From Small Data to Big Data: The Evolution of AI techniques [2]

Snehanshu Saha, Ph.D

Center for AstroInformatics Modeling and Simulation (CAMS), Anuradha and Prashanth Palakurthi Centre for Artificial Intelligence Research (APPCAIR), BITS PILANI K K Birla Goa Campus, astrirg.org

Data/Papers: http://astrirg.org/projects.html
ML Blog: https://beginningwithml.wordpress.com/
Github: https://github.com/sahamath?tab=repositories
Youtube Lectures: https://tinyurl.com/yyn6e64q

[2] SAiDL Summer Symposium:: Machine Learning Nuances: Art or Science?

Highlights

A Fragmented Timeline:

- Pattern Recognition in Small Data (SDPR)
- The Elegance and the Masters
- Big Data (BDPR) : Changed Landscape?
- Porting SDPR-What did we miss?
- The Elegance can't be ignored
- A (mild ?) critique of Deep Learning in Big Data
- Re-emergence of (Methodological) elegance in BDPR

Data Science Applications

SDPR

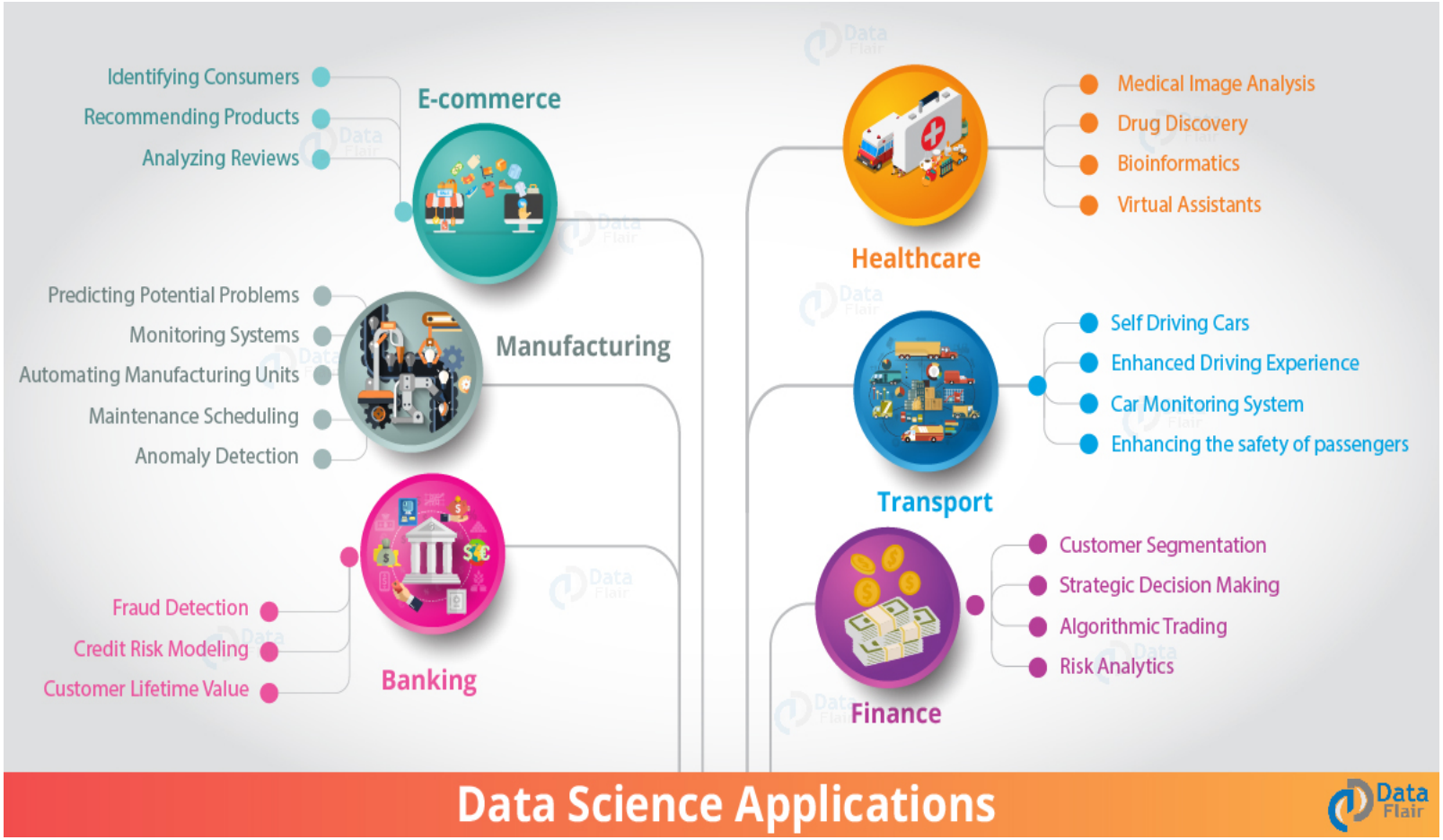

Figure: Data Science is not necessarily Big Data!

$$d(p,q) = \sqrt{(p_1 - q_1)2 + (p_2 - q_2)2}$$

Euclidean Distance

This formula may be extended to as many dimensions you want:

$$d(p,q) = \sqrt{(p_1 - q_1)2 + (p_2 - q_2)2 + \cdots + (p_n - q_n)2}$$

$$D^2 = (x - m)^T \cdot C^{-1} \cdot (x - m)$$

Mahalanobis Distance

where,

- D^2 is the square of the Mahalanobis distance.
- x is the vector of the observation (row in a dataset),
- m is the vector of mean values of independent variables (mean of each column),
- C^(-1) is the inverse covariance matrix of independent variables.

