Multivariate analysis techniques and machine learning algorithms

Stefan Ohm

Fermi Summer School, Lewes, 03.06.2017

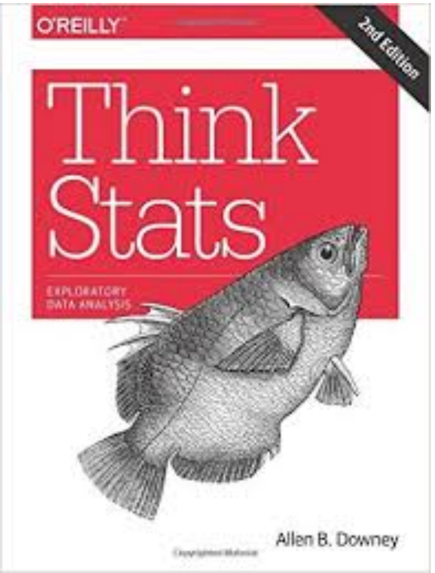
Outline

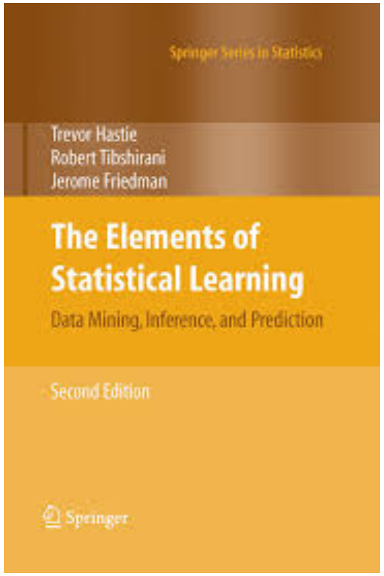
- What this lecture is *not*
  - a math and statistics class, a programming course
- What this lecture tries to be
  - an “experimental astroparticle physicists approach to data”
- PART 1 (~15 minutes)
  - Data Science: the bigger picture, or, what we are all facing sooner or later
  - Machine learning: The basic basics
- PART 2 (~35 minutes)
  - Multivariate analysis (MVA) techniques: decision trees, neural networks, deep learning
  - Real-life examples: Boosted Decision Trees in γ-ray astronomy
- If anything is unclear just ask

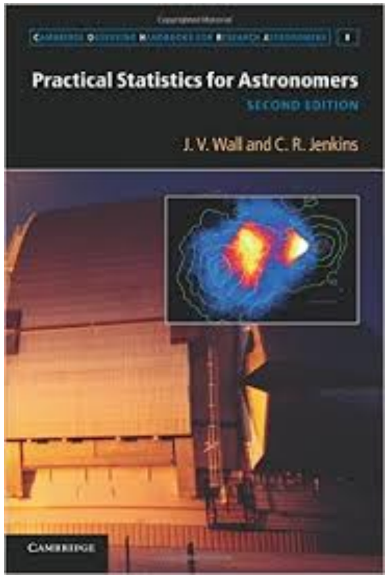
Literature and Sources

> Many of the slides are inspired by these books and lecture notes

- Think Stats: Exploratory Data Analysis in Python, A.B. Downey, Green Tea Press, 2014
- The Elements of Statistical Learning: Data Mining, Interference, and Prediction, Hasti, Tibshirani, Friedman, 2009, Springer Series in Statistics
- Data Science from Scratch: Joel Grus, O'Reilly, 2015
- Practical Statistics for Astronomers, Jasper Wall (http://www.astro.ubc.ca/people/jvw/ASTROSTATS/)
- other references given throughout the presentation and at the end




Springer Series in Statistics
Trevor Hastie
Robert Tibshirani
Jerome Friedman
The Elements of Statistical Learning
Data Mining, Inference, and Prediction
Second Edition
Springer

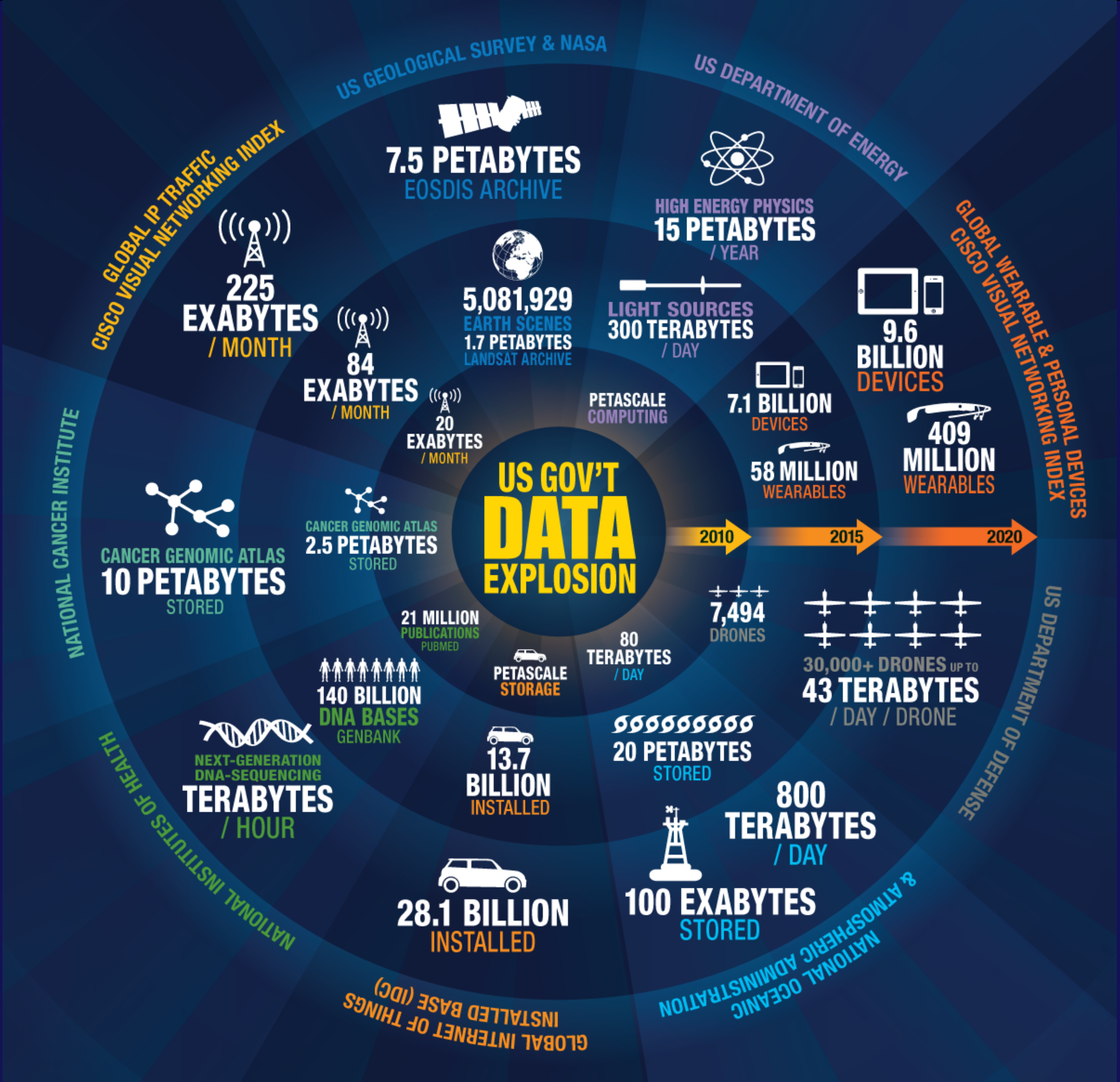

Practical Statistics for Astronomers
SECOND EDITION
J. V. Wall and C. R. Jenkins
CAMBRIDGE

Data Science and the basics about Machine Learning

DATA EXPLOSION

- in science
- in industry
- in every-day life

US GOV'T DATA EXPLOSION



US GEOLOGICAL SURVEY & NASA

7.5 PETABYTES
EOSDIS ARCHIVE

5,081,929
EARTH SCENES
1.7 PETABYTES
LANDSAT ARCHIVE

US DEPARTMENT OF ENERGY

HIGH ENERGY PHYSICS
15 PETABYTES
/ YEAR

LIGHT SOURCES
300 TERABYTES
/ DAY

PETASCALE
COMPUTING

GLOBAL WEARABLE & PERSONAL DEVICES
CISCO VISUAL NETWORKING INDEX

9.6
BILLION
DEVICES

409
MILLION
WEARABLES

7.1 BILLION
DEVICES

58 MILLION
WEARABLES

US DEPARTMENT OF DEFENSE

7,494
DRONES

30,000+ DRONES UP TO
43 TERABYTES
/ DAY / DRONE

80
TERABYTES
/ DAY

NATIONAL OCEANIC & ATMOSPHERIC ADMINISTRATION

20 PETABYTES
STORED

800
TERABYTES
/ DAY

100 EXABYTES
STORED

GLOBAL INTERNET OF THINGS INSTALLED BASE (IDC)

PETASCALE
STORAGE

13.7
BILLION
INSTALLED

28.1 BILLION
INSTALLED

NATIONAL INSTITUTES OF HEALTH

21 MILLION
PUBLICATIONS
PUBMED

140 BILLION
DNA BASES
GENBANK

NEXT-GENERATION
DNA-SEQUENCING
TERABYTES
/ HOUR

NATIONAL CANCER INSTITUTE

CANCER GENOMIC ATLAS
10 PETABYTES
STORED

CANCER GENOMIC ATLAS
2.5 PETABYTES
STORED

GLOBAL IP TRAFFIC
CISCO VISUAL NETWORKING INDEX

225
EXABYTES
/ MONTH

84
EXABYTES
/ MONTH

20
EXABYTES
/ MONTH

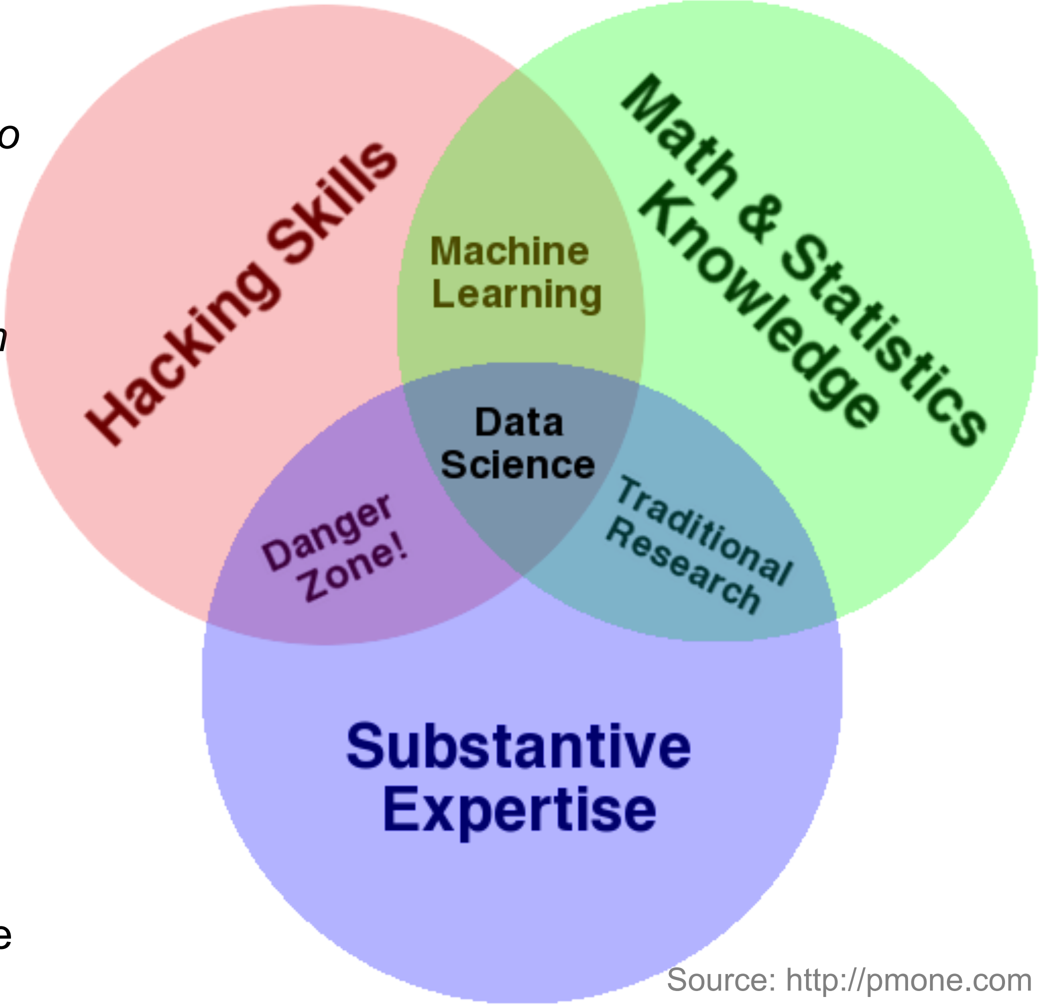
SCALITY www.scality.com @scality

Sources: IDC, Cisco, U.S. DOE, USGS, NIH, Earth Imaging Journal, Wired, Washington Times, Information Week, Health IT Analytics.

Big Data & Data Science

- According to Wikipedia
  - *“Data Science is an interdisciplinary field about processes and systems to extract knowledge or insights from large volumes of data in various forms, either structured or unstructured, which is a continuation of some of the data analysis fields such as data mining and predictive analytics, as well as knowledge discovery in databases”*

- Venn Diagram
  - Physicists and astronomers are the definition of Data Scientists
  - Fulfill all requirements and are developers of widely applicable code
  - Beware of the Danger Zone!



Source: http://pmone.com

How does machine learning fit in?

