An Introduction to Machine Learning

Research Group Meeting 2017

Melissa R. McGuirl

February 22, 2017

Table of Contents

1. Introduction
2. Supervised Learning
   - Generalized Linear Models
   - Support Vector Machines
   - Decision Trees
3. Unsupervised Learning
   - Manifold learning
   - Clustering
   - Neural Networks
4. Model Selection and Evaluation
5. Applications
6. Pitfalls to avoid
7. Machine Learning that Matters
8. Bibliography

General Set-up

Set-up and Goal

Suppose we have $X_1, X_2, \ldots, X_n$ data samples. Can we predict properites about any given $X_{n+1}, X_{n+2}, \ldots, X_N$?

Machine learning systems attempt to predict properties of unknown data based on the attributes or features of the data.

Supervised learning

1. Data comes with attributes that we want our algorithm to predict
2. Machine learning algorithm is given data attributes and desired outputs and the goal is to learn a way to map inputs to outputs in a general way
3. 2 main types of learning:
    1. **Classification**: Data belongs to different classes/groups and we want to be able to predict which class/group unlabeled data belongs to
    2. **Regression:** Data labeled with one or more continuous variables (parameters) and the task is to predict the value of these variables for unknown data

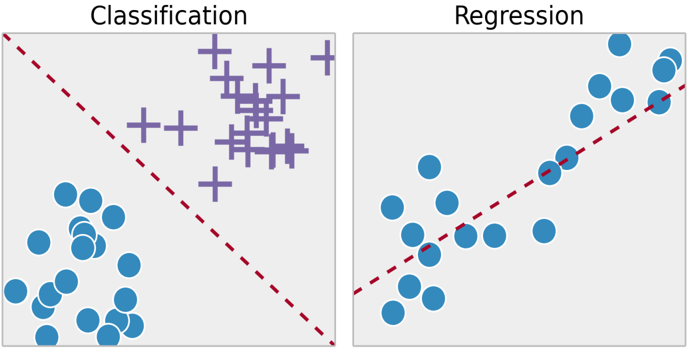


Image source: `http://ipython-books.github.io/featured-04/`

Unsupervised Learning

1. No labels are given to the algorithm
2. Training data consists of a set of input vectors $\{X_1, X_2, \ldots, X_n\}$ with no target outputs
3. 3 Types of goals in this setting:
    1. **Clustering**: discover groups with similar features within the data
    2. **Density Estimation:** determine distribution of data within the input space
    3. **Dimensionality Reduction:** project data into a lower dimensional space than input space

Generalized Linear Models

Set-up: Data in the form of $X_i = (x_1^i, x_2^i, \ldots, x_p^i)$

Goal: Regression

Find $\hat{y}$

$$\hat{y}(w, X_i) = w_0 + w_1 x_1^i + w_2 x_2^i + \ldots w_p x_p^i,$$

where w_0 is called the *intercept* and $w_1, \ldots w_p$ are the *coefficients*.

GLM: Ordinary Least Squares

- Fits linear model $\hat{y}(w, X_i) = w_0 + w_1 x_1^i + w_2 x_2^i + \ldots w_p x_p^i$ where $w = (w_1, ..., w_p)$ coefficients obtained by solving

$$\min_w ||Xw - y||_2^2$$

- Model relies on independence of model terms
- If terms are correlated then the least-square estimate is highly sensitive to random errors in the observed response (large variance)
- Complexity: $O(np^2)$

GLM: Ridge Regression

- Fits linear model $\hat{y}(w, X_i) = w_0 + w_1 x_1^i + w_2 x_2^i + \ldots w_p x_p^i$ where $w = (w_1, ..., w_p)$ coefficients obtained by solving

$$\min_w ||Xw - y||_2^2 + \alpha ||w||_2^2$$

- $\alpha > 0$ is a complexity parameter that controls how robust the coefficients are to linearity
- Penalty on the size of coefficients addresses issues with ordinary least squares method
- Complexity: $O(np^2)$