Figure: Euclid and Mahalanobis

India's first Data Scientist

Contribution and Application:

Figure: Prof. P.C Mahalanobis

Application

Removal of Statistical bias while comparing different dimensions, clustering: Customer segmentation in Retail business

Big Data Landscape

What's new:

Figure: V^3

Application

Volume, Veracity, Velocity: New Architectures as solutions?

AI in Big Data:

An Example

- Modern astronomical instruments record huge volumes of data in the form of images, catalogs, raw data and signals in different bandwidths.
- A new wave of pursuits in astronomy thus involve the use of **statistics**, **machine learning**, and **artificial intelligence** to solve problems. An emerging **interdisciplinary** area of study which calls for scientists to collaborate from the fields of:
  1. Astronomy and astrophysics
  2. Statistics
  3. Mathematics
  4. Computer and Information sciences.

Mentoring Information:

1. GALEX (The Galaxy Evolution Explorer) 30 TB
2. SDSS (The Sloan Digital Sky Survey) 40 TB
3. LSST (The Large Synoptic Survey Telescope) 200 PB expected

Big Data vs. Small Data

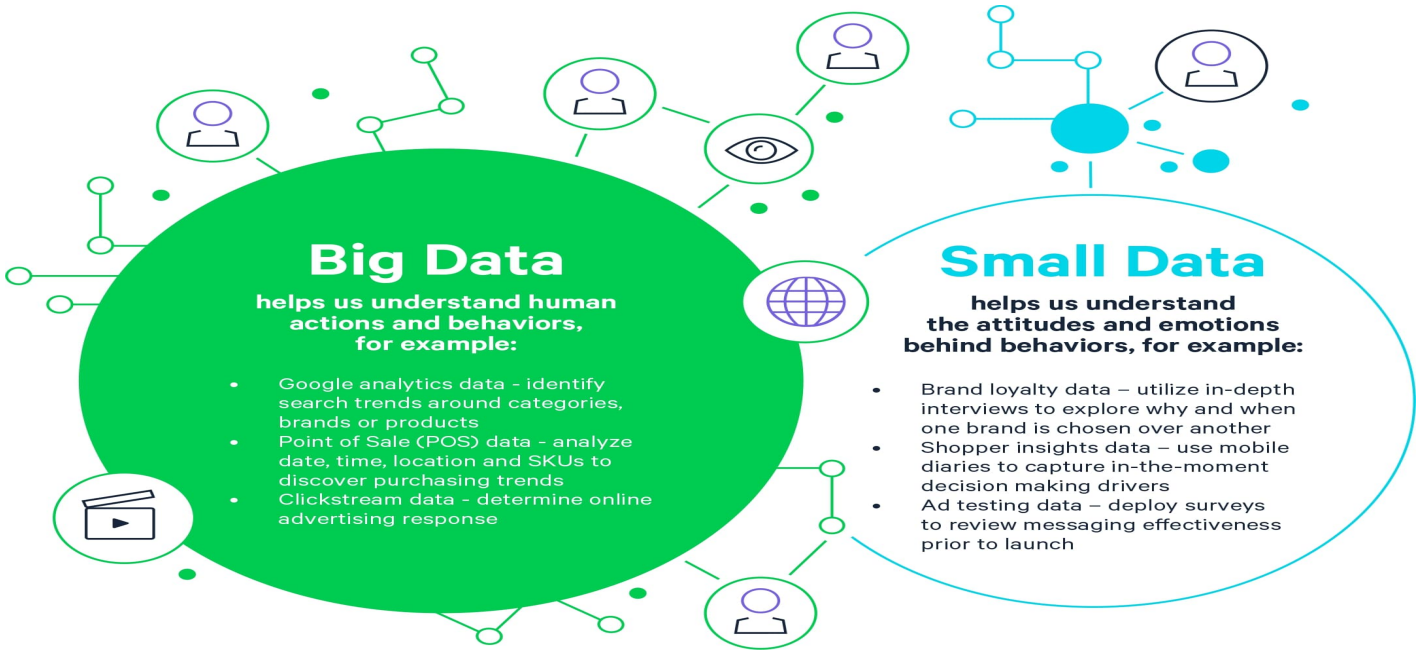

Figure: What Changes?

Are the methods still important? Yes!

- Over-reliance on Architecture necessary? No
- Is the elegance lost? No– LEARN FAST, NOT DEEP

The Really Deep Neural Nets

Society is about to experience an epidemic of false positives coming out of big-data projects-Prof. Michael Jordan, UC Berkeley

Figure: A certain fool's gold element to our current obsession

Deep NNs leading to false, meaningless inferences...

The cost

Computing Setup

The Really Deep Neural Nets

Master Node 2 x Intel(R) Xeon(R) CPU E5-2620 six-core 48 GB Memory (4GB per core) • 900GB SAS 10K x 4 Compute Node : 10 Qty • 2 x Intel(R) Xeon(R) CPU E5-2650 v2 eight-core • 64 GB Memory (4GB per core) • 600GB SAS 10K x 1 GPU Node : 2 Qty • Nvidia Tesla K20Xm cards • 2 x Intel(R) Xeon(R) CPU E5-2650 v2 eight-core • 128 GB Memory (8GB per core) • 900GB SAS 10K x 4

Neural networks, commonly known as Artificial Neural network(ANN), is a system of interconnected units organized in layers, which processes information signals by responding dynamically to inputs. Layers of the network are oriented in such a way that inputs are fed at input layer and output layer receives output after being processed at neurons of one or more hidden layers.