- Data Science covers techniques from different fields
  - signal processing, probability models, **machine learning**, statistical learning, data mining, database, data engineering, pattern recognition and learning, visualization, predictive analytics, uncertainty modeling, […], high performance computing
- Difficult to talk about machine learning without talking about data
- A few examples
  - *Facebook* uses hometown and current location to identify global migration patterns
  - *Target* tracks purchases and interactions to predict which of its customers is pregnant
  - *2012 US election* – Obama employed hundreds of data scientists to identify potential voters
- Data Science for social good (http://dssg.uchicago.edu)
  - Improve government
  - Help homeless people
  - Improve health care

Data analysis

> Data retrieval

- from experiments
- generate your own data
- from internet or an Application Programming Interface (API)

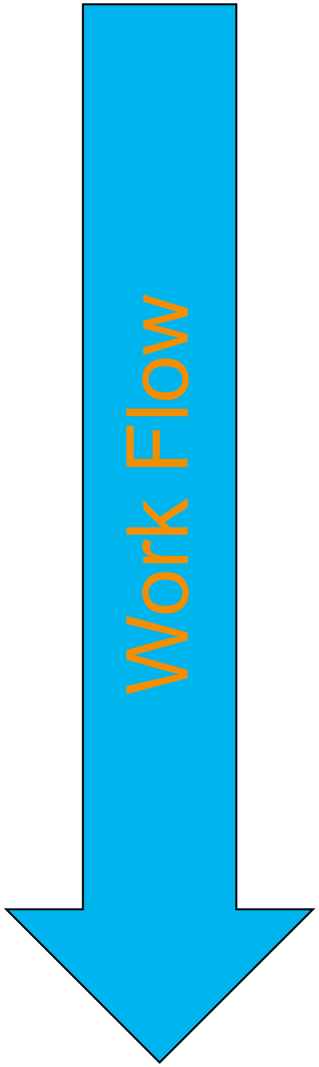
> Data preparation

- calibrate raw data
- remove outliers or noise

> Data pre-processing

- Identify parameters with valuable information in calibrated data (signal/background classification, pattern recognition)
- First-level selection cuts and data reduction (“outliers”)

> Data Mining (Machine Learning with multivariate analysis techniques)



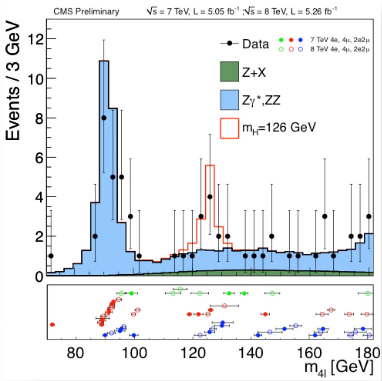
Work Flow

→ *What are the tools to do that?*

The toolkits

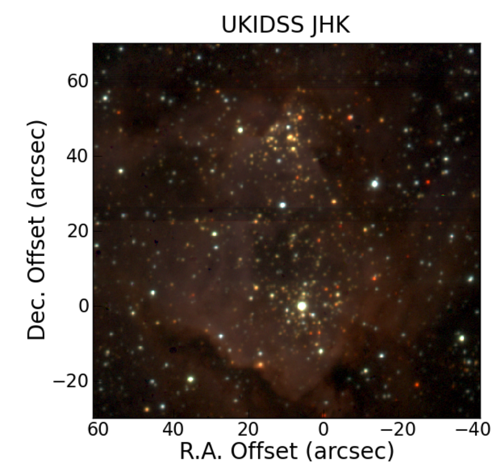
ROOT

- Mostly used in particle physics
- Adapted in astronomy and astroparticle physics
- C++, object-oriented
- Higgs-discovery plots produced with ROOT



IDL, MATLAB, R, ds9, …

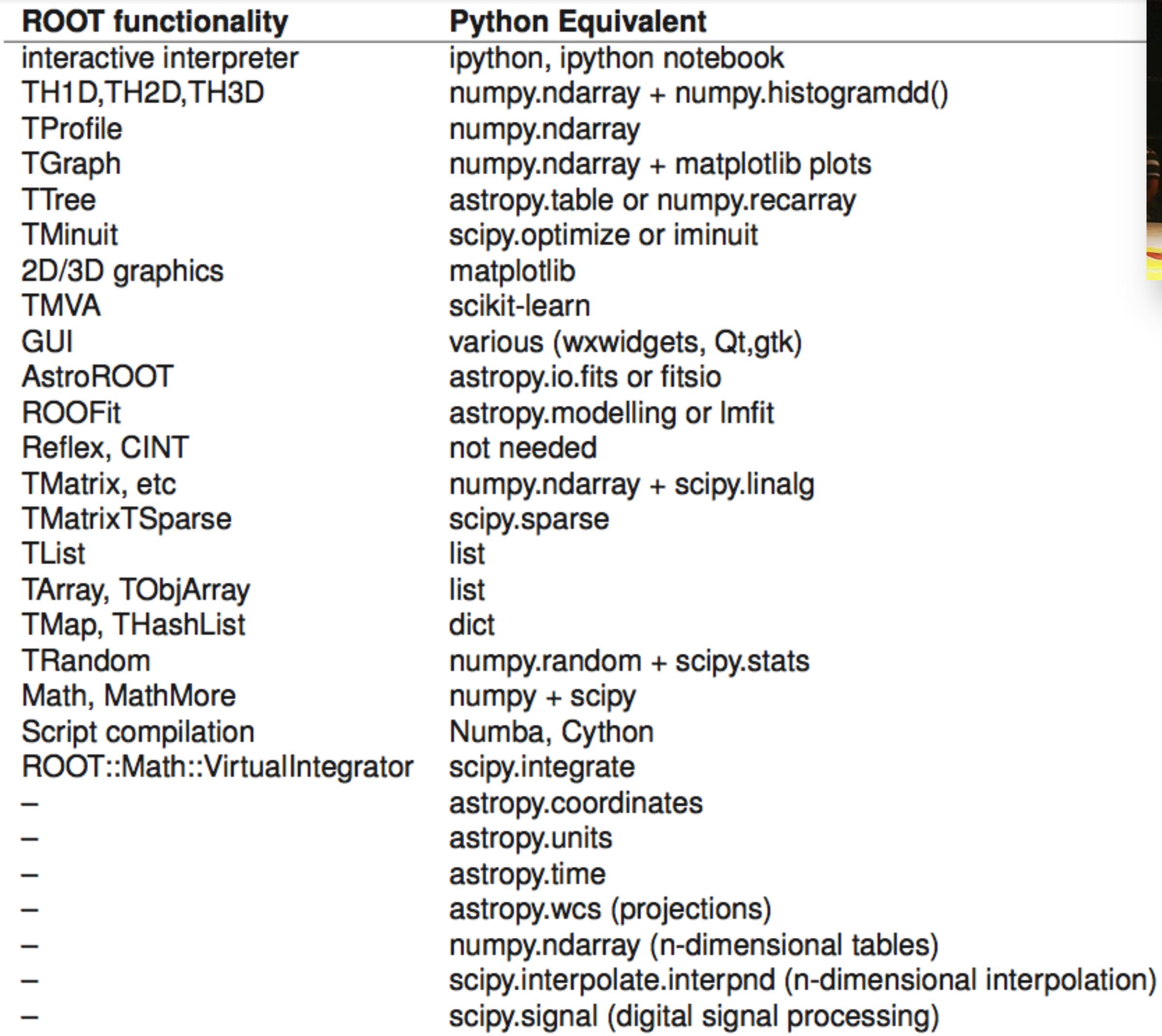
- many statistics features and numerical methods
- standard in different fields for many years, now being replaced by python



Python

- the new standard in industry and science

ROOT vs. python



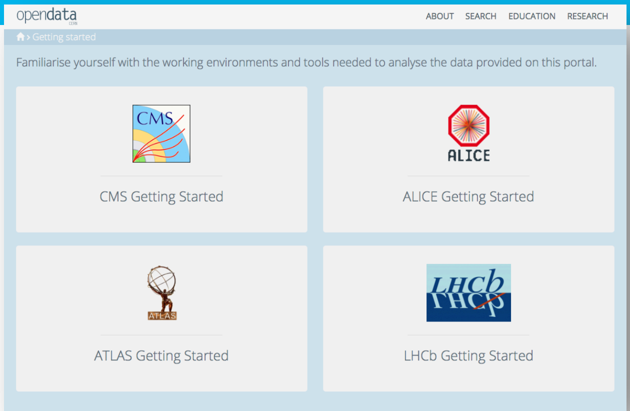
ROOT functionality	Python Equivalent
interactive interpreter	ipython, ipython notebook
TH1D,TH2D,TH3D	numpy.ndarray + numpy.histogramdd()
TProfile	numpy.ndarray
TGraph	numpy.ndarray + matplotlib plots
TTree	astropy.table or numpy.recarray
TMinuit	scipy.optimize or iminuit
2D/3D graphics	matplotlib
TMVA	scikit-learn
GUI	various (wxwidgets, Qt,gtk)
AstroROOT	astropy.io.fits or fitsio
ROOFit	astropy.modelling or lmfit
Reflex, CINT	not needed
TMatrix, etc	numpy.ndarray + scipy.linalg
TMatrixTSparse	scipy.sparse
TList	list
TArray, TObjArray	list
TMap, THashList	dict
TRandom	numpy.random + scipy.stats
Math, MathMore	numpy + scipy
Script compilation	Numba, Cython
ROOT::Math::VirtualIntegrator	scipy.integrate
–	astropy.coordinates
–	astropy.units
–	astropy.time
–	astropy.wcs (projections)
–	numpy.ndarray (n-dimensional tables)
–	scipy.interpolate.interpnd (n-dimensional interpolation)
–	scipy.signal (digital signal processing)

python

- provides more functionality than ROOT
- vastly more external developers
- shared by many communities (scientific and non-scientific)
- better/cleaner design
- quicker and easier to get things done
- more lightweight

Data retrieval

- Open access is the way to go
  - Astronomical observatories after proprietary time
  - Provided via webpages and online archives
  - Mostly in FITS format
  - Particle physics community is joining
  - Often via Virtual Machines
    - (Almost) no fiddling around with software versions, installations, external packages, etc.



opendata ABOUT SEARCH EDUCATION RESEARCH

> Getting started

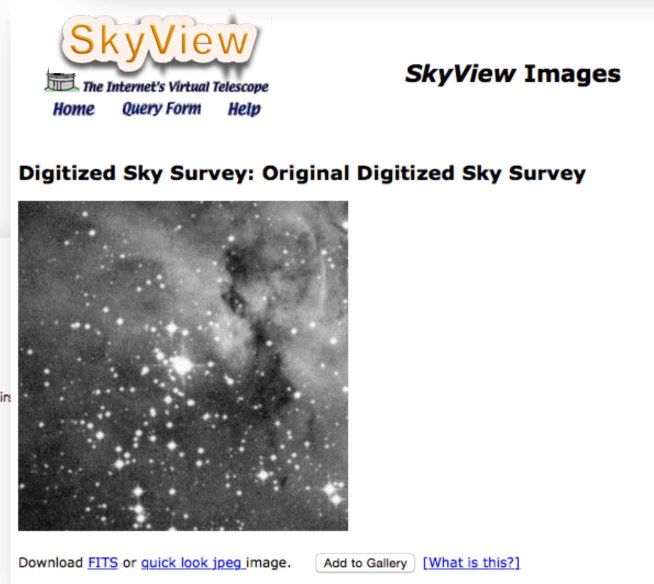
Familiarise yourself with the working environments and tools needed to analyse the data provided on this portal.

CMS Getting Started

ALICE Getting Started

ATLAS Getting Started

LHCb Getting Started



SkyView

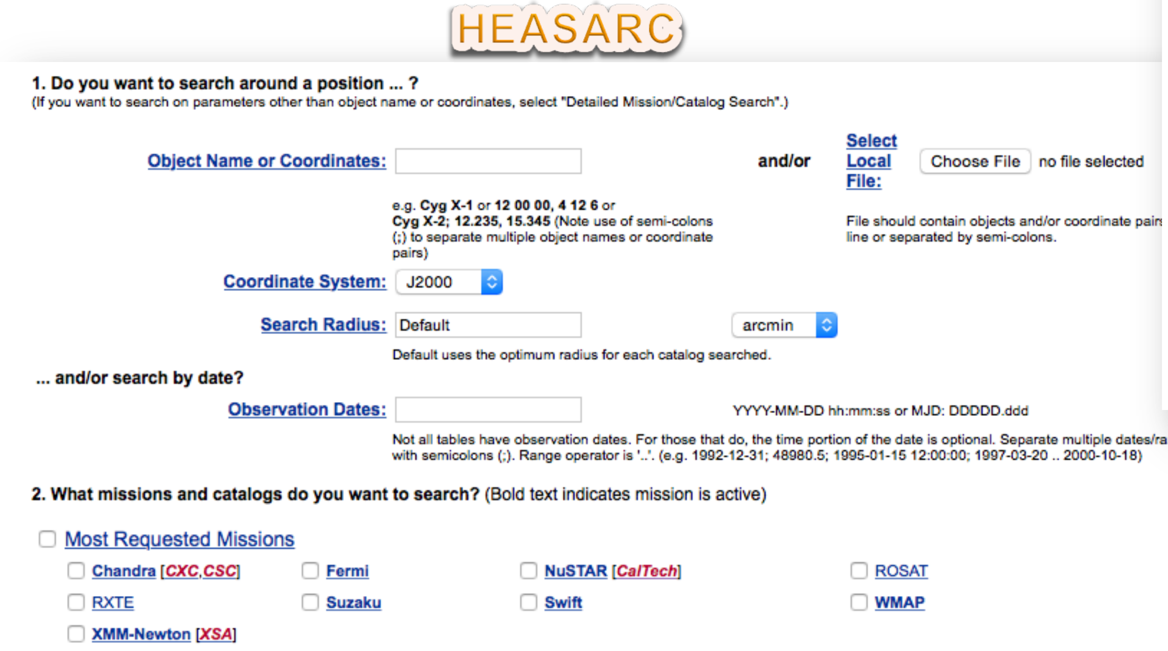
The Internet's Virtual Telescope

Home Query Form Help

SkyView Images

Digitized Sky Survey: Original Digitized Sky Survey

Download FITS or quick look jpeg image. Add to Gallery [What is this?]



HEASARC

1. Do you want to search around a position ... ?
(If you want to search on parameters other than object name or coordinates, select "Detailed Mission/Catalog Search".)

Object Name or Coordinates:		and/or	Select Local File: Choose File no file selected
	e.g. **Cyg X-1** or **12 00 00, 4 12 6** or **Cyg X-2; 12.235, 15.345** (Note use of semi-colons (;) to separate multiple object names or coordinate pairs)		File should contain objects and/or coordinate pair line or separated by semi-colons.
Coordinate System:	J2000		
Search Radius:	Default	arcmin	
	Default uses the optimum radius for each catalog searched.		

... and/or search by date?