GLM: Lasso

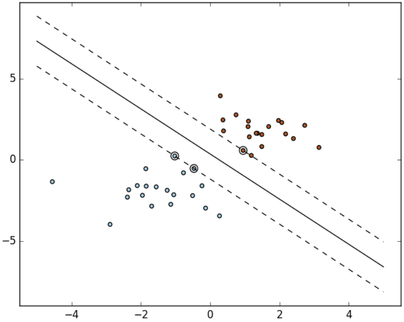
- Fits linear model $\hat{y}(w, X_i) = w_0 + w_1 x_1^i + w_2 x_2^i + \ldots w_p x_p^i$ where $w = (w_1, ..., w_p)$ coefficients obtained by solving

$$\min_{w} \frac{1}{2n} ||Xw - y||_2^2 + \alpha ||w||_1^1$$

- l_1 norm means we will this linear model estimates sparse coefficients
- This method reduces number of variables upon which the solution is dependent
- Useful in compressed sensing and can be used for feature selection

SVM: Mathematical Setup

A SVM constructs a hyperplane (or set of hyperplanes) in a high or infinite dimensional space, which can be used for classification, regression or other tasks.



SVM: Linear, separable case

We want to find the hyperplane that maximizes the margin as follows:

Mathematical formulation

Given $X_1, X_2, \dots X_n \in \mathbb{R}^p$, with labels $Y_i \in [-1, 1]$, minimize $||w||^2$ subject to

$$\begin{cases} (w \cdot X_i + b) \geq 1 & Y_i = 1 \\ (w \cdot X_i + b) \leq -1 & Y_i = -1 \end{cases}$$

Equivalently,

$$Y_i(w \cdot X_i + b) \geq 1$$

Decision function:

$$f(X, w, b) = \text{sign}(w \cdot X + b)$$

SVM: Linear, non-separable case

Rewrite problem as

Mathematical formulation

Given $X_1, X_2, \dots X_n \in \mathbb{R}^p$, with labels $Y_i \in [-1, 1]$, minimize $||w||^2 + C\sum_{i=1}^{n} \xi_i$ subject to

$$Y_i(w \cdot X_i + b) \geq 1 - \xi_i, \;\; \xi_i \geq 0$$

This added a hinge-loss term.

General

Decision: $f(X) = w \cdot X + b$
Solve:

$$\min P(w,b) = \frac{1}{2}||w||^2 + C\sum_i H_1[Y_i f(X_i)]$$

$$= \text{maximize margin} + \text{minimize error}$$

SVM: Linear, non-separable case

Rewrite problem as

Mathematical formulation

Given $X_1, X_2, \dots X_n \in \mathbb{R}^p$, with labels $Y_i \in [-1,1]$, minimize $||w||^2 + C\sum_{i=1}^{n} \xi_i$ subject to

$$Y_i(w \cdot X_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$$

This added a hinge-loss term.

General

Decision: $f(X) = w \cdot X + b$

Solve:

$$\min P(w,b) = \frac{1}{2}||w||^2 + C\sum_i H_1[Y_i f(X_i)]$$

$$= \text{maximize margin} + \text{minimize error}$$

SVM: non-linear case

- For some data, linear classifiers are not complex enough
- The solution is to map data into a feature space, and then construct a hyperplane in the feature space as before:
  - $X \mapsto \Phi(X)$
  - Learn $f(X) = w \cdot \Phi(X) + b$

SVM: non-linear case

- For some data, linear classifiers are not complex enough
- The solution is to map data into a feature space, and then construct a hyperplane in the feature space as before:
  - $X \mapsto \Phi(X)$
  - Learn $f(X) = w \cdot \Phi(X) + b$

SVM: non-linear case

- For some data, linear classifiers are not complex enough
- The solution is to map data into a feature space, and then construct a hyperplane in the feature space as before:
    - $X \mapsto \Phi(X)$
    - Learn $f(X) = w \cdot \Phi(X) + b$

SVM: Kernel Trick

- Kernel trick: if $\Phi(X)$ is high-dimensional, it can be hard to solve for w
- By representer theorem (Kimeldorf & Wahba, 1971)

$$w = \sum_i \alpha_i \Phi(X_i)$$

for some α_i

- Optimize α_i instead of w:

$$f(X) = \sum_i \alpha_i K(X_i, X) + b, \quad K(X_i, X) = \Phi(X_i) \cdot \Phi(X)$$