The Really Deep Neural Nets

Approximate Budget: 75 Lacs!

The good..

The Really Deep Neural Nets

Resnet 18; Densenet 40; Resnet50: The Computer Vision Community
CNN, RNN, Attention Models
LSTM for time series data

- MNIST
- CIFAR10
- CIFAR 100

The Tools

Scala, Hadoop, Julia

Neural nets are extremely intensive computationally and consume a lot of time and resources while training; we tolerate the *"high maintenance kid"* because the accuracy and other performance metrics are amazing, mostly. Plus, it handles non-linear problems reasonably well.

The bad..

Training Time

Since deep neural networks are function approximators, a large corpus of data is usually required to accomplish reasonable approximation of the error function, which is nothing but the difference between target and predicted labels. While computing resources are available in abundance, this problem didn't receive the attention it deserves, until recently.

- MIT Technology review, by Karen Hao
- The process of deep learning has an outsize environmental impact
- The NLP Model

Building and testing a final paper-worthy model in NLP required training 4,789 models over a six-month period. Converted to CO2 equivalent, it emitted more than 78,000 pounds. **the process can emit more than 626,000 pounds of carbon dioxide equivalent–approximately five times the lifetime emissions of the average American car**.

and the ugly..tiny image dataset

MIT apologizes, permanently pulls offline huge dataset that taught AI systems to use racist, misogynistic slurs!

- LARGE IMAGE DATASETS: A PYRRHIC WIN FOR COMPUTER VISION? Vinay Uday Prabhu & Abeba Birhane, July 1, 2020
- The Word-2-vec, Wordnet: Mothers are homemakers!

Your Big Data ML models are as good as data; We're just processing the volume efficiently!

The dataset holds more than 79,300,000 images, scraped from Google Images, arranged in 75,000-odd categories.

The key problem is that the dataset includes, for example, pictures of Black people and monkeys labeled with the N-word; women in bikinis, or holding their children, labeled w****; parts of the anatomy labeled with crude terms...

Back to another Master

and his elegance...SVM on **small data**

- Large Margin Classifier
- 1964-1992 (the timeline to finish a beauty)

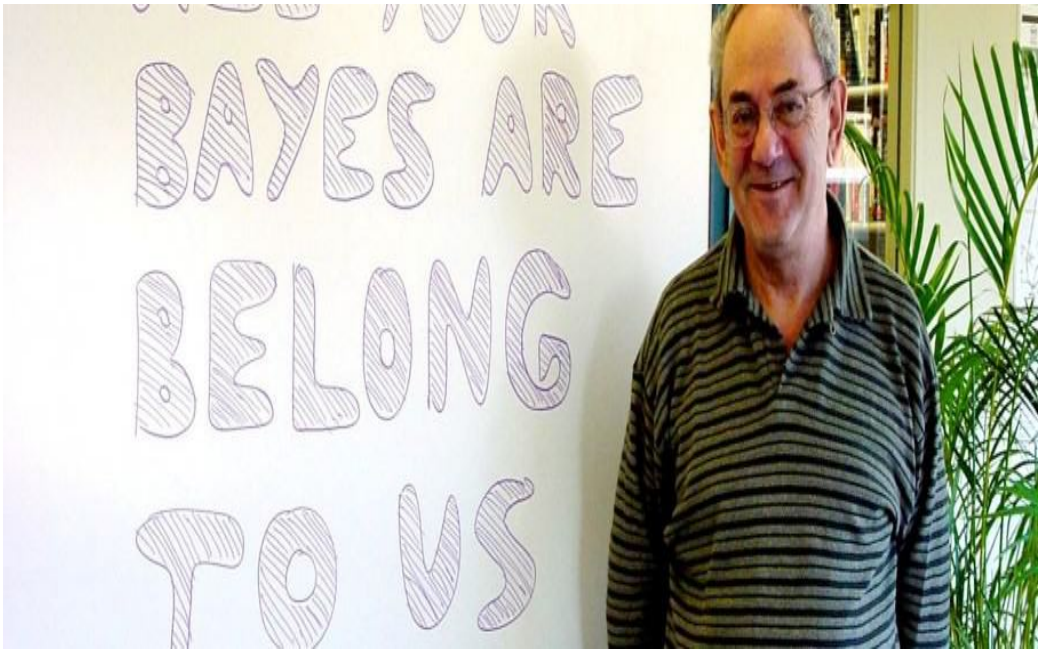

Figure: Prof. Vladimir Vapnik: A Beautiful Mind

1960-1990: renaissance in the field of statistics;
several ground breaking theories were built
new algorithms and philosophies on statistical learning theory

- Development of SVM coincides with the progress made during the evolution of learning theory
- Generalized Portrait Problem–1964
- SVM as large margin classifier- 1992
- SVM for non-linear data- 1995
- Masterclass- 1903

