# Feature Engineering in Machine Learning

**Research** · September 2015

DOI: 10.13140/RG.2.1.3564.3367



Nayyar Abbas Zaidi
Monash University (Australia)

# Feature Engineering in Machine Learning

Nayyar A. Zaidi

Research Fellow
Faculty of Information Technology,
Monash University, Melbourne VIC 3800, Australia

August 21, 2015

# Outline

- A Machine Learning Primer
  - Machine Learning and Data Science
  - Bias-Variance Phenomenon
  - Regularization
- What is Feature Engineering (FE)?
- Graphical Models and Bayesian Networks
- Deep Learning and FE
- Dimensionality Reduction
- Wrap-up
  - Current Trends
  - Practical Advice on FE

# Machine Learning

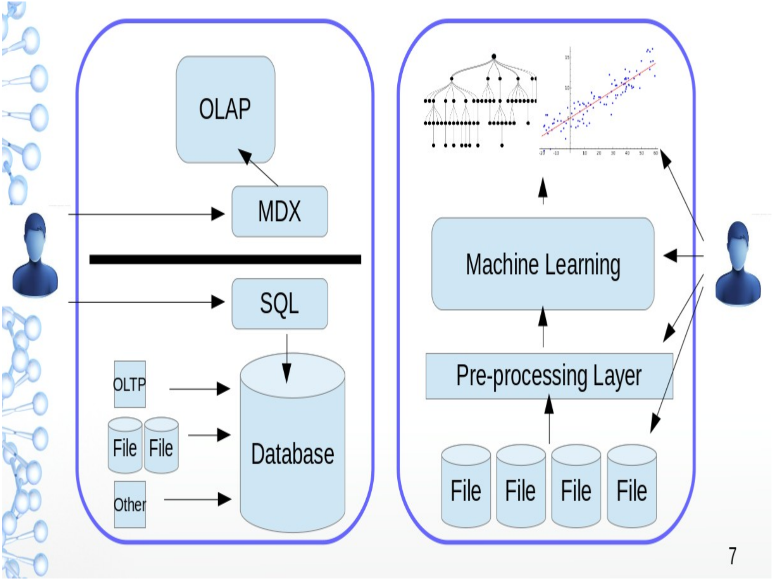
- Suppose there exists a function y = f(x), now given examples of the form (y,x), can we determine the function f? [1]
  - Functional approximation
  - Input is the Data
  - Roots in Statistics
  - Kin to Data Mining
  - Subset of Data Science
- Countless applications in:
  - Medicine, Science, Finance, Industry, etc.
- Renewed interests due to the emergence of big data
- Part of multi-billion analytics industry of 21st century

# Typical Machine Learning Problems

- Supervised Learning (Classification, Regression)
- Un-supervised Learning (Clustering)
- Recommender Systems
- Market Basket Analysis (Association Rule Discovery)
- Ad-placement
- Link Analysis
- Text Analysis (e.g., mining, retrieval)
- Social Network Analysis
- Natural Language Processing

# Applications of Machine Learning

# Tale of Two Worlds

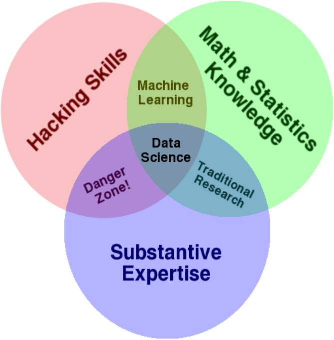


# Data Science

- The two worlds are merging into each other day by day
- Database community needs analytics and analytics community needs a way to store and manage large quantities of data
- On-going debate about putting databases into analytics or analytics into databases
- SQL vs. NoSQL
- Database world
  - Pros: Good at storing, accessing and managing large quantities of data
  - Cons: Very bad for analytics (assumes a structure)
- Analytics world
  - Pros:Good at analyzing
  - Cons: Poor at managing data

# Data Science

- What constitutes data science:
  - Analytics
  - Storage
  - Visualization
  - Munging