Observation Dates:		YYYY-MM-DD hh:mm:ss or MJD: DDDDD.ddd

Not all tables have observation dates. For those that do, the time portion of the date is optional. Separate multiple dates/ranges with semicolons (;). Range operator is '..'. (e.g. 1992-12-31; 48980.5; 1995-01-15 12:00:00; 1997-03-20 .. 2000-10-18)

2. What missions and catalogs do you want to search? (Bold text indicates mission is active)

□ Most Requested Missions

□ **Chandra** [*CXC*,*CSC*]
□ RXTE
□ **XMM-Newton** [*XSA*]
□ **Fermi**
□ **Suzaku**
□ **NuSTAR** [*CalTech*]
□ **Swift**
□ ROSAT
□ **WMAP**

Data retrieval

> Generate your own data

- very useful in many circumstances
- using random number generators
- large-scale Monte Carlo simulations

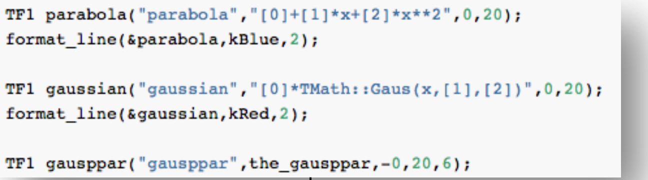
> Monte Carlo simulations

- most of you know those
- for optimization, instrument characterization, …
- drawing from probability distributions

> Application Programming Interfaces

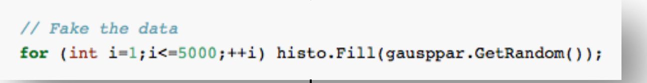
- Many webpages provide APIs
- Interface to receive data in structured format (XML, JSON)
- Examples: Amazon, Ebay, Facebook, Geopy, Google Maps, Last.fm, Rotten Tomatoes, Twitter

Generate function



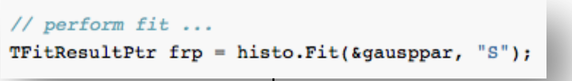
```
TF1 parabola("parabola","[0]+[1]*x+[2]*x**2",0,20);
format_line(&parabola,kBlue,2);

TF1 gaussian("gaussian","[0]*TMath::Gaus(x,[1],[2])",0,20);
format_line(&gaussian,kRed,2);

TF1 gausppar("gausppar",the_gausppar,-0,20,6);
```

Fake data



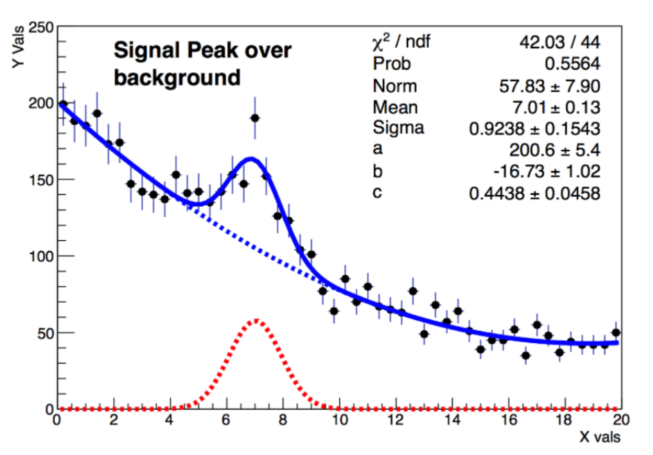
```
// Fake the data
for (int i=1;i<=5000;++i) histo.Fill(gausppar.GetRandom());
```

Fit fake data



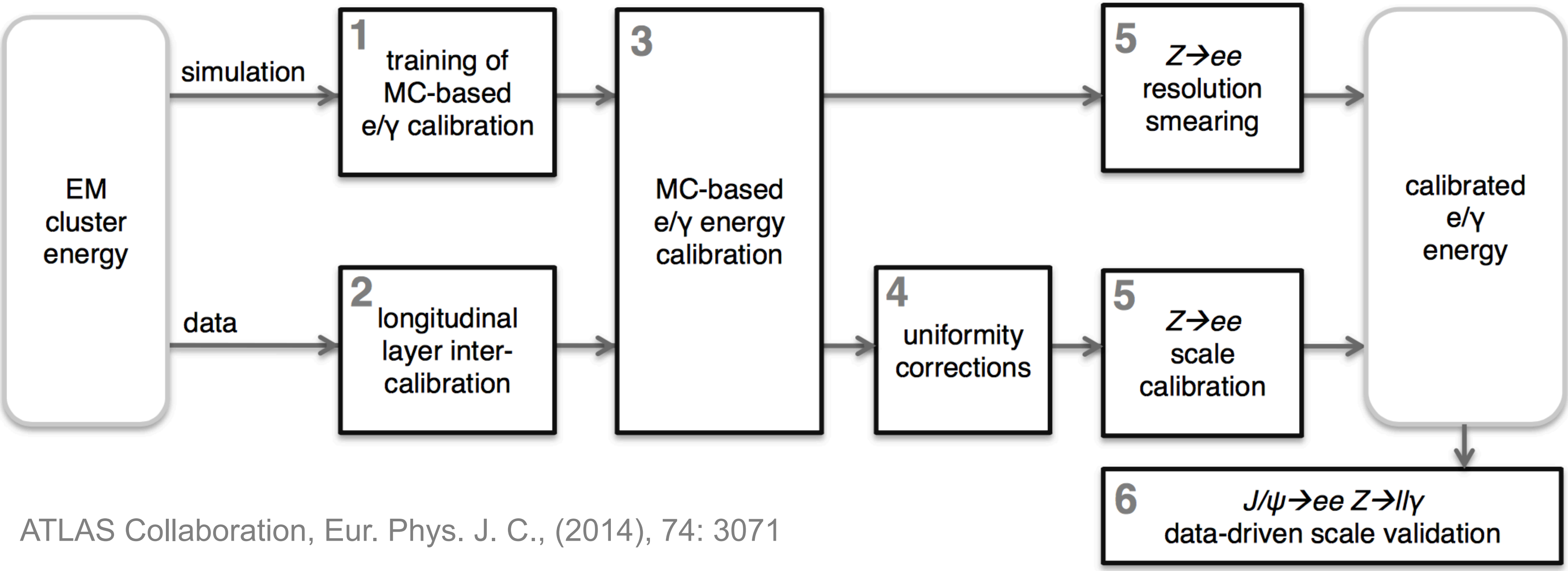
```
// perform fit ...
TFitResultPtr frp = histo.Fit(&gausppar, "S");
```

Draw everything



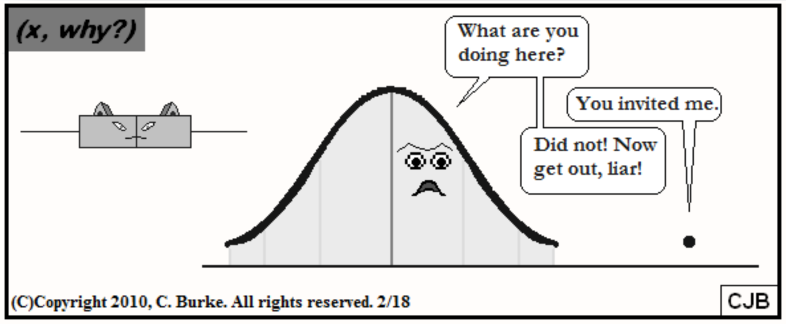
Data preparation

- The data you want to analyze is calibrated
  - Congratulations, someone has done the job for you
  - Don’t forget to check also calibrated data
- The data you want to analyze is not calibrated
  - Lets roll up the sleeves and calibrate the data
  - Illustration of how important calibration (and simulations) are (ATLAS EM calorimeter)


EM cluster energy
simulation
data
1 training of MC-based e/γ calibration
2 longitudinal layer inter-calibration
3 MC-based e/γ energy calibration
4 uniformity corrections
5 Z→ee resolution smearing
5 Z→ee scale calibration
calibrated e/γ energy
6 J/ψ→ee Z→llγ data-driven scale validation

ATLAS Collaboration, Eur. Phys. J. C., (2014), 74: 3071

Data pre-processing


(x, why?)
What are you doing here?
You invited me.
Did not! Now get out, liar!
(C)Copyright 2010, C. Burke. All rights reserved. 2/18
CJB

- Identify outliers during data preparation
- Outliers
  1. Occur during data collection, i.e. human errors
  2. Malicious act (e.g. in questionnaires)
  3. Noise in data (temporal, spatial, etc.)
  4. Incorrect assumptions when looking at data or building model
- Importance
  - 1 & 2 very important in social sciences, medical studies
  - 3 & 4 very important in fields where data is collected with ‘instruments’ (e.g. physics, astronomy, geo-sciences)
- If kept
  - 1 & 2 may influence statistical analyses and outcome of statistical test (e.g. correlations)
  - 3 & 4 can fake signal, wrong understanding of background

Where are we now?

> We retrieved some data

- from an instrument or from elsewhere

> We calibrated the data

- or got calibrated data

> We inspected the data

- identified outliers and removed them,
- or know where they come from and include them in our model

> Building a model and interpreting the data

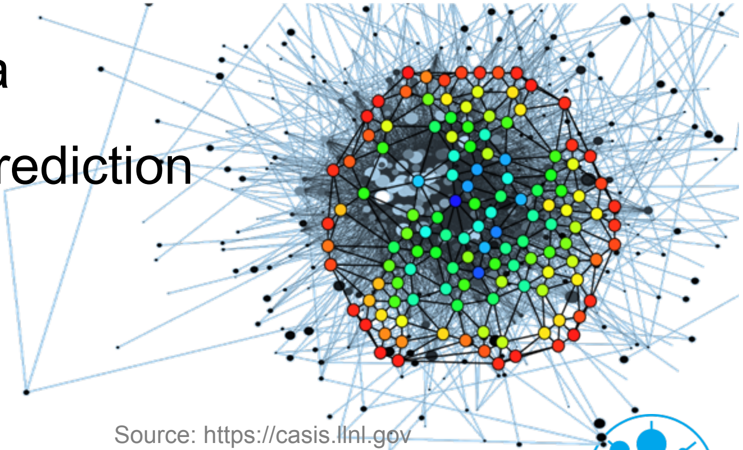
→ Dive a bit into Machine Learning Basics

Machine Learning

- What is Machine Learning?
  - Wikipedia: “*…is a subfield of computer science, evolved from the study of pattern recognition and computational learning theory in artificial intelligence. Machine Learning explores the study and construction of algorithms that can learn from and make predictions on data. Such algorithms operate by building a model from example inputs in order to make data-driven predictions or decisions, rather than following strictly static program instructions.*”

- Algorithms are the tools that perform the learning
- Algorithms are building the model, not the user
- Learn from, and make predictions on data
- The better the input data, the better the prediction

→ *What are these algorithms?*



Source: https://casis.llnl.gov

Machine Learning Categories

- Categorize, based on nature of the learning “signal” or “feedback” available to the system
- Supervised learning
  - algorithm is provided with example input and class labels (the desired output)
  - algorithm tries to find a rule that maps inputs to outputs
- Unsupervised learning
  - algorithm is provided with example input data and no class labels
  - algorithm has to find structure in input
  - E.g. discover hidden patterns in data, search for correlations
  - Learn at a specific task (find features) and learn the features themselves
- Reinforcement learning
  - Constant interaction of algorithm with dynamical environment to perform goal (e.g. drive car, don’t crash it).
  - Learn the rules of a game by playing it (goal: win game)

Machine Learning Categories

- Categorize, based on the desired output
- Classification
  - inputs are divided in two or more classes; produce model to map input to classes
  - Examples: spam filtering, signal/background classification, particle type
- Regression
  - the output is continuous rather than discrete as for classification
  - Example: What is the energy of an event with properties x,y,z
- Clustering
  - input data is to be divided into groups, which are not known beforehand
  - Where do sports-team supporters live and what is their typical age?
- Density estimation
  - finds the distributions of inputs
- Dimensionality reduction

Other types of tasks and problems

Learning to learn

- learns its own inductive bias based on past experience (frequent change of properties necessary to map input to output)

Developmental learning

- takes it one step further and generates own sequences of learning situations to acquire skill set
- employing active learning, maturation, motor synergies, and imitation
- that does sound like robots and AI, doesn’t it?

Relation to other fields

- **ML** focuses on predicting, based on *known* properties – **data mining** focuses on discovering, based on *unknown* properties
- **ML** and **Statistics** are closely related fields, now also with interdisciplinary approaches (Statistical Learning)
- **Statistics** and ML now more or less combined in **Data Science**

PART 1

> Summary

- Data explosion in the coming years
- Data is the bread and butter for physicists/scientists, but they also bring the skill set to do data science
- Data needs to be retrieved, prepared and cleaned before usage
- Data needs to be parameterized (e.g. PDFs) for generalization
- Plenty of toolkits available to do analysis – some with more focus on statistical methods than others
- Machine Learning algorithms build model, learn from data & make predictions on data
- Different classes of algorithms are used for different tasks (e.g. supervised vs. unsupervised learning)

→ PART 2: Discuss different ML algorithms

PART 2

Machine Learning Algorithms

Outlook

1. Toolkits

2. Decision Trees

3. Artificial Neural Networks

4. Deep Learning

PART 2.1

Toolkits

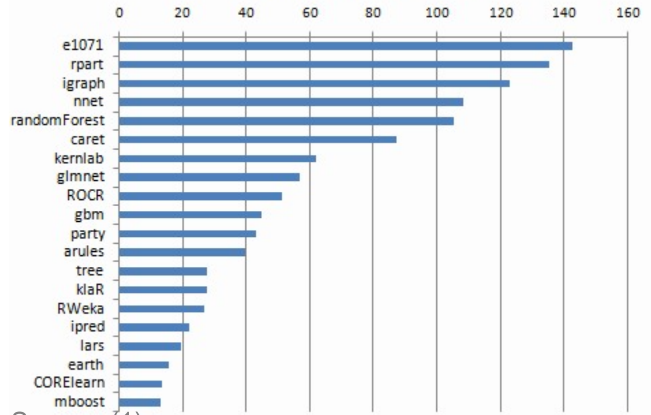
Toolkits

- ML algorithms
  - widely used in many programming languages
  - Fortran, C, C++, .NET, JAVA, python

- Nowadays, community is
  - moving towards toolkits
  - growing larger and larger
  - moving from science-driven to science/industry-driven development

- Here, introduce two packages
  - TMVA
  - scikit-learn

Top 20 R Machine Learning packages, by Downloads (000) from CRAN

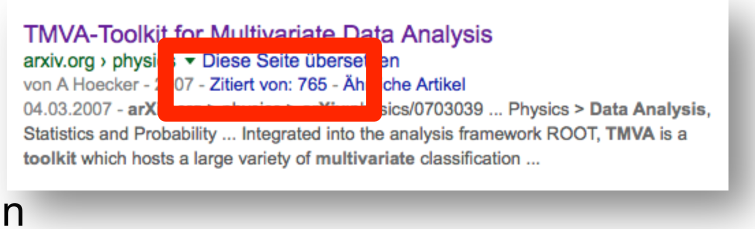


Source: (1)

TMVA

> TMVA: Toolkit for Multivariate Data Analysis

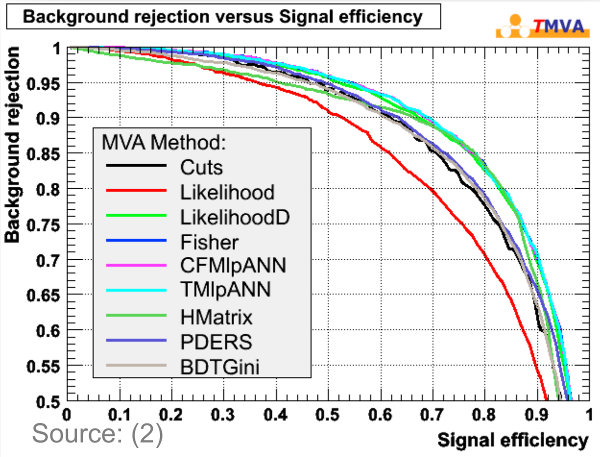
- built upon ROOT
- provides framework for *supervised learning techniques*
- provides processing, parallel evaluation and application of multivariate classification and regression methods, model selection and evaluation



TMVA-Toolkit for Multivariate Data Analysis
arxiv.org › physi ▾ Diese Seite überset en
von A Hoecker - 07 - Zitiert von: 765 - Äh che Artikel
04.03.2007 - **arX** sics/0703039 ... Physics > **Data Analysis**, Statistics and Probability ... Integrated into the analysis framework ROOT, **TMVA** is a **toolkit** which hosts a large variety of **multivariate** classification ...