- Rewrite all SVM equations as before but with $w = \sum_i \alpha_i \Phi(X_i)$
- Dual:

$$\min P(w,b) = \frac{1}{2} || \sum_i \alpha_i \Phi(X_i) ||^2 + C \sum_i H_1[Y_i f(X_i)]$$

Common kernel examples

- RBF-SVM:

$$K(X,X') = \exp(-\gamma||X-X'||^2)$$

  Adds a bump around each data point

- Polynomial-SVM:

$$K(X,X') = (X \cdot X')^d$$

  When d is large, the kernel still only requires n computations, whereas explicit representation may not fit in memory

Pros and Cons of SVM

Advantages

- Effective in high dimensional spaces
- Can be used for classification or regression
- Memory efficient since it only uses a subset of training points in the decision function
- Versatile since we can use different kernel functions for the decision function

Pros and Cons of SVM

Disadvantages

- Non-Probabilistic: SVMs do not directly provide probability estimates
- Method will likely not do well if the number of features is much greater than the number of samples

Decision Trees formulation

- Set-up training vectors $X_i \in \mathbb{R}^n$, $i = 1, ..., p$ and a label vector $Y \in \mathbb{R}^p$
- A decision tree recursively partitions the space such that the samples with the same labels are grouped together
- Denote data at node m by Q
- Consider all possible splits of Q into left and right groups
- Choose splits which minimizes some measure of impurity
- Maximum tree depth parameter

Decision Tree Example

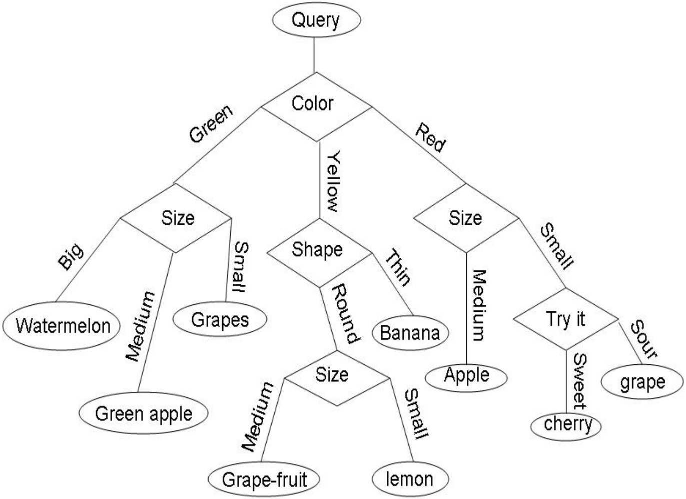


Image source: https://www.projectrhea.org/rhea/index.php/Lecture_21_-_Decision_Trees(Continued)_Old_Kiwi

Pros and Cons of Decision Trees

Advantages

- Simple to understand and to interpret. Trees can be visualized
- The cost of using the tree (i.e., predicting data) is logarithmic in the number of data points used to train the tree
- Possible to validate a model using statistical tests
- Able to handle both numerical and categorical data

Pros and Cons of Decision Trees

Disadvantages

- Decision-tree learners can create over-complex trees that do not generalize the data well
- Decision tree learners create biased trees if some classes dominate
- The problem of learning an optimal decision tree is known to be NP-complete under several aspects of optimality and even for simple concepts

Other Supervised Learning Methods

- Stochastic gradient descent
- Nearest neighbors
- Gaussian process
- Naive bayes
- Ensemble methods

Manifold Learning

- A dimension reduction attempt to generalize linear frameworks like PCA to be sensitive to non-linear structure in data
- Examples:
    - Isomap: seeks a lower-dimensional embedding which maintains geodesic distances between all points
    - Locally linear embedding: seeks a lower-dimensional projection of the data which preserves distances within local neighborhoods
    - Multidimensional scaling: seeks a low-dimensional representation of the data in which the distances respect well the distances in the original high-dimensional space