# Data-driven World

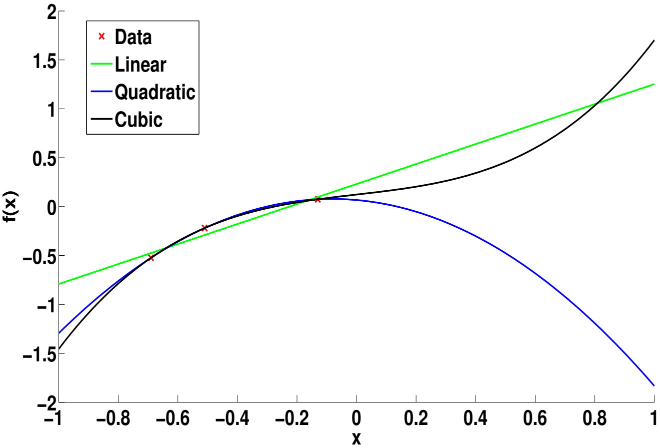
- Four paradigms of science
  - Observational based
  - Experimental based
  - Model based
  - Data based
- "The Fourth Paradigm: Data-Intensive Scientific Discovery", by Jim Gray
- It is not about databases vs. analytics, SQL vs. NoSQL, it is all about data

# Fundamental Problem in Machine Learning

- Regression:
  - $\min_{\beta \in \mathcal{R}^n} \frac{1}{N} \sum_{i=1}^{N} (y_i - \beta^T x)^2 + \lambda \|\beta\|_2^2$
- Classification:
  - $\min_{\beta \in \mathcal{R}^n} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(y_i \beta^T x) + \lambda \|\beta\|_2^2$
- In general:
  - $\min_{\beta \in \mathcal{R}^n} (\mathrm{Loss} + \mathrm{Regularization})$
  - $\min_{\beta \in \mathcal{R}^n} \mathcal{F}(\beta)$
- Important questions:
  - Model selection.
  - Which optimization to use to learn the parameters.
- Different loss functions leads to different classifiers:
  - Logistic Regression
  - Support Vector Classifier
  - Artificial Neural Network

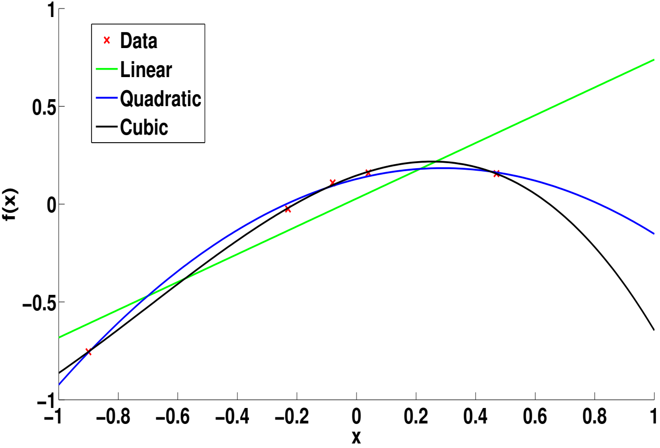
# Model Selection

- Let us visit the simplest of all machine learning problem:



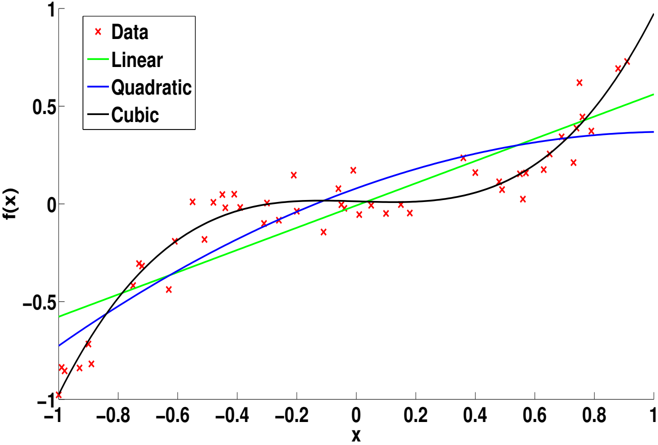
# Model Selection

- Let us visit the simplest of all machine learning problem:



# Model Selection

- Let us visit the simplest of all machine learning problem:



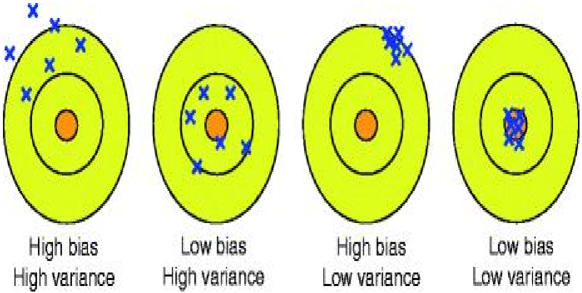
# Model Selection

- Observation – Models vs. Features
  - You can take the cube of the features and fit a linear model.
    - $(x_1, \ldots, x_m) \rightarrow (x_1^3, x_1^2 x_2, x_1 x_2 x_3, \ldots)$
  - This will be equivalent to applying cubic model.
- Question – Which model should you select? Or equivalently, which attributes interactions should you consider?
- Remember – In real world, you will have more than one features - categorical, discrete, etc.
- Hint – With every model selection decision - there is a control of bias and variance:
  - Why not select model by controlling for both bias and variance?

# Parameterizing Model

- How do you handle numeric attributes? One parameter per attribute per class?
- How do you handle categorical attributes? Multiple parameters per attribute per class?
- How do you handle interactions among the variables?
- How do you handle missing values of the attributes?
- How do you handle redundant attributes?
- Over-parameterized model vs. under-parameterized model

# Bias Variance Illustration



# Model Selection

- Low variance model for small data
- Low bias model for big data
- More details in [2]
- Machine learning has been applied on small quantities of data
  - Here, low variance algorithms are the best
  - Low bias algorithms will over-fit and you need to rely on regularization or feature selection
- Feature Selection
  - Forward selection
  - Backward elimination

# Regularizing your Model

- Very powerful technique for controlling bias and variance
- Many different regularizations exists, most common:
  - L2 Regularization
    - $\min_{\beta \in \mathcal{R}^n} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(y_i \beta^T x) + \lambda \|\beta\|_2^2$
  - L1 Regularization (also know as sparsity inducing norms)
    - $\min_{\beta \in \mathcal{R}^n} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(y_i \beta^T x) + \lambda \|\beta\|_1$
  - Or elastic nets
    - $\min_{\beta \in \mathcal{R}^n} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(y_i \beta^T x) + \lambda (\|\beta\|_1 + \gamma \|\beta\|_2^2)$

# Feature Engineering

- Feature engineering is the process of using domain knowledge of the data to create features that make machine learning algorithms work.
- It is fundamental to the application of machine learning, and is both difficult and expensive.
- The need of manual feature engineering can be obviated by automated feature learning.
    - Wikipedia
- Coming up with features is difficult, time-consuming, requires expert knowledge. Applied machine learning is basically feature engineering.
    - Andrew Ng
- Is this what feature engineering is?

# Feature Engineering (Contd)

- Feature Engineering is the next buzz word after big data.
- But. On the basis of Wikipedia definition, one can say that feature engineering has been going on for decades.
  - Why so much attention now?
- In my view – Feature Engineering and Big Data are related concepts.
- For big data, you need big models
  - Big models – Any model with very large no. of parameters.
  - Note that a big model can be simply linear.
- For big models, it is more of an engineering problem as to how handle these parameters effectively.
  - Since the hardware has not scaled-up well with data.