Figure: Mercer: The Kernel (1903) Guy

Figure: David Hilbert: The founding father

Vapnik & SVM

small data AI

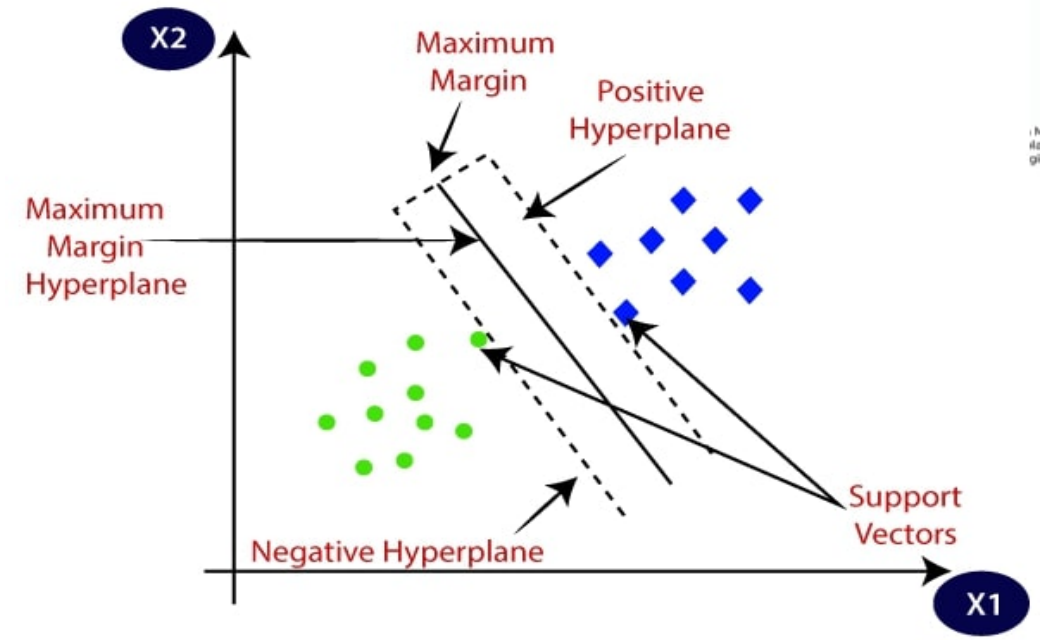

Figure: The 2-class Large Margin classifier

Optimization at the heart of Learning
Classification Problems are optimization problems

Linearly Separable SVM

small data AI

$Min \sum_{j=0}^{r} \sum_{k=0}^{r} a_j a_k < x_j, x_k >$ such that $y_j < w, x_j > \geq 1$
Note: The number of independent variables $(a_1, ...a_r)$ is equal to the size of the training data.

Separable training data (x_j, y_j) for linear SVM. Here $j = 1....r$ and
$y_j = 1$ if $x_j \in$ class 1
$y_j = -1$ if $x_j \in$ class 2
The margin w is given as $w = a_1 x_1 + ... + a_r x_r$ and the goal is find $a_1, a_2....a_r$ such that $\|w\|^2$ is maximized.

Non-Linearly Separable SVM

small data AI

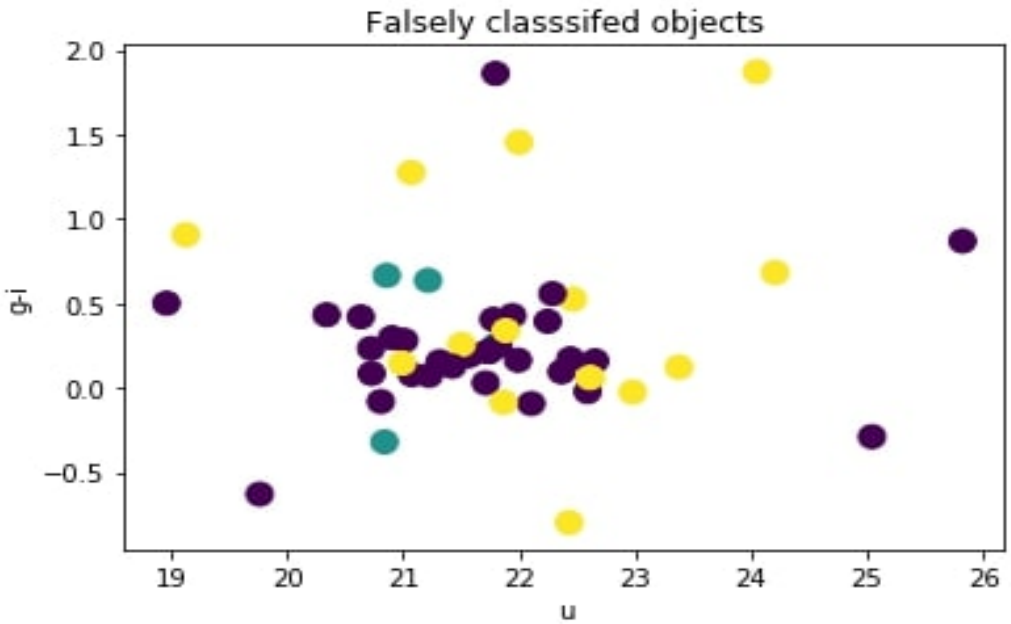

Figure: The 2-class classifier: Redshift can't be used as feature–Violet: stars, yellow: quasars and blue: galaxies

False positives; detected by redshift

Hilbert, Mercer and Vapnik

a 100 year span

Embed $x_1, ...x_r$ in a much higher dimensional vector space (i.e. from $w = w_1, ...w_{10}$ to $W = W_1, ...W_{100000000}$). The embedding space is required to be extremely large, theoretically speaking, it's infinite dimensional (**note here that Hilbert space is infinite dimensional**). We know that linear SVM depends on inner product of vectors and every vector is in an inner product space.