> Methods

- Rectangular cut optimization
- Projective and multi-dimensional likelihood estimation
- Linear and non-linear discriminant analysis
- Neural Networks
- Decision Trees
- Support Vector Machines, etc.



Source: (2)

> Usage

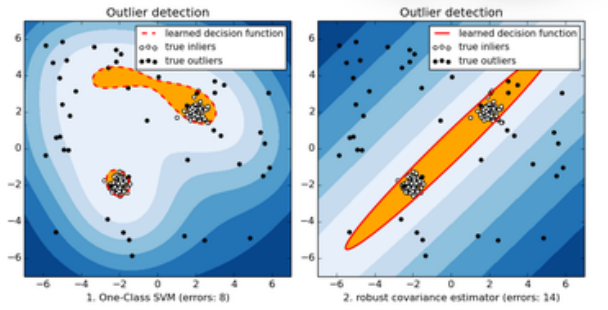
- driven by the needs for high-energy physics applications
- also applied in γ-ray astronomy (see later)

scikit-learn

Machine Learning in python

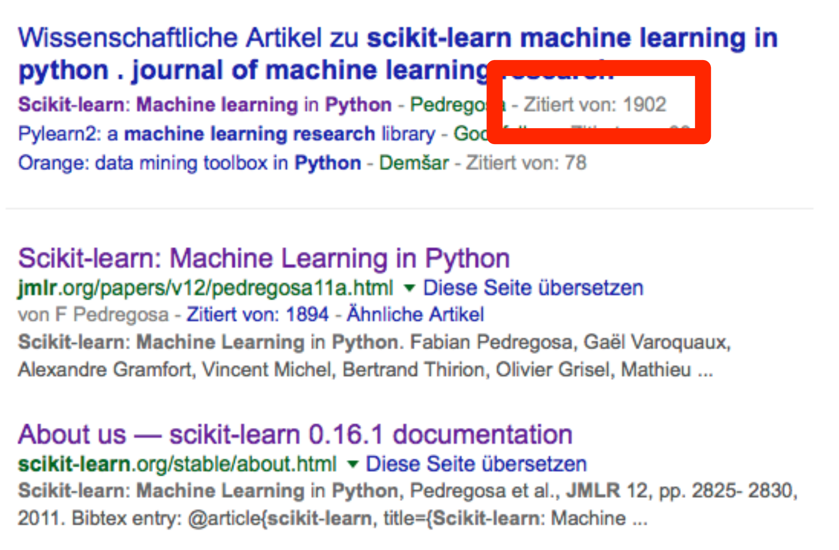
- built upon python (fast, clean, robust, comprehensive and easy-to-use)
- provides *supervised, semi-supervised, unsupervised* techniques,
- additionally: dataset loading and transformations, model selection and evaluation, scaling to bigger data

Source: (3)



Methods

- all of the TMVA methods, plus
- Clustering
- Decomposing signals in components
- Density estimation
- Unsupervised Neural Networks
- Dimensionality reduction and pre-processing



Wissenschaftliche Artikel zu **scikit-learn machine learning in python . journal of machine learning research**

Scikit-learn: Machine learning in **Python** - Pedregosa - Zitiert von: 1902

Pylearn2: a **machine learning research** library - Goo [illegible]

Orange: data mining toolbox in **Python** - Demšar - Zitiert von: 78

Scikit-learn: Machine Learning in Python

jmlr.org/papers/v12/pedregosa11a.html ▾ Diese Seite übersetzen

von F Pedregosa - Zitiert von: 1894 - Ähnliche Artikel

Scikit-learn: Machine Learning in **Python**. Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu ...

About us — scikit-learn 0.16.1 documentation

scikit-learn.org/stable/about.html ▾ Diese Seite übersetzen

Scikit-learn: Machine Learning in **Python**, Pedregosa et al., **JMLR** 12, pp. 2825- 2830, 2011. Bibtex entry: @article{**scikit-learn**, title={**Scikit-learn**: Machine ...

Usage

- widely used in science and industry (Evernote, Spotify)

PART 2.2

Decision Trees

Decision Trees

> Concept

- a decision tree is a predictive modeling tool
- uses a tree structure to represent a number of possible decision paths
- *classification trees* predict class
- *regression trees* predict continuous (real) number
- decision trees map a *n*-dimensional input to a 1-dimensional output

Decision Trees

> A general description

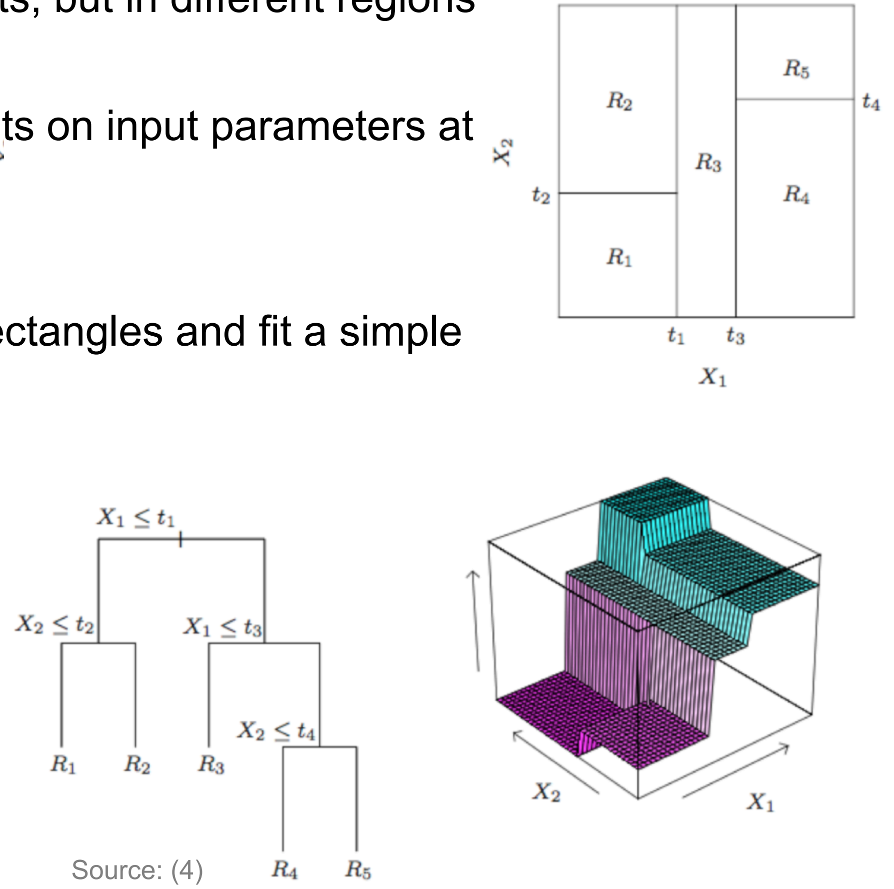
Consider inputs X_1 and X_2 mapped onto Y

a) Feature space can be modeled by constants, but in different regions of parameter space

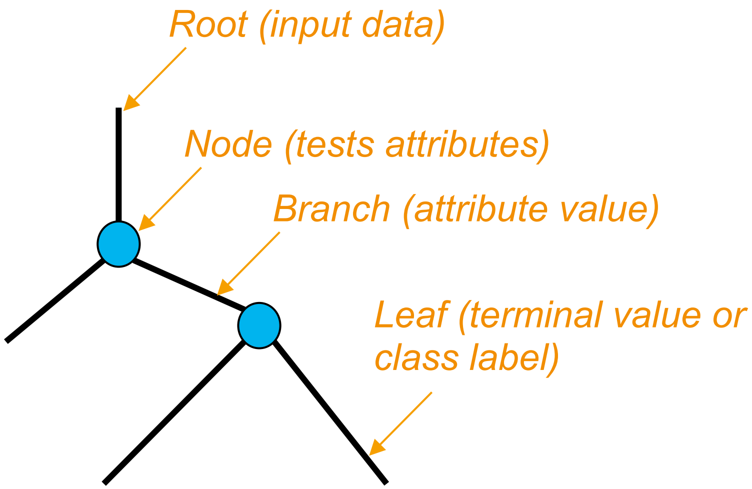
b) Described by a binary decision tree with cuts on input parameters at *nodes*

c) Terminal nodes declare class labels

→ Decision Trees split parameter space into rectangles and fit a simple model in each one (e.g. a constant)



Source: (4)



Root (input data)

Node (tests attributes)

Branch (attribute value)

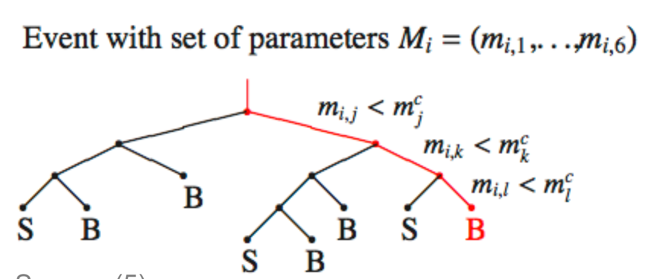
Leaf (terminal value or class label)

Decision Trees

Learning

- using representative training data
- identify variables/parameters with classification potential

1. Split the source set (consisting of all classes) into subsets based on an attribute value test (binary decision or cut on continuous variable)
2. Repeat process on subsets → grow decision tree
3. Stop at a certain point (purity in leaves)


Event with set of parameters $M_i = (m_{i,1}, \ldots, m_{i,6})$
$m_{i,j} < m_j^c$
$m_{i,k} < m_k^c$
$m_{i,l} < m_l^c$
S
B
B
S
B
B
S
B
S
B

Source: (5)

User input

- Training set consisting of events of known classes
- List of variables with classification potential

Decision Tree output

- Each leaf returns signal/background (classification mode)
- Each leaf returns a specific value of the target variable

Decision Tree Learning

Advantages

- simple to understand and interpret
- almost no data preparation
- able to handle both, numerical and categorical data
- ‘white box’, completely transparent
- robust, even if the assumptions for the model do not perfectly describe the real data
- performs well with large data sets
- can classify data for which attributes are missing

Limitations

- decision trees learn to locally optimize
- Overfitting and over-complex trees do not generalize well from training data
- Some concepts are hard to learn by trees (e.g. parity, odd/even number classification)

Decision Tree: Ensemble Methods

- Single decision trees tend to overfit
  - i.e. instable to statistical fluctuations in the training sample

  → classifier response is altered compared to real data

  → lower performance in classification problems
- Ways out
  - train a forest of decision trees
  - classify events based on a majority vote of many trees
  - since same input training sample is used, we have to come up with different tree structures

  → Boosting (Boosted Decision Trees)
    - re-weighting of misclassified events, when building the next decision tree

  → Randomization (Random Forests)
    - randomly choose subset of events or classifying parameters for training of single tree

  → Pruning
    - grow trees to maximum extent, cut back insignificant leaves

Example: BDTs & IACTs

> Improve sensitivity of IACTs in the analysis

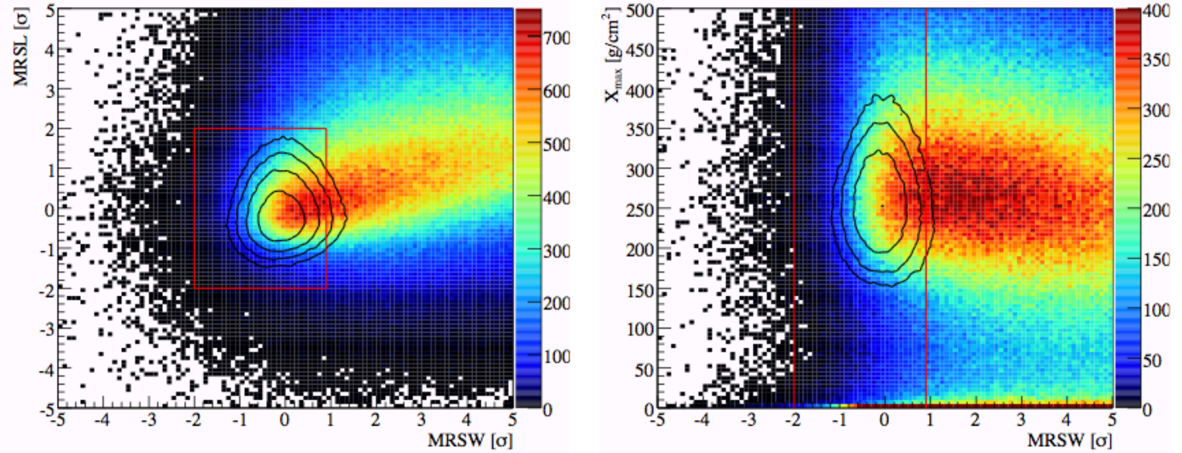
- improve reconstruction of showers
- two possibilities:
  - classification of events using multivariate analysis techniques (MVAs; e.g. neural networks, boosted decision trees)
  - use information in all pixels to do reconstruction (full-blown, CPU-intense likelihood fitting)

> Why cuts on scaled parameters are not sufficient

- Cuts on shower shape (Width/Length) are box cuts
- other parameters have separation potential as well (such as height of maximum Cherenkov light emission)
- box cuts ignore correlations