Clustering: K-means

Goal: Separate samples into K clusters C by solving

$$\sum_{i=0}^{n} \min_{\mu_j \in C}(||x_j - \mu_i||^2),$$

where μ_j is the centroid of the jth cluster

- The within-cluster sum of squares criterion is referred to as inertia and it measures how internally coherent clusters are.
- Inertia makes the assumption that clusters are convex and isotropic
- Inertia is not a normalized metric, better to run PCA before doing K-means clustering if you're in a high dimensional space (curse of dimensionality)

K-means clustering

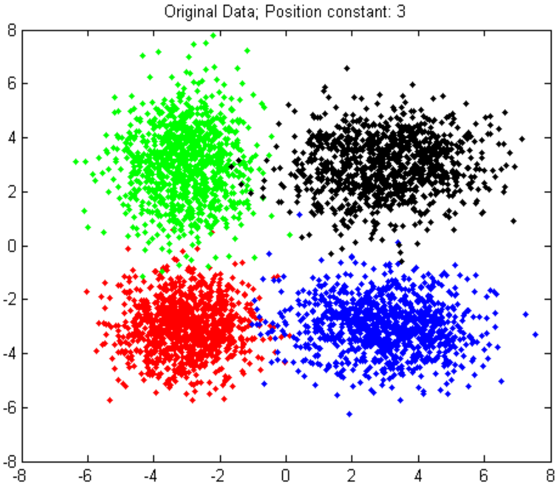


Image source:
http://webstaff.itn.liu.se/~reile/edu/TNM025/Matlab/html/KMeansDemo.html

Clustering: Spectral

- Spectral clustering does a low-dimension embedding of the affinity matrix between samples, followed by a KMeans in the low dimensional space
- The general approach to spectral clustering is to use a standard clustering method on relevant eigenvectors of a Laplacian matrix of similarity matrix A, where $A_{ij} \geq 0$ represents a measure of the similarity between data points with indexes *i* and *j*
- Works well for a small number of clusters but is not advised when using many clusters
- Very useful when the structure of the individual clusters is highly non-convex, or more generally when a measure of the center and spread of the cluster is not a suitable description of the complete cluster

Neural Networks

- A neural network is a system that receives an input, process the data, and provides an output
- Algorithm mimics biological processes of a neuron
- Simplest case: “Neuron” is a computational unit that takes as input $x_1, x_2, x_3, \ldots$ and outputs

$$h_{W,b}(x) = f(W^T x) = f(\sum_i W_i x_i + b),$$

  where $f : \mathbb{R} \to \mathbb{R}$ is called the activation function, W_i are weights and b is the bias
- Common choice for the activation function is tanh
- Add hidden layers to create a more sophisticated network
- Need to learn interconnection pattern between the different layers of neurons and weights of the interconnections based on a cost function

Neural Networks

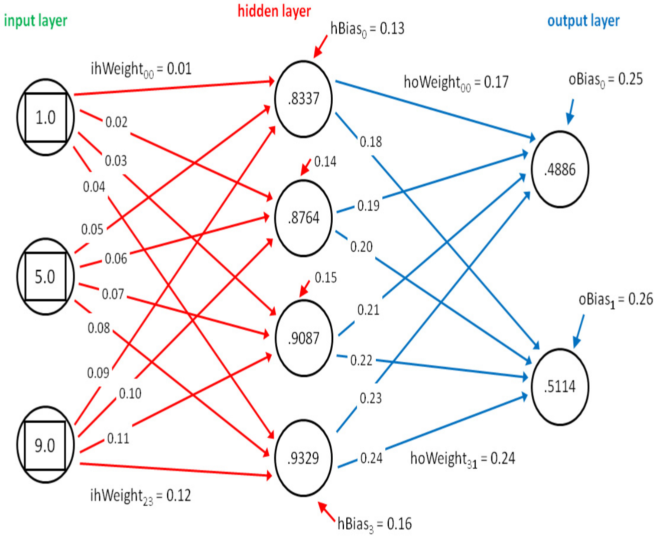


Image source: https://visualstudiomagazine.com/articles/2014/11/01/

Pros and Cons of Neural Networks

Advantages

- Can be trained directly on data with thousands of inputs
- Once trains, predictions are fast
- Performs tasks that linear programs cannot
- Can be used with supervised of unsupervised learning
- Can be done in parallel