Solve the costly process via kernel (Mercer)

for a inner product to become a kernel, we need to satisfy symmetry and positive semi-definite (PSD). Hence we need to show that the kernel needs to be symmetric and PSD. This allows to define the embedding i.e. a mapping $\phi : \Re^n \rightarrow \Re^N$ where $N \gg n$ (N $= \infty$, theoretically)

The Elegant Trio: HMV

small data AI

Non-linear SVM under this embedding is written as
Minimize $\frac{1}{2} < w, w >$ s.t. $y_j < w_j, x_j > \geq 1$
$\Rightarrow$ Minimize $\frac{1}{2} < \phi(w), \phi(w) >$ s.t. $y_j < \phi(w_j), \phi(x_j) > \geq 1$
Define $K < w, x > = < \phi(w), \phi(x) >$ and thus the optimization problem can be written as
Minimize $\sum_{j=1}^{r} \sum_{k=1}^{r} a_j a_k K(x_j, x_k)$
subject to conditions $y_j \sum_{k=1}^{r} a_k K(x_j, x_k) \geq 1$
K is the kernel that replaces the embedding ϕ with the same properties as $< \phi(w), \phi(x) >$ for $N \gg n$ and must satisfies
$K < w, x > = K < x, w >$ and $K < w, w >> 0$

The beauty of small data inference: the solid inference techniques
Big data Learning could imbibe!

Kernel Embedding

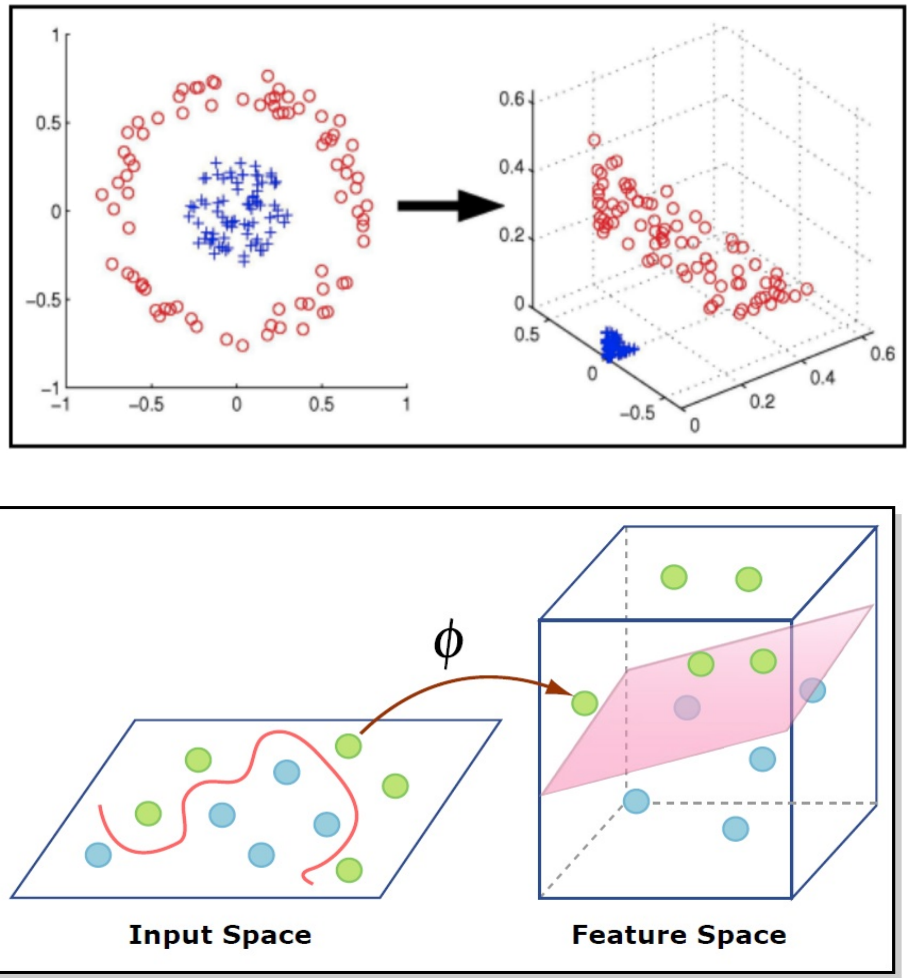

Figure: Non-linear decision boundary embedded to Linear via Mercer Theorem

Figure: A former student of mine training his DL model in NC State

- Optimize, optimize, optimize.....
- New methods in optimization for prediction error to hit rock bottom!
- Get rid of expensive architectures (too many hidden layers, too many neurons, too many weight updates)
- Single-shot learning

Tame the bandits i.e. hyperparameters

What we need

Advanced Calculus, Advanced Statistics, Linear Algebra
Need some geniuses and their theories: Hilbert, Banach, Cauchy, Lipschitz...
Functional Analysis, Convex optimization, Differential Equations, Chaos theory

What are the performance benchmarks we should be looking at

- Accuracy, loss etc.. (obvious)
- Epochs to converge to optima
- CPU/memory utilization
- Carbon footprint

What we should avoid?

- Inferences that make no sense
- Avoid ridiculous mistakes (redshift, Surf Temp estimation)
- ethical/social/racial misdemeanor

Optimization: Revisit the Neural classification

Devil's in the details: lr :: tune the bandits??

Optimise weights and biases by using back propagation on SBAF::
Initialize all weights w_{ij}**, biases** b_i**,** n_epochs**, lr, k and** α**;**

The forward Pass: Use appropriate function approximation

- ODE theory to the rescue.....for k, α

Adapative learning to the rescue.....for **lr**

"lr", the learning rate in weight update, recall?

back propagation & gradient descent: the squared loss function; Gradient Descent update Rule: REVISIT $\mathbf{w} := \mathbf{w} - \alpha \cdot \nabla_{\mathbf{w}} f$

- Use Lipschitz Constant on the squared loss function
- Use MVT
- **Minimal assumption:** functions are Lipschitz continuous and differentiable up to first order only [a]
- $\alpha = \frac{1}{L}$, we force $\Delta \mathbf{w} \leq 1$, constraining the change in the weights.