→ MVAs can take care of this

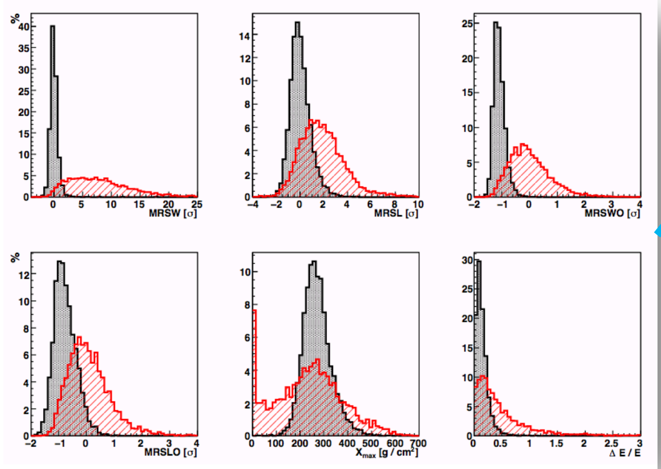
Colour = CRs
Contours = γ rays

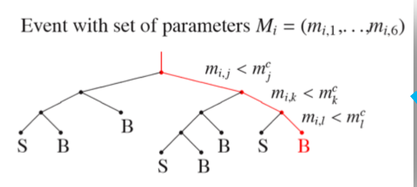


BDTs for γ/hadron classification

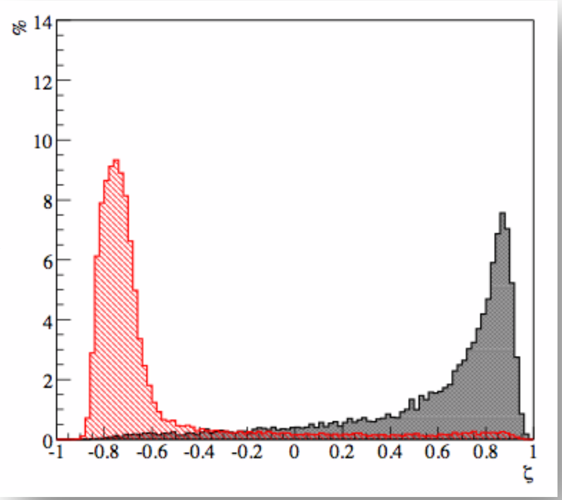
BDT settings

- 200 trees trained (compromise between performance and processing)
- Gini index used to split at nodes, similar performance achieved with other techniques
- Stop splitting at $(N_1 + N_2) / (10 * N^2_{par})$ → taking into account the training statistics and number of training parameters
- Number of steps when scanning input parameters for best cut set to 100
- Input parameters (everything with classification potential)
    - MRSW, MRSL, MRSWO, MRSLO, σE/E, X_{max}
- Output is (cut) parameter that measures hadroness or γ-ray likeliness





Source: (5)

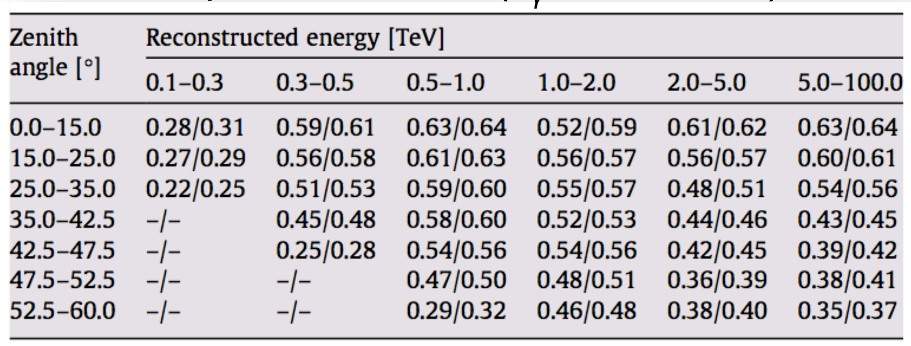


Some more details

> Training sample

- classification should work for full dynamic range of instrument (energy, zenith angle)
- if distribution of input parameters for signal and background change as a function of observation condition, the BDT response will change as well
- Train in energy and zenith angle bands
- Training statistics between 120k/240k events and 15k/25k events for γ/hadrons
- Cut on BDT output alone not a good idea (would result in a signal efficiency that changes with observation conditions) → cut on signal efficiency instead
- Note the range of BDT output distributions

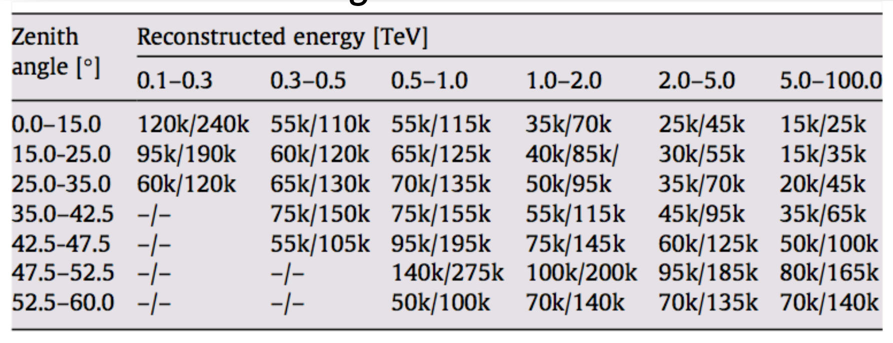
Cut parameters for (ε_γ = 0.84/0.83)



Zenith angle [°]	Reconstructed energy [TeV]					
	0.1–0.3	0.3–0.5	0.5–1.0	1.0–2.0	2.0–5.0	5.0–100.0
0.0–15.0	0.28/0.31	0.59/0.61	0.63/0.64	0.52/0.59	0.61/0.62	0.63/0.64
15.0–25.0	0.27/0.29	0.56/0.58	0.61/0.63	0.56/0.57	0.56/0.57	0.60/0.61
25.0–35.0	0.22/0.25	0.51/0.53	0.59/0.60	0.55/0.57	0.48/0.51	0.54/0.56
35.0–42.5	–/–	0.45/0.48	0.58/0.60	0.52/0.53	0.44/0.46	0.43/0.45
42.5–47.5	–/–	0.25/0.28	0.54/0.56	0.54/0.56	0.42/0.45	0.39/0.42
47.5–52.5	–/–	–/–	0.47/0.50	0.48/0.51	0.36/0.39	0.38/0.41
52.5–60.0	–/–	–/–	0.29/0.32	0.46/0.48	0.38/0.40	0.35/0.37

Training statistics

Source: (5)

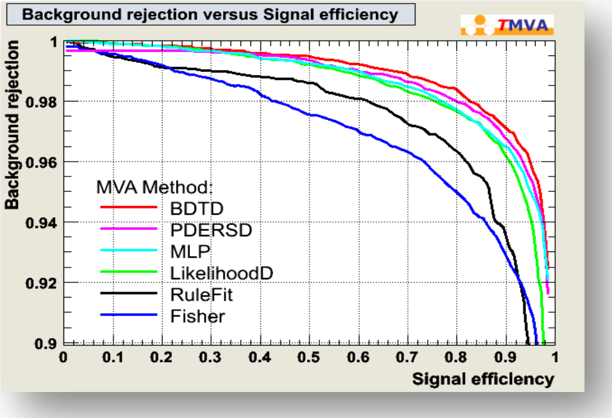


Zenith angle [°]	Reconstructed energy [TeV]					
	0.1–0.3	0.3–0.5	0.5–1.0	1.0–2.0	2.0–5.0	5.0–100.0
0.0–15.0	120k/240k	55k/110k	55k/115k	35k/70k	25k/45k	15k/25k
15.0-25.0	95k/190k	60k/120k	65k/125k	40k/85k/	30k/55k	15k/35k
25.0-35.0	60k/120k	65k/130k	70k/135k	50k/95k	35k/70k	20k/45k
35.0–42.5	–/–	75k/150k	75k/155k	55k/115k	45k/95k	35k/65k
42.5-47.5	–/–	55k/105k	95k/195k	75k/145k	60k/125k	50k/100k
47.5–52.5	–/–	–/–	140k/275k	100k/200k	95k/185k	80k/165k
52.5–60.0	–/–	–/–	50k/100k	70k/140k	70k/135k	70k/140k

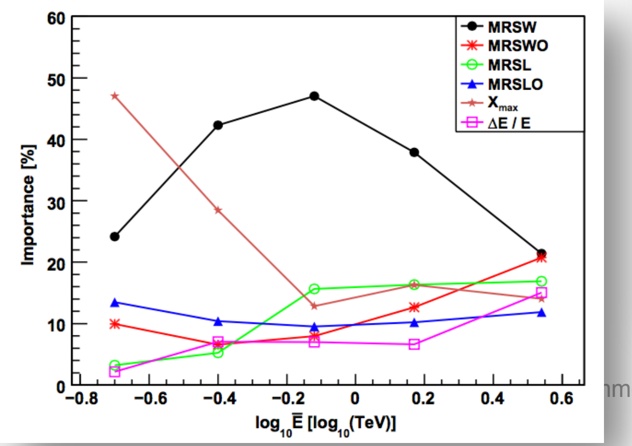
BDTs in H.E.S.S.

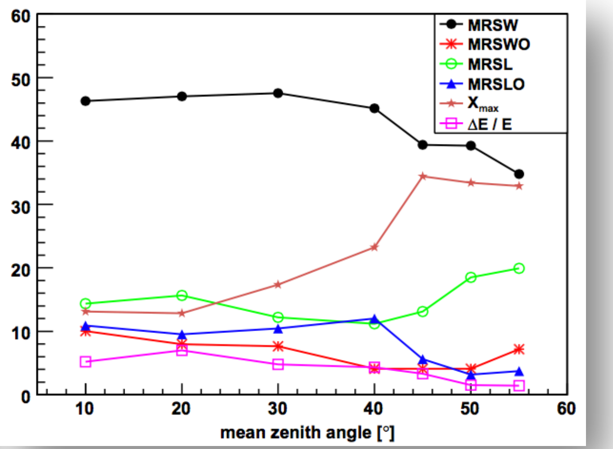
Results

- BDTs are a robust, simple and sensitive analysis method (better than others)
- Training in energy and zenith angle bands can take out parameter dependencies
- Classifier response changes as a function of energy/zenith
- Parameter importance changes as a function of energy/zenith
  - X_{max} at low energies
  - MRSW at medium energies
  - all parameters at high energies



Source: (5)





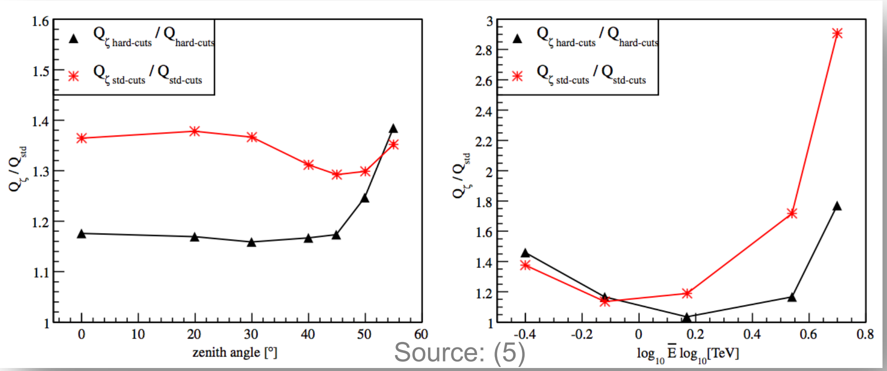
BDTs in H.E.S.S.

> Tests on real data

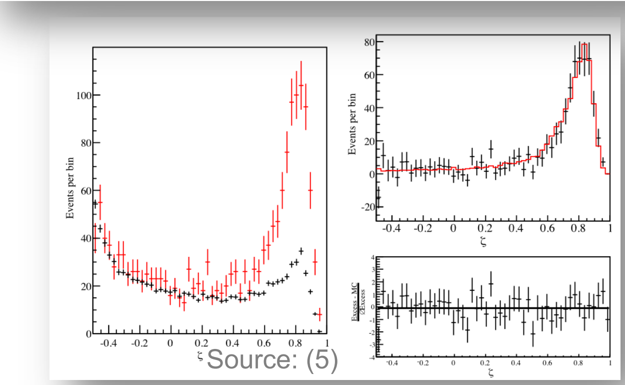
- BDTs are much more sensitive than box cuts performed on individual parameters
- for HESS and Hillas-based parameters, 45% less observation time for one cut set
- Performance improvement across all energies
- Very good agreement between Monte-Carlo simulations and real data

> Lessons learnt

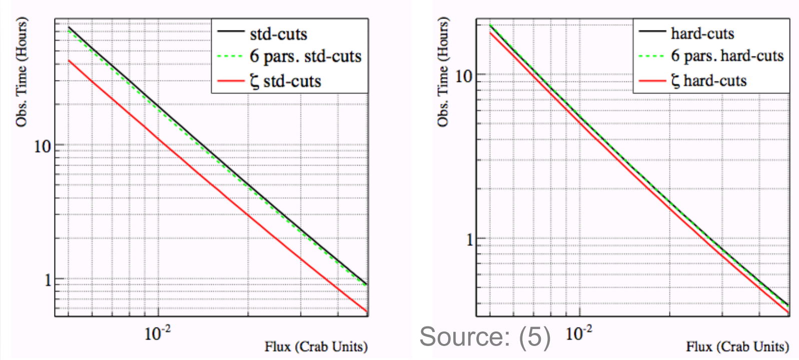
- Test different classifiers and settings other than default
- Deep trees are good for complex problems
- Easy to extend to more input parameters (Becherini et al. 2011, Naumann-Godo et al. 2009)



Source: (5)



Source: (5)

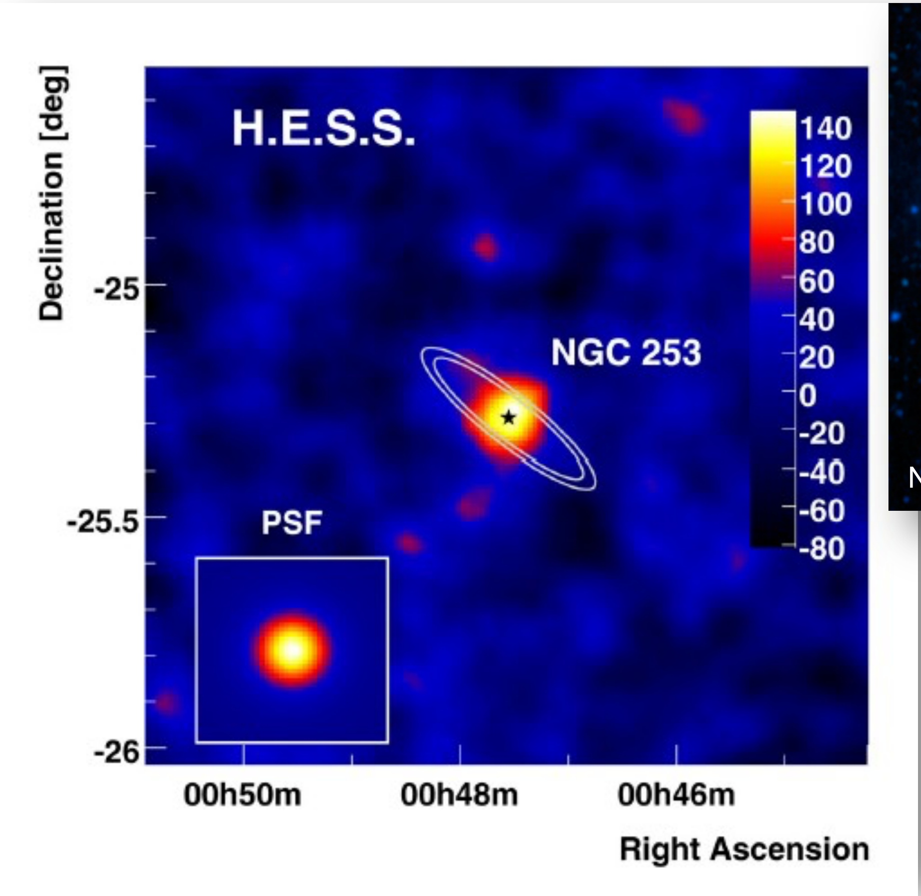


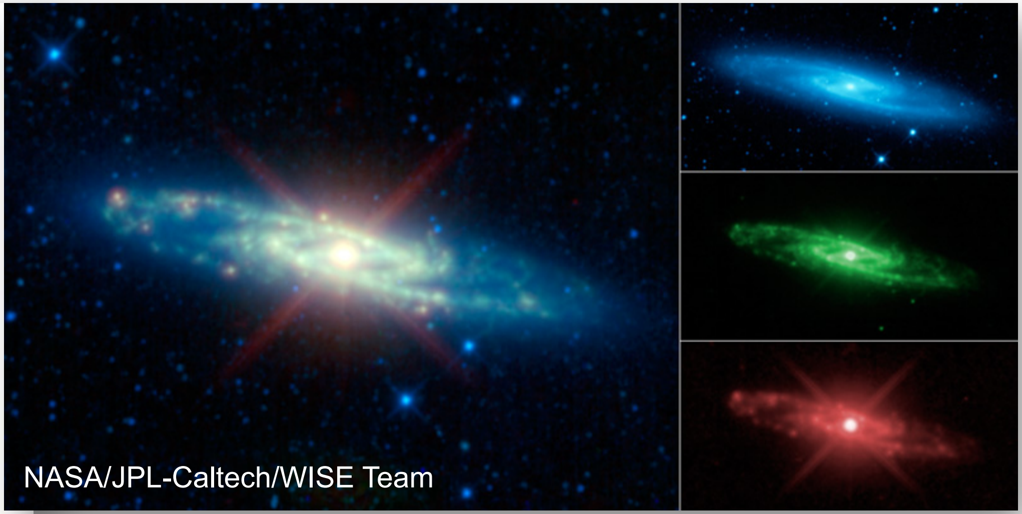
Source: (5)