# Feature Engineering (Contd)

- Given a model, learning problem is the learning of the parameters of model.
- The number of parameters depends on the number of features that you have:
  - Exception, if you are solving for the dual.
  - There will be some hyper-parameters.
- Parameter estimation is done by optimizing some objective function.
- Traditionally, there has been four elements of interest:
  - Features
  - Model
  - Objective function
  - Optimization

# Feature Engineering (Contd)

- Models and features are related.
  - Let us not worry about that for the time being.
- There have been two main objective functions that is:
  - Conditional Log-Likelihood – $P(y|x)$.
  - Log-Likelihood – $P(y, x)$.
- This distinction has led to generative-discriminative paradigms in machine learning.
  - Generative models $P(y, x)$.
  - Discriminative models $P(y|x)$.
  - Very confusing distinction with no obvious benefits.

# Generative vs. Discriminative Models/Learning

- Bayes rule: $P(y|x) \propto P(y, x)$
- Converting $\propto$ to $=$, we get: $P(y|x) = \frac{P(y,x)}{P(x)}$.
- And therefore, $P(y|x) = \frac{P(y,x)}{\sum_c P(c,x)}$.

# Generative vs. Discriminative Models/Learning

- A well-known generative model – Naive Bayes

$$P(y|x) = \frac{\pi_y \prod_i P(x_i|y)}{\sum_c \pi_c \prod_i P(x_i|c)} \tag{1}$$

- A well-known discriminative model – Logistic Regression

$$P(y|x) = \frac{\exp(\beta_y + \sum_i \beta_{i,x_i,y} x_i)}{\sum_c \exp(\beta_c + \sum_i \beta_{i,x_i,c} x_i)} \tag{2}$$

# On the Equivalence of Generative vs. Discriminative Models/Learning

- We have naive Bayes as:

$$P(y|x) = \exp(log \pi_y + \sum_i \log P(x_i|y) - log(\sum_c \pi_c \prod_i P(x_i|c)))$$

- Same exp and log trick:

$$\begin{aligned} P(y|x) &= \exp(\log \pi_y + \sum_i \log P(x_i|y) - \\ &\quad log(\sum_c \exp(\log \pi_c + \sum_i \log P(x_i|c)))) \end{aligned}$$

$$\begin{aligned} \log P(y|x) &= \log \pi_y + \sum_i \log P(x_i|y) - \\ &\quad log(\sum_c \exp(\log \pi_c + \sum_i \log P(x_i|c))). \end{aligned}$$

# On the Equivalence of Generative vs. Discriminative Models/Learning

- Now, let us take the log of LR:
  $\log P(y|x) = \beta_y + \sum_i \beta_{i,x_i,y} x_i - \log(\sum_c \exp(\beta_c + \sum_i \beta_{i,x_i,c} x_i))$
- Reminder, for NB we had:

$$\begin{aligned}\log P(y|x) &= \log \pi_y + \sum_i \log P(x_i|y) - \\ & log(\sum_c \exp(\log \pi_c + \sum_i \log P(x_i|c))).\end{aligned}$$

- NB and LR are just re-parameterization of each other for example: $\beta_y = \log \pi_y$ and $\beta_{i,x_i,y} = \log P(x_i|y)$.

# On the Equivalence of Generative vs. Discriminative Models/Learning

- Modifying NB:

$$\log P(y|x) = w_y \log \pi_y + \sum_i w_{x_i|y} \log P(x_i|y) - log(\sum_c \exp(w_c \log \pi_c + \sum_i w_{x_i|c} \log P(x_i|c))).$$

- This leads to:

$$P(y|x) = \frac{\pi_y^{w_y} \prod_i P(x_i|y)^{w_{x_i|y}}}{\sum_c \pi_c^{w_c} \prod_i P(x_i|c)^{w_{x_i|c}}}$$

- NB and LR are just re-parameterization of each other, so why the fuss?
- More details in [3, 4, 5]

# Summary so far

- Model selection and feature selection are tightly coupled.
  - Feature Engineering
- The distinction between CLL and LL is confusing. Rather, superfluous.
- They only differ in the way parameters are being learned:
  - For LL (naive Bayes), parameters are empirical estimates of probability
  - For CLL (LR), parameters are learned by iterative optimization of some soft-max.
- This leaves two things:
  - How to do feature engineering?
  - How to actually optimize?

