, 2016]

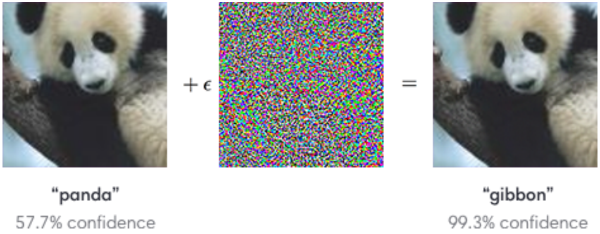


[Goodfellow et al, 2015]

Uncertainty Calibration and Brittleness



[Lapuschkin et al, 2016]

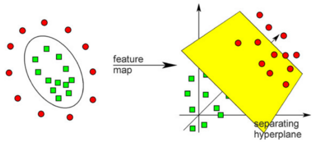


[Goodfellow et al, 2015]

Need for (scalable) statistical tools for model criticism and interpretability.

Kernel Trick and Kernel Mean Trick

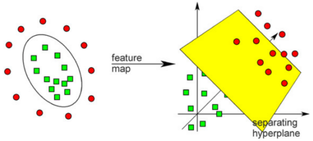
- implicit feature map $x \mapsto k(\cdot, x) \in \mathcal{H}_k$
  replaces $x \mapsto [\phi_1(x), \dots, \phi_s(x)] \in \mathbb{R}^s$
- $\phi(x)^\top \phi(y) = k(x, y) = \langle k(\cdot, x), k(\cdot, y) \rangle_{\mathcal{H}_k}$
  inner products readily available
  - nonlinear decision boundaries, nonlinear regression functions, learning on non-Euclidean/structured data



[Cortes & Vapnik, 1995; Schölkopf & Smola, 2001]

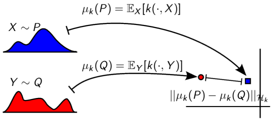
Kernel Trick and Kernel Mean Trick

- implicit feature map $x \mapsto k(\cdot, x) \in \mathcal{H}_k$
  replaces $x \mapsto [\phi_1(x), \ldots, \phi_s(x)] \in \mathbb{R}^s$
- $\phi(x)^\top \phi(y) = k(x, y) = \langle k(\cdot, x), k(\cdot, y) \rangle_{\mathcal{H}_k}$
  inner products readily available
  - nonlinear decision boundaries, nonlinear regression functions, learning on non-Euclidean/structured data



[Cortes & Vapnik, 1995; Schölkopf & Smola, 2001]

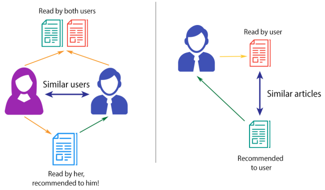
- **RKHS embedding**: implicit feature mean
  [Smola et al, 2007; Sriperumbudur et al, 2010]
  $P \mapsto \mu_k(P) = \mathbb{E}_{X \sim P} k(\cdot, X) \in \mathcal{H}_k$
  replaces $P \mapsto [\mathbb{E}\phi_1(X), \ldots, \mathbb{E}\phi_s(X)] \in \mathbb{R}^s$
- $\langle \mu_k(P), \mu_k(Q) \rangle_{\mathcal{H}_k} = \mathbb{E}_{X \sim P, Y \sim Q} k(X, Y)$
  inner products easy to estimate
  - nonparametric two-sample, independence, conditional independence, interaction testing, learning on distributions, model criticism and interpretability



$\mu_k(P) = \mathbb{E}_X[k(\cdot, X)]$
$X \sim P$
$\mu_k(Q) = \mathbb{E}_Y[k(\cdot, Y)]$
$Y \sim Q$
$\|\mu_k(P) - \mu_k(Q)\|_{\mathcal{H}_k}$

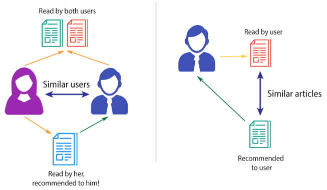
[Gretton et al, 2005; Gretton et al, 2006; Fukumizu et al, 2007; Muandet et al, 2012; DS et al, 2013; Szabo et al, 2015; Kim et al, 2016]