Major Achievement

- Hint of signal seen from NGC 253 after 50 hours of data taking with BDT analysis triggered deep observations

→ First detection of a Starburst Galaxy in TeV γ rays (Science, 2009)




NASA/JPL-Caltech/WISE Team

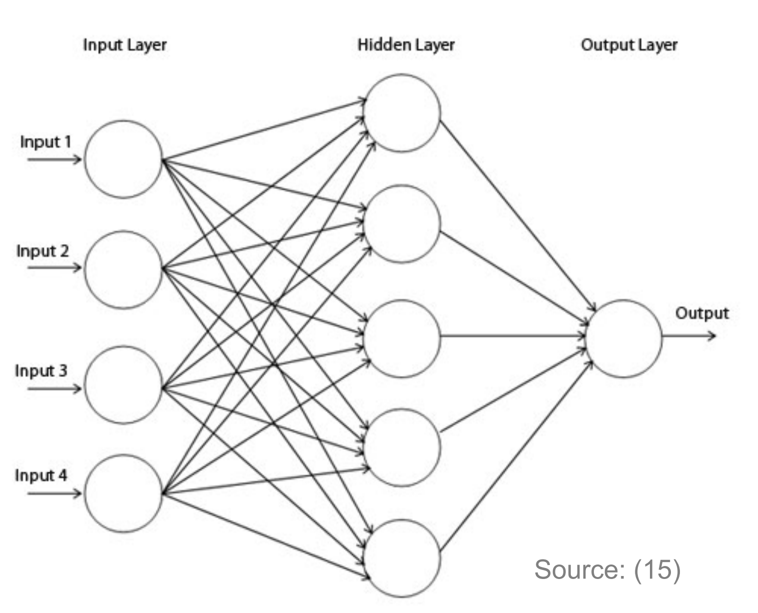
PART 2.3

Artificial Neural Networks

Neural Networks (NNs)

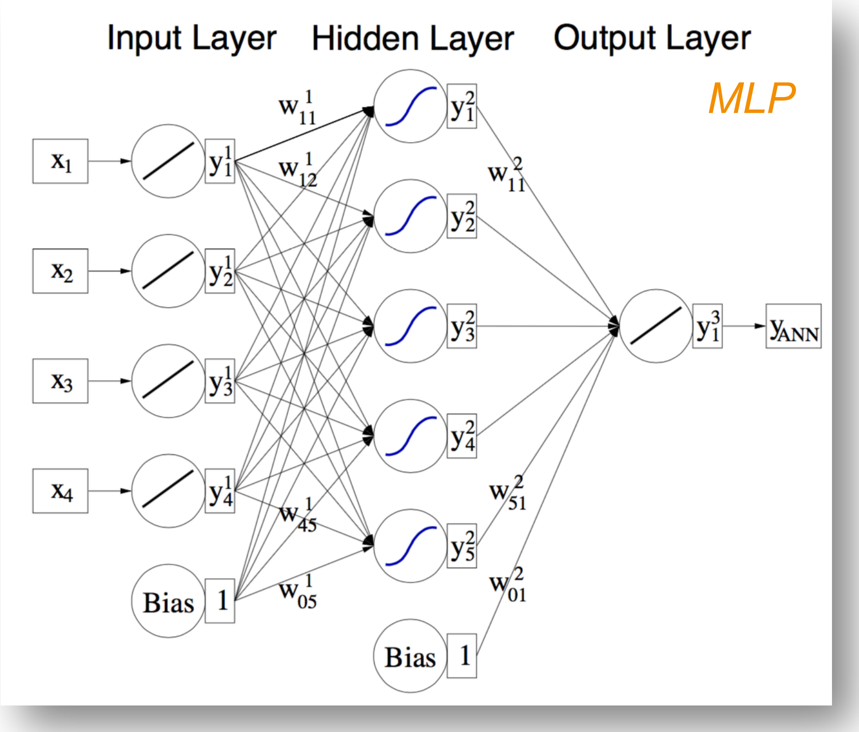
> Concept

- Inspired by humans' brain
- Is simulated collection of inter-connected "neurons"
- Each neuron returns certain response to set of input signals
- Neural net is put into defined state by applying external input signals that can be measured by the response of one (or multiple) output neurons
- Connections have numeric weights that determine importance of inputs



Source: (15)

Working Principle



Source: (18)

> Artificial NNs

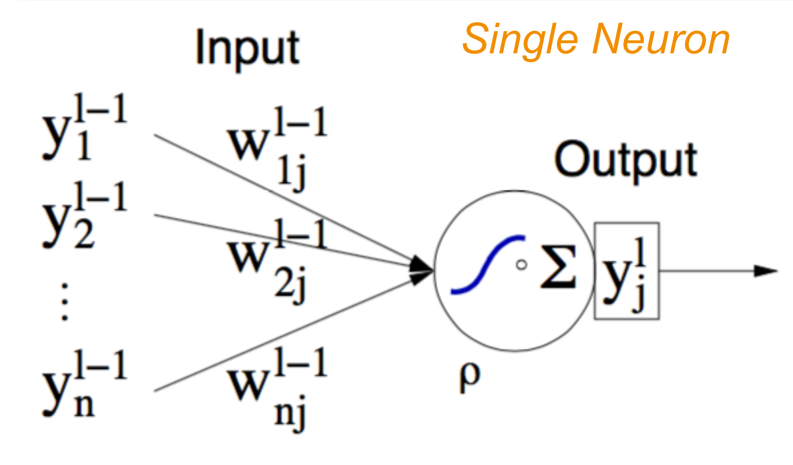
- ANN is a mapping of n-dimensional input parameter space $x_1, \ldots, x_n$ to m-dimensional output parameter space $y_1, \ldots y_m$
- m = 1 is classical signal/background problem
- mapping linear if all neurons have linear response; non-linear if response of one neuron is non-linear
- behavior is determined by
  - layout of neurons
  - weights of inter-neuron connections, *w*
  - and the neuron response function (activation function)
- Regression layout similar, but with one output neuron per target

Neuron response function

> Mathematical description

- neuron response function *ρ* maps $i_1, \ldots, i_n$ onto neuron output
- can often be separated into a synapse function *κ* and a neuron activation function *α*
- In e.g. TMVA, the functions are implemented as

$$\kappa : \left(y_1^{(\ell)}, .., y_n^{(\ell)} | w_{0j}^{(\ell)}, .., w_{nj}^{(\ell)}\right) \rightarrow \begin{cases} w_{0j}^{(\ell)} + \sum_{i=1}^{n} y_i^{(\ell)} w_{ij}^{(\ell)} & \textit{Sum}, \\ w_{0j}^{(\ell)} + \sum_{i=1}^{n} \left(y_i^{(\ell)} w_{ij}^{(\ell)}\right)^2 & \textit{Sum of squares}, \\ w_{0j}^{(\ell)} + \sum_{i=1}^{n} |y_i^{(\ell)} w_{ij}^{(\ell)}| & \textit{Sum of absolutes}, \end{cases}$$



Source: (18)

$$\alpha : x \rightarrow \begin{cases} x & \textit{Linear}, \\ \frac{1}{1+e^{-kx}} & \textit{Sigmoid}, \\ \frac{e^x - e^{-x}}{e^x + e^{-x}} & \textit{Tanh}, \\ e^{-x^2/2} & \textit{Radial}. \end{cases}$$

Feed-forward Networks and Learning

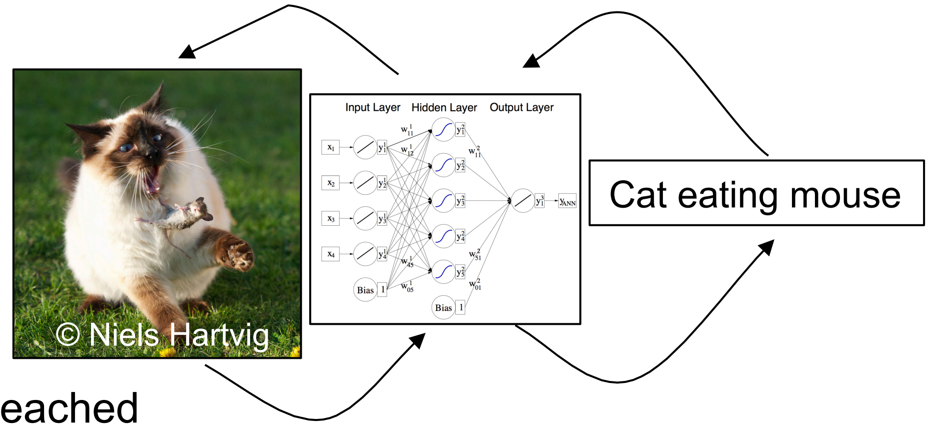
> Characteristics

- ANNs are organized in layers: input, output and hidden layers
- Each neuron in one layer is connected to all neurons in next layer → feed forward
- No connection among neurons in same layer

> Multiple hidden layers possible

> Back-propagation and learning

- Provide *N* training events of known type
- Define set of input parameters
- Let ANN classify event
- Compare with expectation
- Adjust weights, repeat until minimum is reached
- Adjust weights by means of an error function (compare output with expectation)
- change weights via gradient decent (differentiate functions)



Cat eating mouse

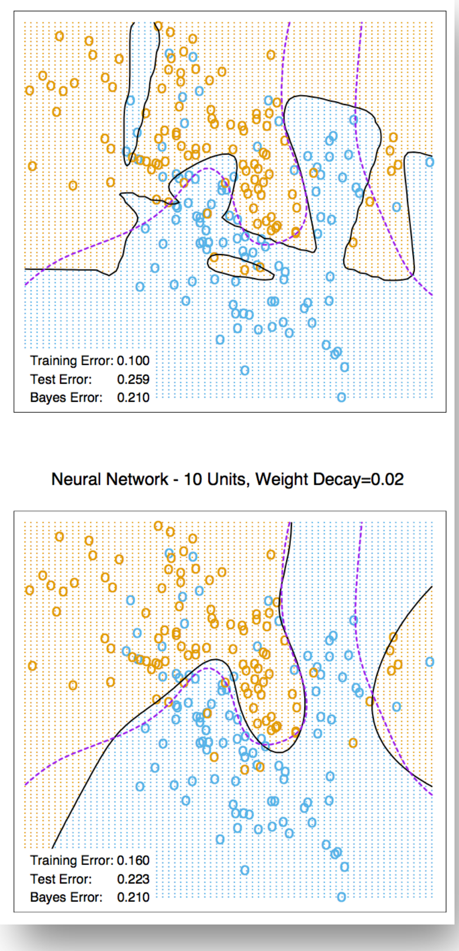
Practical training issues

Generally

- problem is overparametrized
- optimisation is unstable
- issues may appear depending on the non-linear function type used
- Overfitting is an issue (add *weight decay* or *weight elimination* (i.e. adding penalty to the error function)

Network architecture

- influences the performance a lot
- too many hidden layers is better than not enough
- less hidden layers means worse response to non-linearities in data
- Multiple layers allow for construction of hierarchy and search for different levels of detail (resolution) in data



Neural Network - 10 Units, Weight Decay=0.02

Source: (19)

Preliminary Summary

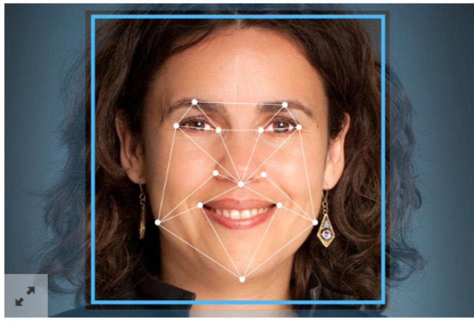
- NNs attracted a lot of attention, but
  - tend to overfit (there are ways out of it)
  - are computationally expensive → NNs with one hidden layer
  - are black boxes
  - most applications could be realized with other, more transparent methods, achieving the same performance
  - good in cases where prediction without interpretation is required
  - less so if model interpretation is needed, or where physical quantities of inputs and their inter-relation is required
- Mid 2000's
  - Feature extraction in big data
  - renewed interest in neural networks, especially in the context of deep learning
  - industry entered the game

Deep Learning

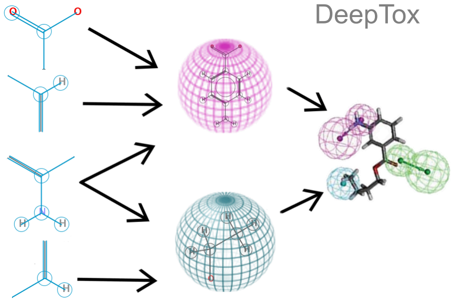
- Is the new kid on the block
- Most popular approach to tackle artificial intelligence (AI) problems
- Used to describe the world (i.e. data) with networks of hierarchical non-linear functions (i.e. NNs or NNs combined with other classifiers)
- Used by biggest (i.e. data-intense) internet companies in the world