Pros and Cons of Neural Networks

Disadvantages

- Training is computationally expensive
- Complex and can be hard to interpret
- Could require a lot of parameter tweaking

Things to consider:

- What is your goal? Classify? Parameter prediction?
- Dimension of your data
- Variability of data
- Pre-existing knowledge of data

- **Classification Metrics:**

$$\text{accuracy}(y,\hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=0}^{n_{\text{samples}}-1} \mathbf{1}(\hat{y}_i = y_i)$$

- **Regression Metrics:**

$$\text{MAE}(y,\hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=0}^{n_{\text{samples}}-1} |y_i - \hat{y}_i|$$

$$\text{MSE}(y,\hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=0}^{n_{\text{samples}}-1} (y_i - \hat{y}_i)^2$$

$$R^2(y,\hat{y}) = 1 - \frac{\sum_{i=0}^{n_{\text{samples}}-1} (y_i - \hat{y}_i)^2}{\sum_{i=0}^{n_{\text{samples}}-1} (y_i - \bar{y})^2}$$

- **Classification Metrics:**

$$\text{accuracy}(y, \hat{y}) = \frac{1}{n_\text{samples}} \sum_{i=0}^{n_\text{samples}-1} \mathbf{1}(\hat{y}_i = y_i)$$

- **Regression Metrics:**

$$\text{MAE}(y, \hat{y}) = \frac{1}{n_\text{samples}} \sum_{i=0}^{n_\text{samples}-1} |y_i - \hat{y}_i|$$

$$\text{MSE}(y, \hat{y}) = \frac{1}{n_\text{samples}} \sum_{i=0}^{n_\text{samples}-1} (y_i - \hat{y}_i)^2$$

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=0}^{n_\text{samples}-1} (y_i - \hat{y}_i)^2}{\sum_{i=0}^{n_\text{samples}-1} (y_i - \bar{y})^2}$$

Applications of Machine Learning

- Spam email detection
- Improving weather prediction
- Targeted advertising and web searches
- Predicting emergency room wait times using staffing levels, patient data, charts, and layout of ER
- Identifying heart failure from physician's notes
- Predicting hospital readmissions
- Learning dynamical systems models directly from high-dimensional sensor data (Byron Boots, Georgia Tech)

Applications of Machine Learning

- Spam email detection
- Improving weather prediction
- Targeted advertising and web searches
- Predicting emergency room wait times using staffing levels, patient data, charts, and layout of ER
- Identifying heart failure from physician's notes
- Predicting hospital readmissions
- Learning dynamical systems models directly from high-dimensional sensor data (Byron Boots, Georgia Tech)

Applications of Machine Learning

- Spam email detection
- Improving weather prediction
- Targeted advertising and web searches
- Predicting emergency room wait times using staffing levels, patient data, charts, and layout of ER
- Identifying heart failure from physician's notes
- Predicting hospital readmissions
- Learning dynamical systems models directly from high-dimensional sensor data (Byron Boots, Georgia Tech)

Applications of Machine Learning

- Spam email detection
- Improving weather prediction
- Targeted advertising and web searches
- Predicting emergency room wait times using staffing levels, patient data, charts, and layout of ER
- Identifying heart failure from physician's notes
- Predicting hospital readmissions
- Learning dynamical systems models directly from high-dimensional sensor data (Byron Boots, Georgia Tech)

Applications of Machine Learning

- Spam email detection
- Improving weather prediction
- Targeted advertising and web searches
- Predicting emergency room wait times using staffing levels, patient data, charts, and layout of ER
- Identifying heart failure from physician's notes
- Predicting hospital readmissions
- Learning dynamical systems models directly from high-dimensional sensor data (Byron Boots, Georgia Tech)