Causality in Machine Learning

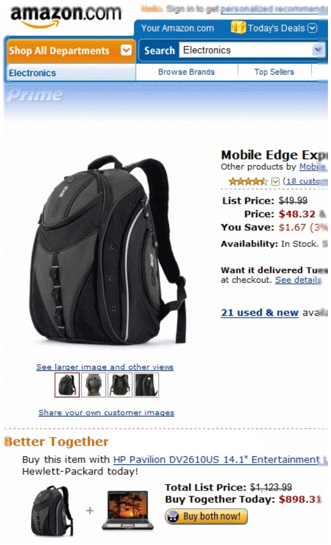


CUSTOMERS WHO BOUGHT THIS ITEM:
ALSO BOUGHT:
piccolo

Causality in Machine Learning



CUSTOMERS WHO BOUGHT THIS ITEM:
ALSO BOUGHT:
piccolo



Mobile Edge Exp
Other products by Mobi
(18 custo
List Price: ~~$49.99~~
Price: $48.32
You Save: $1.67 (3
Availability: In Stock.

Want it delivered Tue
at checkout. See details

21 used & new avai

See larger image and other views

Share your own customer images

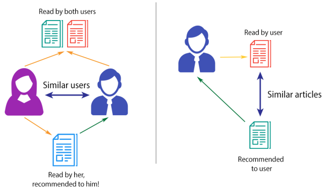
Better Together

Buy this item with HP Pavilion DV2610US 14.1" Entertainment
Hewlett-Packard today!

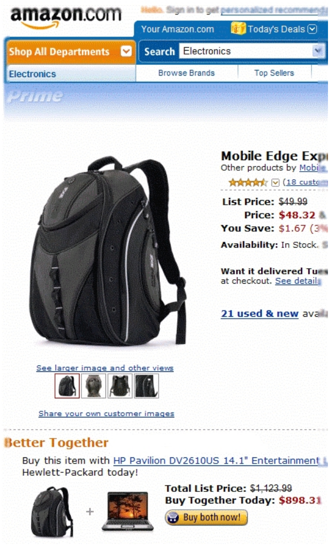
Total List Price: ~~$1,123.99~~
Buy Together Today: $898.31
Buy both now!

[example by Bernhard Schölkopf]

Causality in Machine Learning



CUSTOMERS WHO BOUGHT THIS ITEM:
ALSO BOUGHT:
piccolo

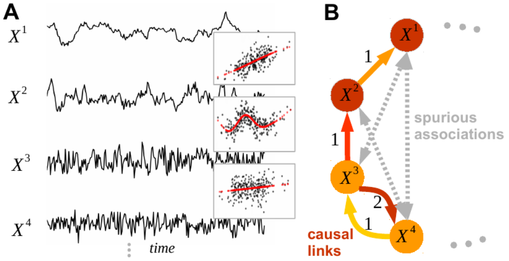


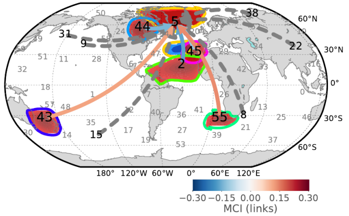
Mobile Edge Exp
Other products by Mobi
(18 custo
List Price: ~~$49.99~~
Price: $48.32
You Save: $1.67 (3
Availability: In Stock.
Want it delivered Tue
at checkout. See detail
21 used & new ava
See larger image and other views
Share your own customer images
Better Together
Buy this item with HP Pavilion DV2610US 14.1" Entertainment
Hewlett-Packard today!
Total List Price: ~~$1,123.99~~
Buy Together Today: $898.31
Buy both now!

[example by Bernhard Schölkopf]

Disentangling Correlation from Causation in Machine Learning?

Causal Discovery from Time Series Data





[J. Runge, S. Flaxman and DS, Detecting causal associations in large nonlinear time series datasets, ArXiv e-prints:1702.07007, 2017.]

Learning on Aggregates

- *Supervised learning*: obtaining inputs has a lower cost than obtaining outputs/labels, hence we build a (predictive) functional relationship or a conditional probabilistic model of outputs given inputs.
- *Semisupervised learning*: because of the lower cost, there is much more unlabelled than labelled inputs.
- *Weakly supervised learning on aggregates*: because of the lower cost, inputs are at a much higher resolution than outputs.