[a] Note this is a weaker condition than assuming the gradient of the function being Lipschitz continuous. We exploit merely the boundedness of the gradient.

What is it?

For a function, the Lipschitz constant is the least positive constant L such that $\|f(\mathbf{w}_1) - f(\mathbf{w}_2)\| \leq L \|\mathbf{w}_1 - \mathbf{w}_2\|$

$$\begin{aligned} f(\mathbf{w}_1) - f(\mathbf{w}_2) &= \nabla_{\mathbf{w}} f(\mathbf{v}) \mathbf{w}_1 - \mathbf{w}_2 \\ &\leq \sup_{\mathbf{v}} \nabla_{\mathbf{w}} f(\mathbf{v}) \mathbf{w}_1 - \mathbf{w}_2 \end{aligned}$$

Thus, $\sup_{\mathbf{v}} \nabla_{\mathbf{w}} f(\mathbf{v})$ is such an L. Since L is the least such constant, $L \leq \sup_{\mathbf{v}} \nabla_{\mathbf{w}} f(\mathbf{v})$. We use $\max \nabla_{\mathbf{w}} f$ to derive the Lipschitz constants.

LALR: knowing the "lr"

Minimal assumption: functions are Lipschitz continuous and differentiable up to first order only [a]

[a] Note this is a weaker condition than assuming the gradient of the function being Lipschitz continuous. We exploit merely the boundedness of the gradient.

$\alpha = \frac{1}{L}$, we force $\Delta \mathbf{w} \leq 1$, constraining the change in the weights. Stress: Not computing the Lipschitz constants of the *gradients* of the loss functions, but of the losses themselves. Assume: the loss is L-Lipschitz. Argument: set the learning rate to the reciprocal of the Lipschitz constant. Claim: supported by our experimental results.

Let the loss be given by $E(\mathbf{a}^{[L]}) = \frac{1}{2m}\left(\mathbf{a}^{[L]} - \mathbf{y}\right)^2$ *where the vectors contain the values for each training example.*

Note: Chain Rule in GD

$$\frac{\partial E}{\partial w_{ij}^{[L]}} = \frac{\partial E}{\partial a_j^{[L]}} \cdot \frac{\partial a_j^{[L]}}{\partial z_j^{[L]}} \cdot \frac{\partial z_j^{[L]}}{\partial w_{ij}^{[L]}}$$
$$= \frac{\partial E}{\partial a_j^{[L]}} \cdot \frac{\partial a_j^{[L]}}{\partial z_j^{[L]}} \cdot a_j^{[L-1]}$$

This gives us $\max_{i,j} \frac{\partial E}{\partial w_{ij}^{[L]}} = \max_j \frac{\partial E}{\partial a_j^{[L]}} \cdot \max_j \frac{\partial a_j^{[L]}}{\partial z_j^{[L]}} \cdot \max_j a_j^{[L-1]}$. The third part cannot be analytically computed; we denote it as K_z.

The formula!

Using the results above and other "tricks", we obtain:

$\max_{i,j} \frac{\partial E}{\partial w_{ij}^{[L]}} = \frac{K}{m}\mathbf{X}^T\mathbf{X} - \frac{1}{m}\mathbf{y}^T\mathbf{X}$; K is the upper bound on the weight vectors

Binary Classification-binary cross entropy loss: $\boxed{L = \frac{1}{2m}\mathbf{X}}$

Multiclass Classification-softmax regression loss: $\boxed{L = \frac{k-1}{km}\mathbf{X}}$

- L is lipschitz constant; **lr**, learning rate $= \frac{1}{L}$

Such choice of Learning Rate?

Assumption: Gradients cannot change arbitrarily fast

A typical Gradient Descent algorithm is in the form of

$$\boldsymbol{w^{k+1}} = \boldsymbol{w^k} - \eta_k \nabla_w f \tag{1}$$

for $k \in \mathbb{Z}$ and stop if $\|\nabla_w f\| \leq \epsilon$
$\forall v, w, \exists L$ such that $\|f(\mathbf{v}) - f(\mathbf{w})\| \leq L \|\mathbf{v} - \mathbf{w}\|$. Also, $\nabla^2 f(w) \leq LI$

$$\boldsymbol{f(v)} = \boldsymbol{f(w)} + \nabla \boldsymbol{f(w)^T(v-w)} + \frac{1}{2}(\boldsymbol{v-w})^T \nabla^2 \boldsymbol{f(w)(v-w)}$$

$$\boldsymbol{f(v)} \leq \boldsymbol{f(w)} + \nabla \boldsymbol{f(w)^T(v-w)} + \frac{L}{2}\|\boldsymbol{v-w}\|^2 \tag{2}$$

- a convex quadratic upper bound which can be minimized using gradient descent with the learning rate $\eta_k = \frac{1}{L}$

Convergence

It follows, $f(w^{k+1}) \leq f(w^k) - \frac{1}{2L} \left\| \nabla f(w^k) \right\|^2$

Gradient Descent decreases $f(w)$ if $\eta_k = \frac{1}{L}$ and $\eta_k < \frac{2}{L}$. This proof also enables one to derive the rate of convergence with Lipschitz adaptive learning rate.