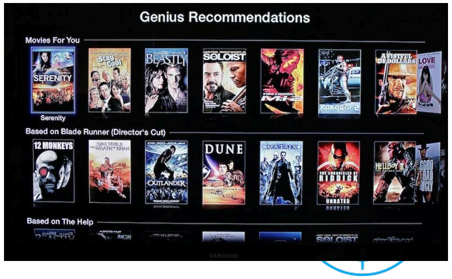
Adobe facebook twitter flickr Google IBM Microsoft

- Very successful in performing specific tasks
  - Social media (e.g. image recognition)
  - Consumer electronics (smartphones, wearables)
  - Entertainment and media (“users who bought this,…”)
  - Medicine, Defense & Intelligence



Shutterstock





Genius Recommendations

Movies For You

Serenity

Based on Blade Runner (Director's Cut)

Based on The Help

Acknowledgement

> We are now entering the part of the lecture, Stefan didn't know much about two years ago. Hence, it is strongly influenced by recent computer science papers, blog posts, presentations, conference proceedings, and NVIDIA developer courses ☺

- [HS1997], Hochreiter, S., and Schmidhuber, J., *"Long Short-Term Memory"*, Neural Computation, 9(8), 1735, 1997
- [K2012] Krizhevsky, A., Sutskever, I., and Hinton, G.E., *"ImageNET Classification with Deep Convolutional Neural Networks"*, Advances in Neural Information Processing Systems 23 (NUPS 2012), 2012
- [Z2013] Zeiler, M.D., and Fergus, R., *"Visualizing and understanding convolutional networks."*, CORR, abs/1311.2901, 2013
- [K2015] Andrej Karpathy blog, *"The unreasonable effectiveness of recurrent neural networks"*, 21.05.2015, http://karpathy.github.io/2015/05/21/rnn-effectiveness/
- [D2014] Sander Dieleman blog, *"Recommending music on Spotify with deep learning"*, 05.08.2014, http://benanne.github.io/2014/08/05/spotify-cnns.html
- [S2015] Schmidhuber, J., *"Deep learning in neural networks: An overview"*, Neural Networks, 61, (2015), 85 – 117
- [M2015] Mnih, V. et al., *"Human-level control through deep reinforcement learning"*, Nature, 518, 529, 2015
- [N2015] NVIDIA webinars/presentations on deep learning, https://developer.nvidia.com/deep-learning
- and Wikipedia of course

Convolutional Neural Network (CNN)

- Widely used for image and video recognition
- Were very popular in the 1990’s for hand-written digit classification and face detection
- Recently renewed interest thanks to
  - much larger data sets of e.g. images that are labeled (millions)
  - better model generalization strategies (see dropout method)
  - powerful GPU implementations allowing for very large models to be trained
- Utilize
  - Backpropagation (see before)
  - Receptive fields
    - multiple layers of small collections of neurons that learn about a part of an image
    - connected to better describe the boundaries
    - performed for every layer
  - Rectified Linear Units (ReLU)
    - model neuron response as f(x) = max(0,x), rather than tanh or sigmoid function

Convolutional Neural Network (CNN)

Typical layers

- Convolutional layers
  - convolves input image with set of learnable filters
  - reproduces one feature in the output image
  - weights are shared, i.e. same filter is used for all pixels in a receptive field
- Pooling layer
  - compute max/average value of certain feature per sub-image

    → increase robustness to translations in images
- Dropout method
  - prevents overtraining by leaving out nodes with probability 0.5 at each training step → new network architecture every round reduces susceptibility to rely on a few significant nodes
- Loss layer
  - employs different loss functions for different applications (single class, probability, real values)

K2012 approach

- Idea
  - map 2D color input image to probability vector over different classes via series of layers
- Each layer consists of
  1. Convolution of previous layer output with set of learned filters
  2. Passing response through ReLU functions
  3. optionally max-pooling over neighbouring kernels
  4. optionally renormalization through contrast operation

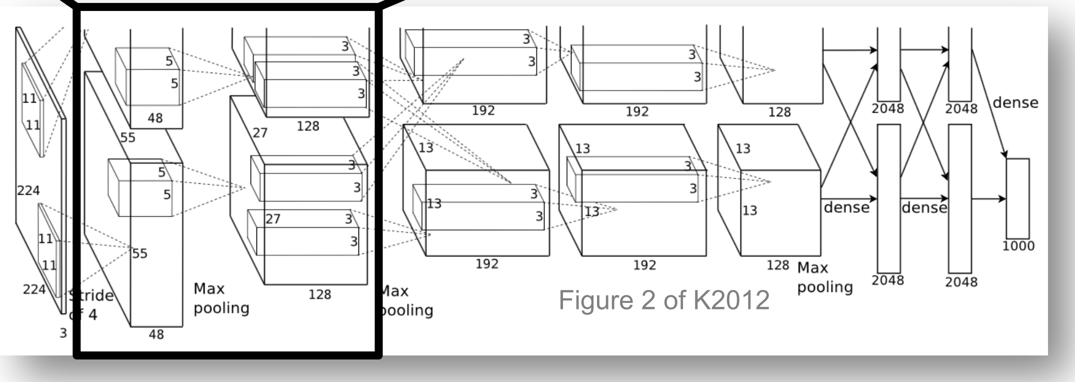


Figure 2 of K2012

Visualizing the K2012 network architecture (Z2013)

- In general hard to interpret or even to understand why certain layouts perform well
- Hence also hard to guide for future improvements
- Z2013 developed visualization technique that
  - is based on a multi-layered deconvolution network (Zeiler et al. 2011)
  - features are far from random or un-interpretable
  - reveal intuitive properties such as compositionality
  - minimum depth of network is vital to performance

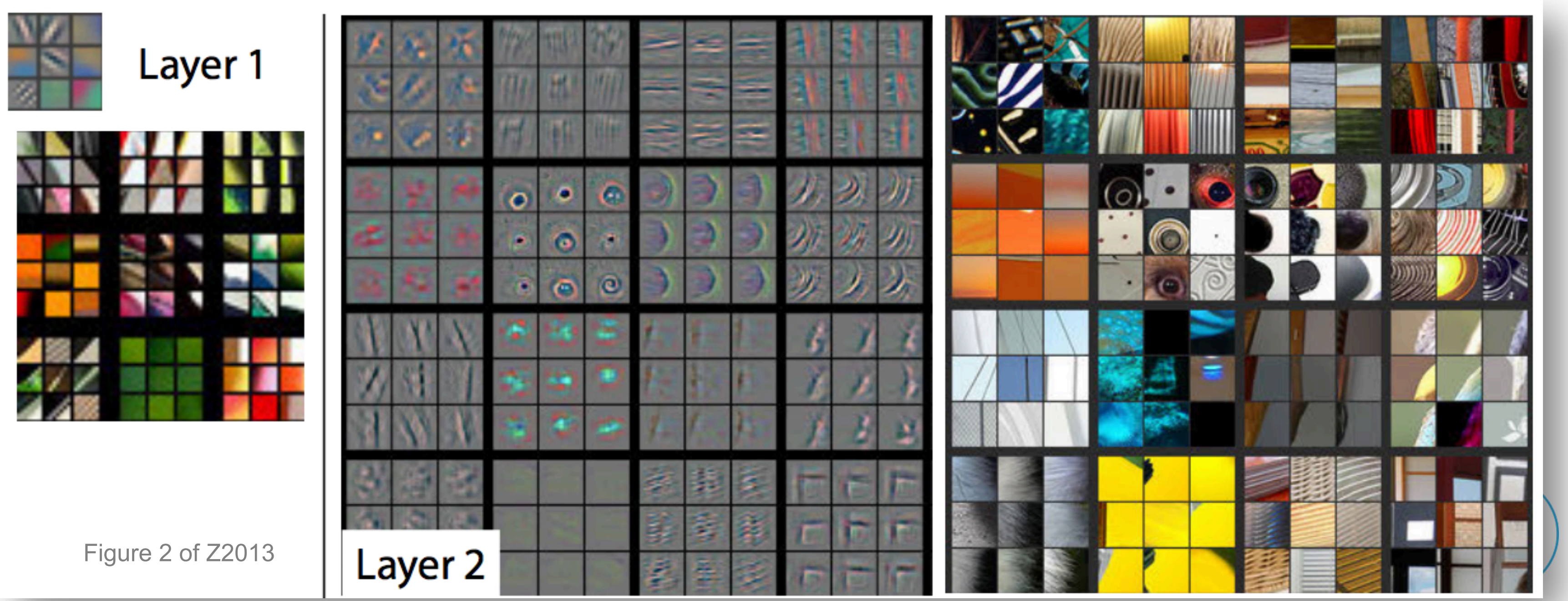


Figure 2 of Z2013

Recurrent Neural Network (RNN)

> Properties

- have feedback connections that can ‘remember’ information about previous training events
- put NN into internal state, allowing for dynamic behavior

> Application

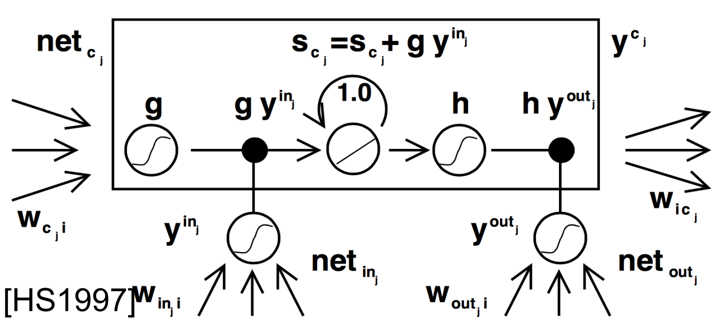
- handwriting/speech recognition
- learning syntax from any input (latex, code, wikipedia)

> Back-propagation through time (BPTT)

- error signals ‘flow back in time’, by connecting units to previous layers

> Reinforcement learning

- inspired by behavioral psychology: NN interacts with environment through observation → action → reward
- use fitness or reward function, where the goal of the NN is to maximize future reward

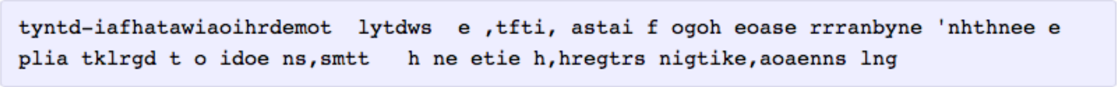


[HS1997]

Applications (K2015)

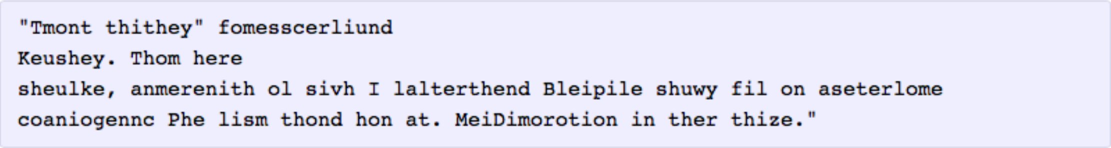
How is it learning

- Use Leo Tolstoys “War and Peace” as training input, sample every x training cycles
- 100 cycles: jibberish, but note words separated by spaces



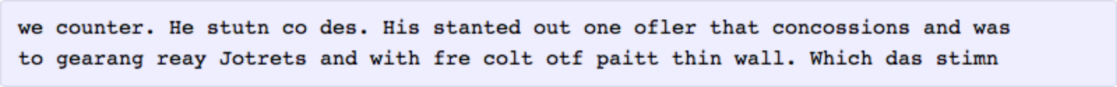
```
tyntd-iafhatawiaoihrdemot  lytdws  e ,tfti, astai f ogoh eoase rrranbyne 'nhthnee e
plia tklrgd t o idoe ns,smtt   h ne etie h,hregtrs nigtike,aoaenns lng
```

- 300 cycles: learns periods at end are followed by spaces



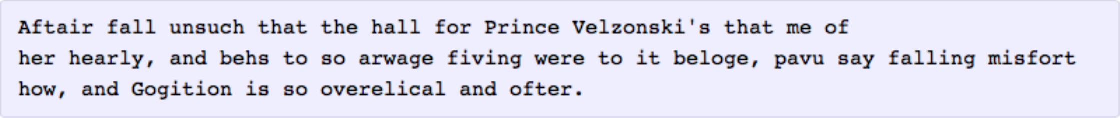
```
"Tmont thithey" fomesscerliund
Keushey. Thom here
sheulke, anmerenith ol sivh I lalterthend Bleipile shuwy fil on aseterlome
coaniogennc Phe lism thond hon at. MeiDimorotion in ther thize."
```

- 500 cycles: shortest and most common words



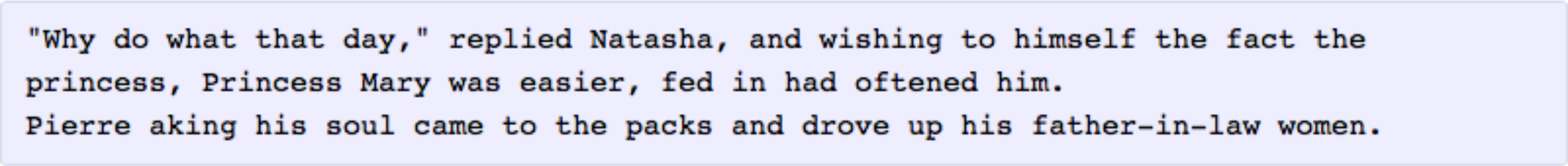
```
we counter. He stutn co des. His stanted out one ofler that concossions and was
to gearang reay Jotrets and with fre colt otf paitt thin wall. Which das stimn
```

- 700 cycles: more and longer words appearing



```
Aftair fall unsuch that the hall for Prince Velzonski's that me of
her hearly, and behs to so arwage fiving were to it beloge, pavu say falling misfort
how, and Gogition is so overelical and ofter.
```

- 2000 cycles: quotations, questions, exclamation marks, names, words

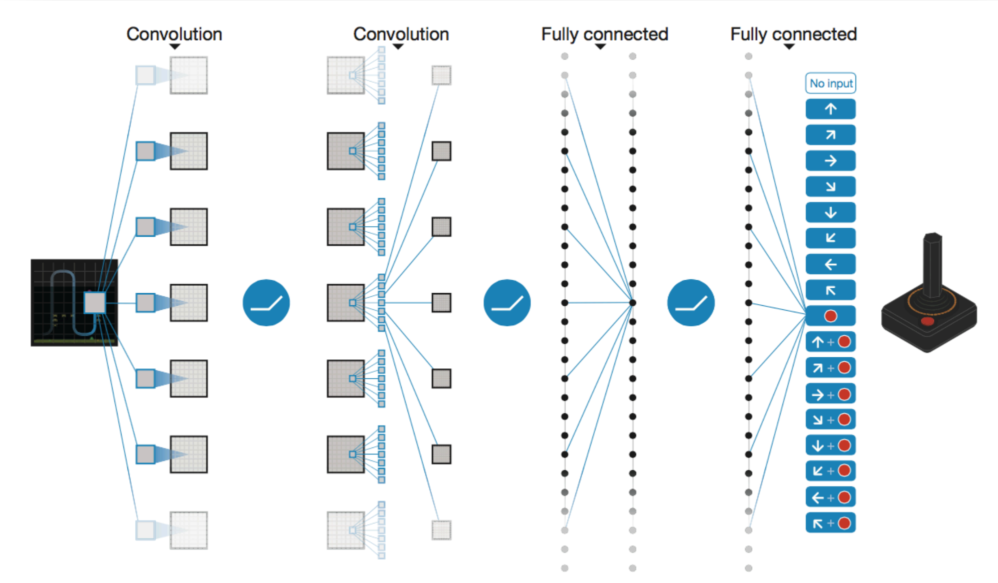


```
"Why do what that day," replied Natasha, and wishing to himself the fact the
princess, Princess Mary was easier, fed in had oftened him.
Pierre aking his soul came to the packs and drove up his father-in-law women.
```