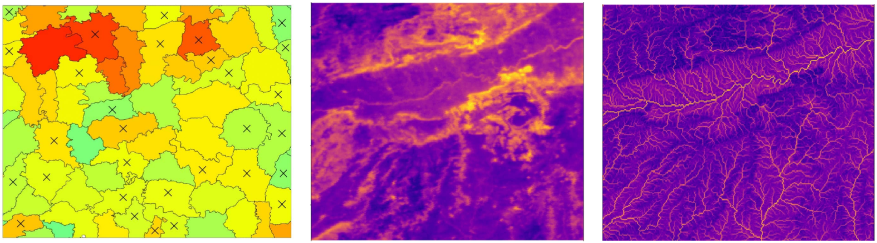


Figure: **left**: Malaria incidences reported per administrative unit; **centre**: land surface temperature at night; **centre**: topographic wetness index

Fine-scale modelling of disease

Suppose you have a country with n regions and data on:

- number of malaria incidences per region (low resolution)
- many covariates per region (high resolution)

Goal: Predict malaria incidences at a higher resolution, given low resolution label data and high definition covariate data.

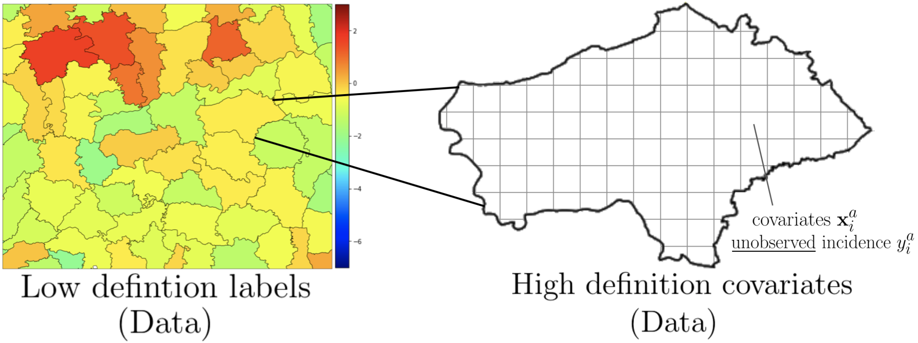

2
0
−2
−4
−6
covariates $\mathbf{x}_i^a$
unobserved incidence y_i^a
Low defintion labels (Data)
High definition covariates (Data)

Figure: Log incidence rate of malaria

[Law et al, Variational Learning on Aggregate Outputs with Gaussian Processes, in NeurIPS, 2018.]

Fine-scale modelling of disease

Suppose you have a country with n regions and data on:

- number of malaria incidences per region (low resolution)
- many covariates per region (high resolution)

Goal: Predict malaria incidences at a <u>higher</u> resolution, given low resolution label data and high definition covariate data.

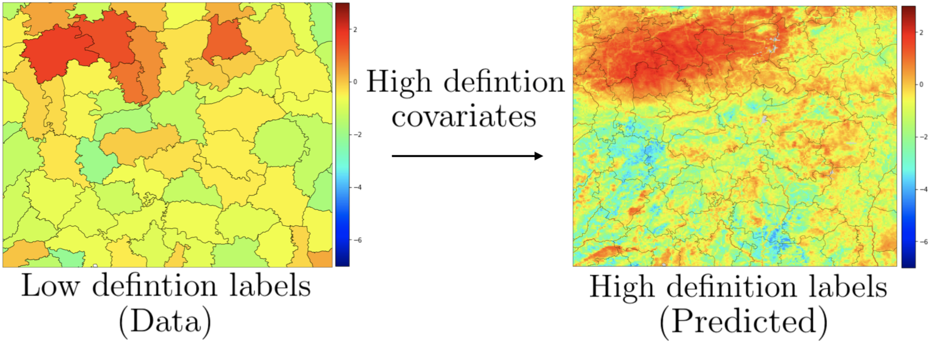


Figure: Log incidence rate of malaria

[Law et al, Variational Learning on Aggregate Outputs with Gaussian Processes, in NeurIPS, 2018.]

Summary

- Statistical machine learning is a vast framework for understanding data, making inferences and predictions based on data in order to guide informed decisions.
- Can be supervised (learning from examples) or unsupervised (extracting patterns)
- Model complexity and generalization ability: trading off bias and variance
- Increasing confluence between statistical modelling and machine learning: interpretability, causality, robustness, fairness,...
- **Further resources:**
  - Oxford course materials: Statistical Machine Learning, Advanced Topics in Statistical Machine Learning
  - Machine Learning Summer Schools, videolectures.net.
  - Conferences: NeurIPS, ICML, UAI, AISTATS.