Let n represent the number of iterations. We know that,

$$f(w^{k+1}) \leq f(w^k) - \frac{1}{2L} \left\| \nabla f(w^k) \right\|^2$$

Several steps after.... $n \geq \frac{2L(f(w^0) - f(w^*))}{\epsilon}$

Gradient Descent requires, at least, $n = O(\frac{1}{\epsilon})$ iterations to achieve the error bound: $\left\| \nabla f(w^k) \right\| \leq \epsilon$

Loss Function, NOT Activation

- For a neural network that uses the sigmoid, ReLU, or softmax activations, it is easily shown that the gradients get smaller towards the earlier layers in backpropagation. Because of this, the gradients at the last layer are the maximum among all the gradients computed during backpropagation.
- Our framework is independent of activation functions
- the framework is extensible to all loss function satisfying the *lipschitz condition*
- Thus, we set up a pipeline for classification problems-SYMNet.

- Economic theory is consistent with Our models: A new SVM Kernel and NN training (Activation functions derived from Lip Conditions on 1st Order DEs)
- We found the *suitable boy* (borrowing from Vikram Seth) LR $= 1/L$ and ..... implies not a **bandit** anymore!.

Progression in parameter handling: from tuning to selection

What's the big deal? Immensely cheaper computation! remarkable prediction performance!!

Recap: The beast conquered!

- Theoretical framework for computing an adaptive learning rate; this is also "adaptive" with respect to the data.
- "large" learning rates may not be harmful as once thought; remedy: guarded value of L_2 weight decay.

What did we gain?

- Novel Approximation functions used during forward pass with "little effort in parameter tuning, k, α; OUTCOME: Accuracy beating state-of-the-art
- Loss Function manipulation to devise learning rate formula; OUTCOME: NO Need to "handcraft" **lr**; performance as good as state of the art
- Metric gains: Time to converge, # of iterations to converge, CPU/RAM utilization

A pipeline with SBAF in forward pass + Adaptive **lr** in GD back-prop:
SYMNET: Released v.1.0 ..https://github.com/sahamath/sym-netv1

Takeaway

So what have these new kids on the block done?

- saves number of iterations to converge
- accuracy, precision, recall and other performance metrics are better, at least on the data sets applied so far (13 different public data sets)
- Improvement in CPU and RAM utilization
- non-negligible improvement in time complexity