Applications of Machine Learning

- Spam email detection
- Improving weather prediction
- Targeted advertising and web searches
- Predicting emergency room wait times using staffing levels, patient data, charts, and layout of ER
- Identifying heart failure from physician’s notes
- Predicting hospital readmissions
- Learning dynamical systems models directly from high-dimensional sensor data (Byron Boots, Georgia Tech)

Applications of Machine Learning

- Spam email detection
- Improving weather prediction
- Targeted advertising and web searches
- Predicting emergency room wait times using staffing levels, patient data, charts, and layout of ER
- Identifying heart failure from physician’s notes
- Predicting hospital readmissions
- Learning dynamical systems models directly from high-dimensional sensor data (Byron Boots, Georgia Tech)

Useful Software Packages

- PyML, pyMC, scikit-learn in Python
- TensorFlow (Google)
- MATLAB's Statistics and Machine Learning toolbox
- Spider (MATLAB)
- Shogun
- mlpack in C++
- Torch (C++)
- Weka (Java)
- Orange (Open source machine learning and data visualization)

Contamination of classifier

- Goal: General classifier
- Problem: Illusion of success (over-tuning parameters)
- Use cross-validation to avoid this:
  - Randomly divide training data into multiple subsets
  - Only use one subset for training at a time
  - Test each classifier on data not used for training
  - Average results to see how well the classifiers do

Contamination of classifier

- Goal: General classifier
- Problem: Illusion of success (over-tuning parameters)
- Use cross-validation to avoid this:
  - Randomly divide training data into multiple subsets
  - Only use one subset for training at a time
  - Test each classifier on data not used for training
  - Average results to see how well the classifiers do

Overfitting

- “Hallucinating a classifier” occurs when data are not sufficient to determine the correct classifier
- In this case the classifiers will be encoding random features in data which are not grounded in reality
- Overfitting can be decomposed into bias and variance:
  - Bias is the learner’s tendency to consistently learn the same (wrong) thing
  - Variance is the tendency to learn random things irrespective of real signal
  - Linear classifiers have high bias
  - Decision trees have low bias but high variance
- A more powerful algorithm is not necessarily better
- Use cross-validation, add a regularization term to avoid overfitting

Overfitting

- “Hallucinating a classifier” occurs when data are not sufficient to determine the correct classifier
- In this case the classifiers will be encoding random features in data which are not grounded in reality
- Overfitting can be decomposed into bias and variance:
  - Bias is the learner’s tendency to consistently learn the same (wrong) thing
  - Variance is the tendency to learn random things irrespective of real signal
  - Linear classifiers have high bias
  - Decision trees have low bias but high variance
- A more powerful algorithm is not necessarily better
- Use cross-validation, add a regularization term to avoid overfitting

Overfitting

- “Hallucinating a classifier” occurs when data are not sufficient to determine the correct classifier
- In this case the classifiers will be encoding random features in data which are not grounded in reality
- Overfitting can be decomposed into bias and variance:
  - Bias is the learner’s tendency to consistently learn the same (wrong) thing
  - Variance is the tendency to learn random things irrespective of real signal
  - Linear classifiers have high bias
  - Decision trees have low bias but high variance
- A more powerful algorithm is not necessarily better
- Use cross-validation, add a regularization term to avoid overfitting

Overfitting

- “Hallucinating a classifier” occurs when data are not sufficient to determine the correct classifier
- In this case the classifiers will be encoding random features in data which are not grounded in reality
- Overfitting can be decomposed into bias and variance:
  - Bias is the learner’s tendency to consistently learn the same (wrong) thing
  - Variance is the tendency to learn random things irrespective of real signal
  - Linear classifiers have high bias
  - Decision trees have low bias but high variance
- A more powerful algorithm is not necessarily better
- Use cross-validation, add a regularization term to avoid overfitting

Bias and Variance

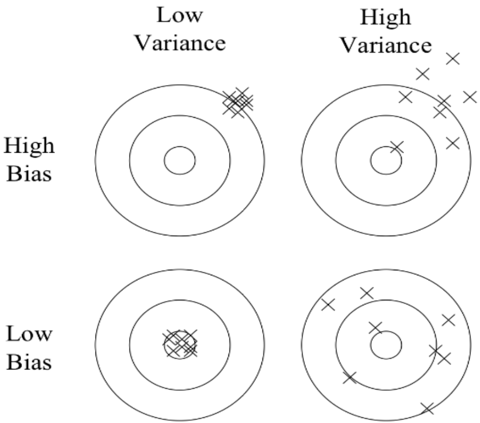


Figure 1: Bias and variance in dart-throwing.