# How to Engineer Features?

- Regularization
- Use domain knowledge
- Exploit existing structure in the data
- Dimensionality reduction
- Use data to build features

# Domain Knowledge

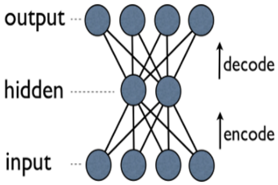
- Use of expert knowledge or your knowledge about the problem
- Use of other datasets to explain your data
- Main advantage is the simplicity and intuitiveness
- Only applies to small number of features
- Access to domain expert might be difficult
- Summary – You use some information at your disposal to build features before starting a learning process

# Dimensionality Reduction

- Feature Selection
  - Filter
  - Wrapper
  - Embedded
- Principle Component Analysis (PCA)
- Linear Discriminant Analysis (LDA)
- Metric Learning [6, 7]
- Auto-encoders

# Auto-Encoders

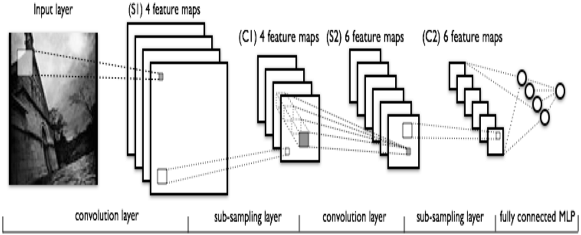
- Similar to multi-layer Perceptron
- Output layer has equally many nodes as input layer
- Trained to reconstruct its own input
- Learning algorithm: Feed-forward back propagation
- Structure:



# Exploit Structures in the Data

- Some datasets have an inherent structure, e.g., in computer vision, NLP
  - You can use this information to build features
- Convolutional Neural Networks
  - CNNs exploit spatially-local correlation by enforcing a local connectivity pattern between neurons of adjacent layers
  - Feed-forward neural network
  - Very successful in digit and object recognition (e.g., MNIST, CIFAR, etc.)

# Convolutional Neural Networks



# Use Data to Build Features

- Restricted Boltzmann Machines
  - Trained by contrastive divergence
- Deep Belief Networks
- Many more variants

# Lessons Learned from Big Data

- Automatic feature engineering helps
- Capturing higher-order interactions in the data is beneficial
- Low-bias algorithms with some regularization on big data leads to state-of-the-art results
- If you know the structure, leverage it to build (engineer) features
- What if you don't know the structure
  - Use heuristics

# Optimization

- Let us focus on the optimization:

$$w_{t+1} = w_t - \eta \frac{\partial \mathrm{OF}(w)}{\partial w} \tag{3}$$

- Going second-order:

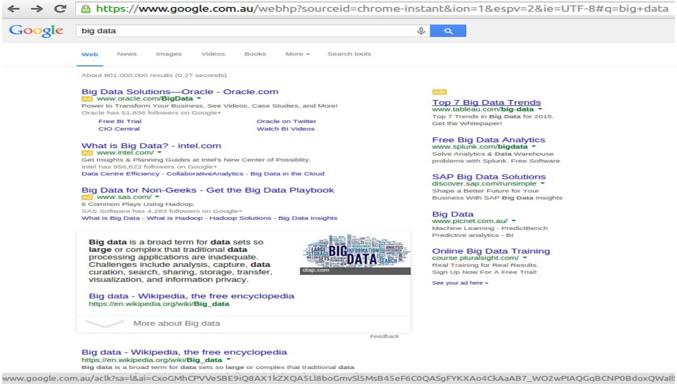
$$w_{t+1} = w_t - \eta \frac{\partial^2 \mathrm{OF}(w)}{\partial w} \tag{4}$$