Parsimonious computing; *something I always wanted to do*

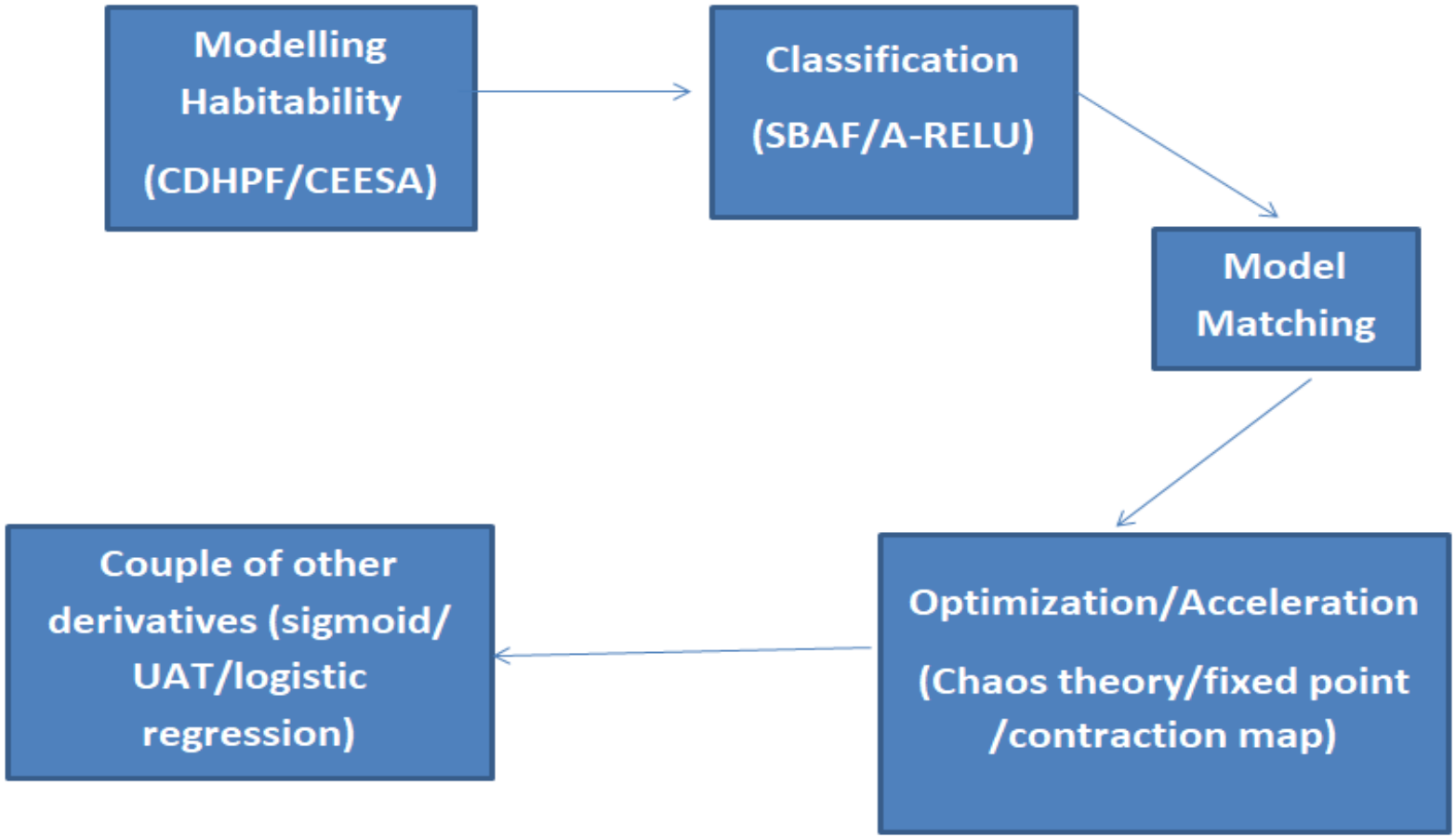

Figure: The Elegance to connect the dots

The classics

- Vladimir Vapnik, Estimation of Dependences Based on Empirical Data
- Vapnik et. al, Support Vector Regression Machines
- Ying Tan, and Jun Wang, Support Vector Machine with a Hybrid Kernel and Minimal Vapnik-Chervonenkis Dimension
- Mikhail Belkina et. al, Reconciling modern machine learning practice and the bias-variance trade-off
- VAPNIK et. al, Support-Vector Networks
- J. Mercer, Functions of Positive and Negative Type, and their Connection with the Theory of Integral Equations
- Vapnik et. al, On a class of perceptrons
- Sebastian Bock et. al, An improvement of the convergence proof of the ADAM-Optimizer
- ZUOWEI SHEN et. al, NONLINEAR APPROXIMATION VIA COMPOSITIONS

Publications to read

- Hrushikesh Mhaskar et. al, When and Why Are Deep Networks Better than Shallow Ones?
- Preetum Nakkiran et. al, Compressing Deep Neural Networks using a Rank-Constrained Topology

Classic Texts in ML

Bishop, Duda & Hart, Tibshirani, Shalev et. al

I don't recommend Goodfellow's Deep Learning book

I recommend this one: Linear Algebra and Optimization in Machine Learning by Charu C. Aggarwal

Mining of Massive Datasets: Jure Leskovec, Anand Rajaraman, Jeffrey D. Ullman

Robert Gallagher's take on Shannon's style of research: http://www.ifp.illinois.edu/ tieliu/Shannon.html

THANK YOU!

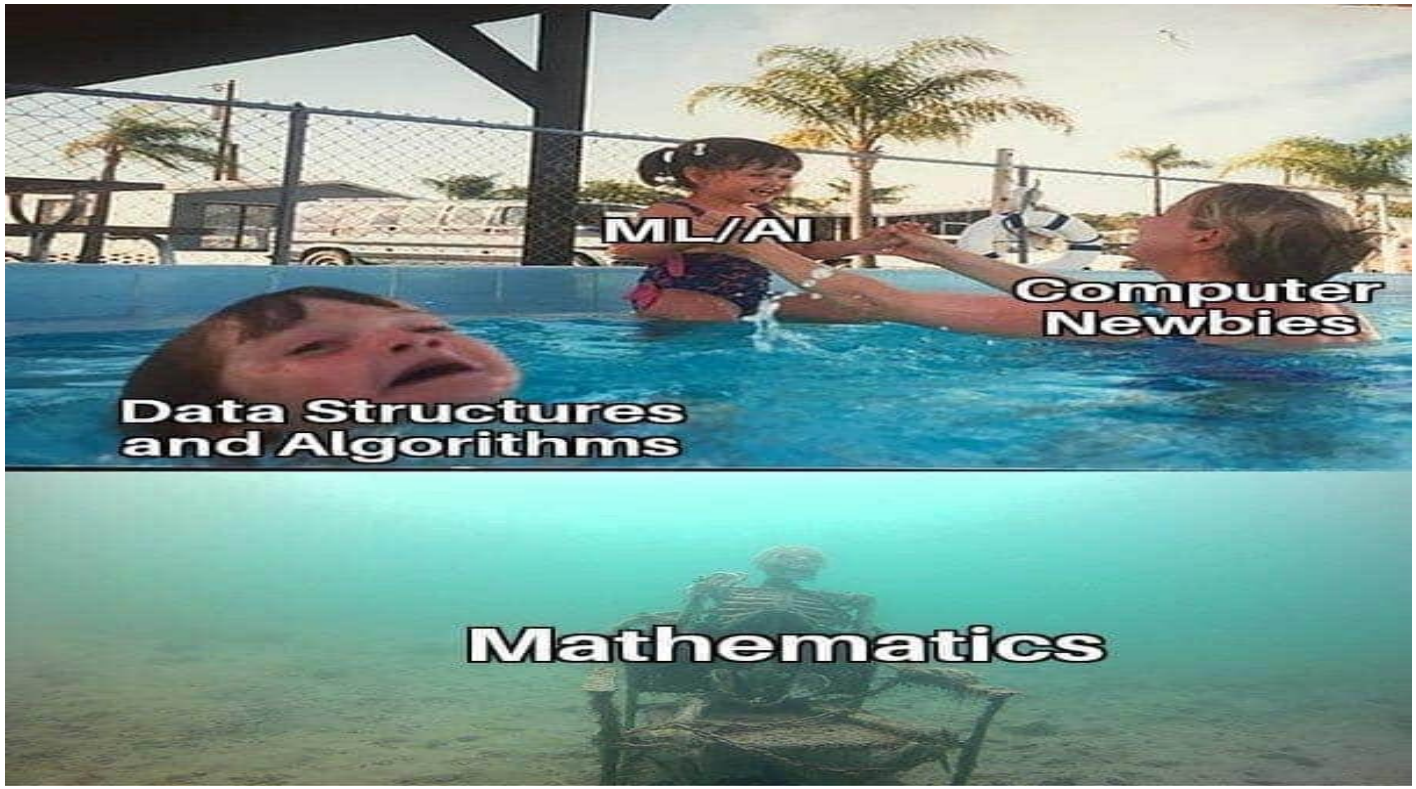

Figure: No Comment

I gratefully acknowledge multiple online sources for images used in the presentation.