Image source: [1]

Curse of Dimensionality

- Second biggest problem in machine learning
- Expression coined in 1961 to refer to fact that many algorithms work well in low dimensions but fail in high dimensions
- Generalizing correctly is exponentially more difficult as the dimensionality, or number of features, increases
- Similarity-based reasoning fails in high dimensions
- Good news: usually high dimensional data are concentrated on/near a lower dimensional manifold so we can use dimension reduction techniques to avoid this “curse ”

Ways Machine Learning Limits Impacts on World

- C. Rudin and K. Wagstaff argue that most people do not ML with the primary intention of having a significant impact on the world
- Many papers in ML community are rejected solely because their algorithms and analyses are not novel, even if their scientific contribution could have an important impact on society
- In a survey of 152 papers published at ICML 2011, only 1% of papers interpret results in domain context, whereas 39% used synthetic data and 37 % used standard data from UCE archive
- Abstract metrics for performance of algorithms do not measure the impact of the results
- Theoretical advances not connected back to real world impact

Making Machine Learning Matter

1. **Step 1:** Define or select evaluation methods that will allow you to measure the impact of your results
2. **Step 2:** Collaborate with experts in other fields who can help define the ML problem and label data for classification and regression tasks
3. **Step 3:** Consider the potential impact when deciding which research problem to work on

Making Machine Learning Matter

1. **Step 1:** Define or select evaluation methods that will allow you to measure the impact of your results
2. **Step 2:** Collaborate with experts in other fields who can help define the ML problem and label data for classification and regression tasks
3. **Step 3:** Consider the potential impact when deciding which research problem to work on

Making Machine Learning Matter

1. **Step 1:** Define or select evaluation methods that will allow you to measure the impact of your results
2. **Step 2:** Collaborate with experts in other fields who can help define the ML problem and label data for classification and regression tasks
3. **Step 3:** Consider the potential impact when deciding which research problem to work on

Wagstaff’s Impact Challenges

(Original list from Carbonell, 1992)

1. A law passed or legal decision made that relies on the results of an ML analysis
2. Save $100 through an improved decision making algorithm provided by an ML system
3. Avert a conflict between nations though high-quality translation provided by an ML system
4. Save a human life through a diagnosis or invention recommended by a ML system
5. Reduce cyber security break-ins by 50% through ML defenses
6. Improve the Human Development Index by 10% in a country using a ML system

Obstacles to ML Impact

1. Jargon
   - ML vocabulary creates barriers between experts and society or even experts in other fields
   - Replace “feature extraction” with “representation,” “variance” with “instability” and so on
2. Risk
   - “With great power comes great responsibility”
   - Who is at fault for errors when errors have a significant impact?
   - High concerns in fields such as medicine, spacecraft, finance, etc.
3. Complexity
   - ML is not simple enough for researchers across fields to use freely
   - Simplifying, maturing, and “robustifying” ML tools will promote wider, more independent uses of ML

Thanks for listening!!!

References

[1] Domingos, Pedro. A Few Useful Things to Know about Machine Learning.
[2] Rudin, Cynthia and Kiri L. Wagstaff. Machine Learning for Science and Society. Mach Lean, *Springer*. (2013)
[3] Wagstaff, Kiri L. Machine Learning that Matters. *Proceedings of the 29th International Conferences on Machine Learning,* Edinburgh, Scotland. (2012)
[4] `scikit-learn.org/stable/user_guide.html`
[5] `https://www.quantstart.com/articles/Support-Vector-Machines-A-Guide-for-Beginners`
[6] `www.cs.columbia/edu/~kathy/cs4701/documents/jason_svm_tutorial.pdf`