- Place regularization to make function smooth for second-order differentiation

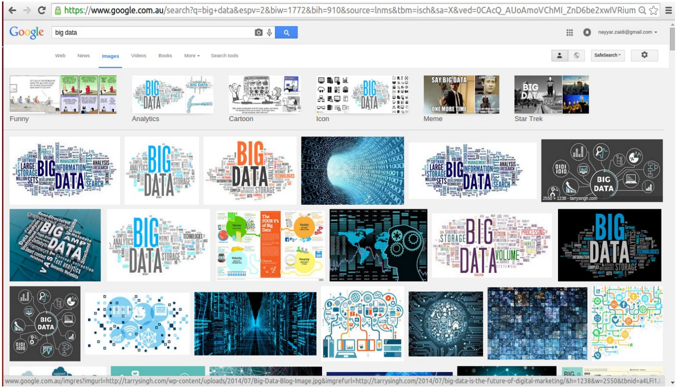
# Implementation issues in Optimization

- Can we leverage map-reduce paramdigm? No
- Massive Parallelization is required to handle parameters
- Batch version – you can distribute parameters across nodes
- SGD – you can rely on mini-batches
- Success in deep learning is attributed to advancements in optimization and implementation techniques

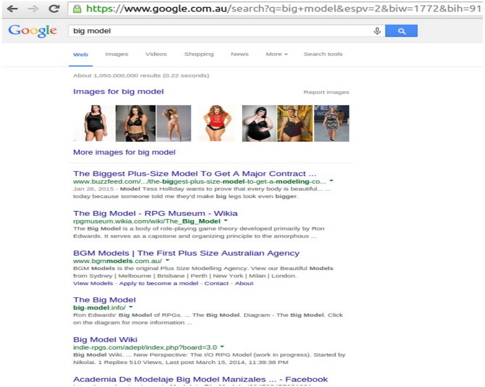
# Big Data vs. Big Models



# Big Data vs. Big Models



# Big Data vs. Big Models



# Big Data vs. Big Models

- Big Data:
    - Lot of hype around it
    - Volume, Velocity, Variety, Varsity
    - Lots of efforts for managing big data
- Big Models:
    - Models that can learn from big data
    - For example: [8, 9]
    - Deep and Broad
    - Ingredients: Feature Engineering + Stable Optimization
- It is the big models that hold key to a break-through than big data.

# Conclusion and Take-away-home Message

- Two most important things: Feature Engineering + Optimization
- Big Model not big data
- Big Model: Low bias + Out-of-core + Minimal tuning parameters + Multi-class
- Big Model: Deep + Broad
- Questions

- R. Duda, P. Hart, and D. Stork, *Pattern Classification*. John Wiley and Sons, 2006.
- D. Brain and G. I. Webb, "The need for low bias algorithms in classification learning from small data sets," in *PKDD*, pp. 62–73, 2002.
- T. Jebara, *Machine Learning: Discriminative and Generative*. Springer International Series, 2003.
- N. A. Zaidi, J. Cerquides, M. J. Carman, and G. I. Webb, "Alleviating naive Bayes attribute independence assumption by attribute weighting," *Journal of Machine Learning Research*, vol. 14, pp. 1947–1988, 2013.
- N. A. Zaidi, M. J. Carman, J. Cerquides, and G. I. Webb, "Naive-bayes inspired effective pre-conditioners for speeding-up logistic regression," in *IEEE International Conference on Data Mining*, 2014.

- N. Zaidi and D. M. Squire, "Local adaptive svm for object recognition," in *Digital Image Computing: Techniques and Applications (DICTA), 2010 International Conference on*, 2010.
- N. Zaidi and D. M. Squire, "A gradient-based metric learning algorithm for k-nn classifiers," in *AI 2010: Advances in Artificial Intelligence*, 2010.
- N. A. Zaidi and G. I. Webb, "Fast and effective single pass bayesian learning," in *Advances in Knowledge Discovery and Data Mining*, 2013.
- S. Martinez, A. Chen, G. I. Webb, and N. A. Zaidi, "Scalable learning of bayesian network classifiers," *accepted to be published in Journal of Machine Learning Research*.