Learning NNs to play ATARI 2600 games (M2015)

Layout

- combines reinforcement learning (LSTMs) with convolutional NNs
- 84 x 84 x 4 image followed by 3 convolutional layers, and 2 fully connected layers
- output is action with joystick
- hidden layers are followed by ReLUs



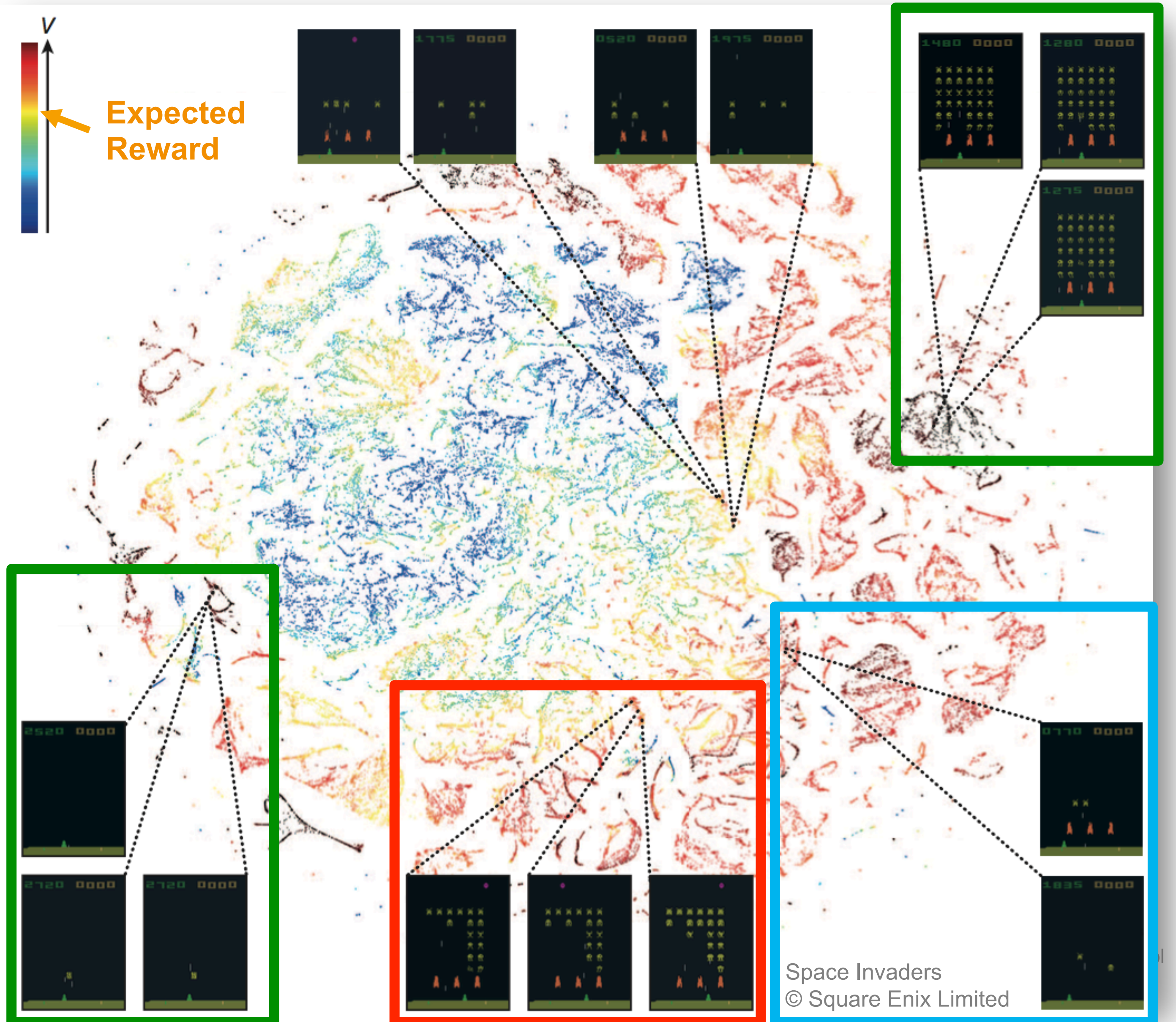
Training

- each game is one set of trained NN
- via a reward function
- agent selects from set of allowed actions, changing internal state → modifying the emulator → reward

→ NN learns based on sequences of actions and observations

Learning NNs to play ATARI 2600 games (M2015)

Visualizing game states of last hidden layer



Space Invaders
© Square Enix Limited

- High reward scenes
  - completing a screen leads to a new screen
- Medium reward scenes
  - in the middle of a level, where rewards are less imminent
  - orange bunkers not that significant towards end of level

How come there is so much fuzz about that now? (N2015)

- Answer is simple
  - Big Data to learn deep neural networks
  - New deep learning techniques, and (most importantly)
  - computing power via CPUs and GPUs
- Why GPUs
  - Neural Networks explore functions, matrices, vector operations, connections
  - Requires many cores to perform calculations (cloud-like CPU computing infrastructure)
  - GPUs are designed for these kind of operations (e.g. *“Deep learning with COTS HPC systems”*, Coates, et al. ICML 2013)
  - GPUs make deep learning accessible when organized in servers with interconnections

Google datacenter

1000 CPU servers
2000 CPUs
16,000 cores

600kWatts, $5M

3 GPU-acc. servers
12 GPUs
18,432 cores

4kWatts, $33k

Stanford AI labs

That’s the game changer

Summary and outlook

Big Data

- we are drowning in data, but
- we have tools to crawl data for relevant information

Machine Learning

- is the way to go
- just have to be smart about the methods and decide what tools to use when/how

Deep Neural Networks

- are widely used in industry and increasingly in science
- if things are industry standard, good to think about how to use them in science
- GPUs have revolutionized the field
- So far we are still computing limited (signal transmission time)

Summary and outlook

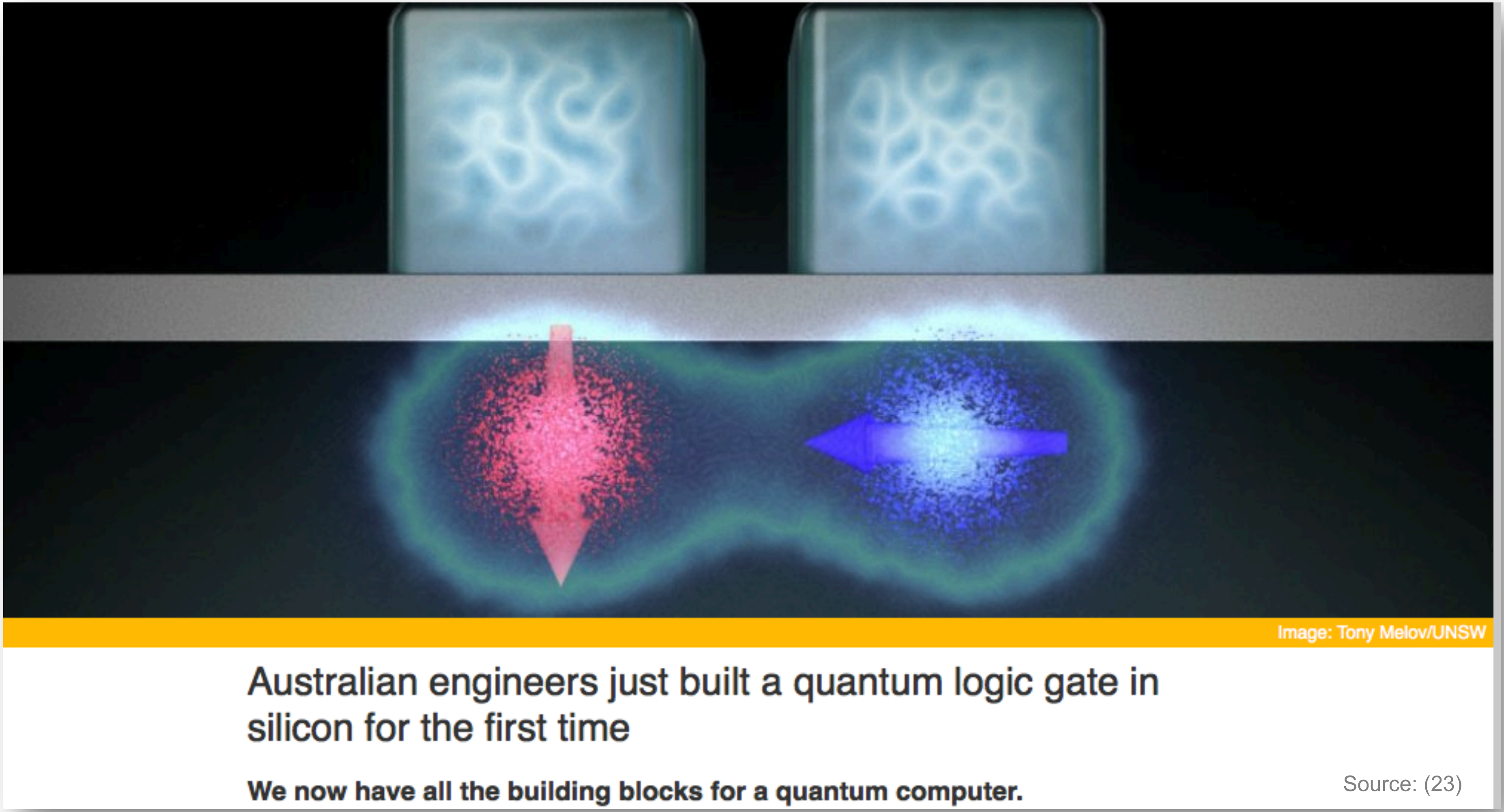


Image: Tony Melov/UNSW

Australian engineers just built a quantum logic gate in silicon for the first time

We now have all the building blocks for a quantum computer.

Source: (23)

- In this context, think about this…

Literature and References

1 http://www.kdnuggets.com/2015/06/top-20-r-machine-learning-packages.html

2 https://root.cern.ch/root/Version511.news.html

3 https://scikit-learn.org

4 Figure 9.2 from “The Elements of Statistical Learning: Data Mining, Interference, and Prediction”, Hasti, Tibshirani, Friedman, 2009, Springer Series in Statistics

5 Ohm, S., van Eldik, C., Egberts, K., (2009) Astropart. Phys.,**31**, 383

6 Figure 9.3 from “The Elements of Statistical Learning: Data Mining, Interference, and Prediction”, Hasti, Tibshirani, Friedman, 2009, Springer Series in Statistics

7 „Large Synoptic Survey Telescope 3 4 render 2013“ von LSST Project Office - http://www.lsst.org/gallery/telescope-rendering-2013. Lizenziert unter CC-BY-SA 4.0 über Wikimedia Commons - https://commons.wikimedia.org/wiki/File:Large_Synoptic_Survey_Telescope_3_4_render_2013.png#/media/File:Large_Synoptic_Survey_Telescope_3_4_render_2013.png

8 „The E-ELT“ von Swinburne Astronomy Productions/ESO - ESO. Lizenziert unter CC-BY 4.0 über Wikimedia Commons - https://commons.wikimedia.org/wiki/File:The_E-ELT.jpg#/media/File:The_E-ELT.jpg

9 http://www.ptf.caltech.edu/image/ptf100814

10 http://www.ptf.caltech.edu/image/ptf100814

11 Law et al., 2009, PASP, **121**, 1395 – 1408

12 Brink, et al., 2013, MNRAS, **435**, 1047 – 1060

13 Wright, et al., 2015, MNRAS, **449**, 451

14 http://www.astro.caltech.edu/digging/content/pdf/Richards_20111212.pdf

15 https://dmm613.files.wordpress.com/2014/12/single_hidden_layer_ann.jpg

16 http://www.alanturing.net/turing_archive/pages/reference%20articles/connectionism/Turing's%20neural%20networks.html#Btypes

17 https://dwave.wordpress.com/2011/05/27/teaching-artificial-intelligences-using-quantum-computers/

18 http://tmva.sourceforge.net/docu/TMVAUsersGuide.pdf

19 Figure 11.4 from “The Elements of Statistical Learning: Data Mining, Interference, and Prediction”, Hasti, Tibshirani, Friedman, 2009, Springer Series in Statistics

20 Pages 404-409 from “The Elements of Statistical Learning: Data Mining, Interference, and Prediction”, Hasti, Tibshirani, Friedman, 2009, Springer Series in Statistics

21 http://on-demand.gputechconf.com/gtc/2015/webinar/deep-learning-course/intro-to-deep-learning.pdf

22 http://codeforyou.info/wp-content/uploads/2014/11/chess-board-rules-chess_00418821jpg-awesome.jpg

23 http://www.sciencealert.com/australian-engineers-have-put-quantum-technology-in-a-silicon-chip-for-the-first-time

BACKUP

Decision Tree Learning

- Adaptive Boost
  - is particularly powerful on weak classifiers, like decision trees with small depth (i.e. layers of decision nodes)
  - solves problem of overtraining
  - “slow learning” can further improve the performance and is regulated by an exponent to the boosting weight: $\alpha \rightarrow \alpha^{\beta}$
- Boosting for regression trees
  - Technique needs to be modified
  - Redefined ‘loss-per-event’ definition that considers how far off the derived value is from the target value
  - all events need to be reweighted, according to Linear, Square or Exponential function

$$\mathit{Linear:} \quad L(k) = \frac{|y(k) - \hat{y}(k)|}{\max\limits_{\text{events } k'}(|y(k') - \hat{y}(k'|)} \,,$$

$$\mathit{Square:} \quad L(k) = \left[\frac{|y(k) - \hat{y}(k)|}{\max\limits_{\text{events } k'}(|y(k') - \hat{y}(k'|)}\right]^2 \,,$$

$$\mathit{Exponential:} \quad L(k) = 1 - \exp\left[-\frac{|y(k) - \hat{y}(k)|}{\max\limits_{\text{events } k'}(|y(k') - \hat{y}(k'|)}\right]$$