01: Machine Learning **July 18th, 2016**

Lecture 1: Decision Tree Learning & Review of Probability

Lecturer: Christopher M. Bishop *Scribes: Yan Yan*

Note: *Notes for machine learning course of Mitchell.*

Disclaimer: *These notes have not been subjected to the usual scrutiny reserved for formal publications. They may be distributed outside this class only with the permission of Mr. Yan.*

This chapter is based on Lecture 1 and Lecture 2 of Mitchell class, the reading references are *Pattern Recognition and Machine Learning* of Bishop and *Machine Learning* of Mitchell. Also the PPT file of Andrew Moore and *The Discipline of Machine Learning* were also mentioned.

1.1 The Discipline of Machine Learning

1.1.1 Defining Questions

THE filed of machine learning seeks to answer the question: ”How can we build computer systems that automatically improve with experience, and what are the fundamental laws that govern all learning processes?”

Define: A machine learns with respects to a particular task T, performance metric P, and a type of experience E, if the system reliably improves its performance P at task T, following experience E.

Depending on how we specify T, P and E, the learning task might also be called by names such as data mining, autonomous discovery, etc.

Computer Science: how can we build machines that solve problems, and which problems are inherently tractable/intractable?

V.S.

Statistics: what can we inferred from data plus a set of modelling assumptions, with what reliability?

Computer Science has focused primarily on how to manually program computers, ML focused on how to get computer to program themselves from experience, whereas Statistics focus on the what conclusions can be inferred from data.

A third field whose defining questions is closely related to ML is the study of human and animal learning in Psychology, Neuroscience, and related fields. Other field like biology, econonics to control theory also have a core interest in the question of how systems can automatically adapt or optimize to their environment. The mathematical models for adaptation inthese other fields are somewhat different from those commonly used in machine learning, suggesting significant potential for cross-fertilisation of models and theories.

1.1.2 State of ML

The real-world applications, such as those listed below.

1. *Speech recognition*

2. *Computer vision* like hand written addresses.
3. *Bio-surveillance* like the RODS project.
4. *Robot control* helicopter flight and helicopter aerobatics. The recent Darpa-sponsored competition involving a robot driving autonomously for over 100 miles in the desert was won by a robot that used machine learning to refine its ability to detect distant objects.
5. *Accelerating empirical sciences*, many data-intensive sciences now make use of machine learning methods to aid in the scientific discovery process.

1.1.3 Place of Machine Learning within Computer Science

In particular, machine learning methods are already the best methods available for developing particular types of software, in applications like:

1. The application in too complex for people to manually design the algorithm - sensor based perception task. computer vision.
2. The application requires that the software customise to its operational environment after it it fielded - speech recognition

By shifting the question from ”how to program computers” to ”how to allow them to program themselves”, machine learning emphasise the design of self monitoring systems that self-diagnose and self-repair and on approaches that model their users, and the take advantage of the steady stream of data flowing through the program rather than simply processing it.

ML will help reshape the field of Statistics, by bringing a computational perspective to the fore, and raising issues such as never-ending learning.

Some current Research Questions:

1. *Can unlabeled data be helpful for supervised learning?*
2. *How can we transfer what is learned for one task to improve learning in other related tasks?*
3. *What is the relationship between different learning algorithms, and which should be used when?*
4. *For learners that actively collect their own training data, what is the best strategy?*
5. *To what degree can we have both data privacy and the benefits of data mining?*

The above research questions are already being energetically pursued by researchers in the field. It is also interesting to consider longer term research questions. Some additional research topics are some potential questions may significantly change the face of machine learning over the coming decades:

1. *Can we build never-ending learners?* Learning in animals or human beings is an ongoing process in which the agent learns many different capabilities, often in a sequenced curriculum, and uses these different learned facts and capabilities in a highly synergistic fashion. - why not build machine learners that learn in this same cumulative way!
2. *Can machine learning theories and algorithms help explain human learning?* Reinforce learning.
3. *Can we design programming languages containing machine learning primitives?*
4. *Will computer perception merge with machine learning?*

1.1.4 Where to Learn More

Related Sources:

1. *International Conference on Machine Learning.*
2. *Conference on Neural Information Processing Systems.*
3. *Annual Conference on Learning Theory.*
4. *Journal of Machine Learning Research(JMLR) - www.jmlr.org*
5. *Machine Learning*

1.2 Decision Tree Learning (Based on Bishop Chapter 14.4)

By partitioning the input space into cuboid regions, whose edges are aligned with the axes, and then assigning a simple model to each region. The process of selecting a specific model, given a new input x, can be described by a sequential decision making process corresponding to the traversal of a binary tree. Here we focus on a particular tree-based framework called CART. Some other type of methods like ID3 and C4.5.

....

In order to learn such a model from a training set, we have to determine the structure of the tree, including which input variable is chosen at each node to form the split criterion as well as the valued of the threshold parameter θ_i for the split. We also have to determine the values of the predictive variable within each region.

1.3 Lecture 1 of Mitchell

1.3.1 Concepts

Machine learning:

Study of algorithms that

* improve their performance P
* at some task T
* with experience E

well-defined learning task:¡P, T, E¿.

1.3.2 Learning to Predict Emergency C-Sections

(Sims et al. 2000) A database with 9714 patient records each with 215 features.

Over training data: 26/41 = .63

Over test data: 12/20 = .60

- Data mining try to learn how to predict future

1.3.3 Learning to detect objects in images

Example training images for each orientation.

1.3.4 Learning to classify text documents.

High dimensional data. -Brain Image data.

1.3.5 Machine learning - Practice

Mining databases;

Speech Recognition;

Control learning;

Object recognition;

Text analysis.

1.3.6 Machine Learning - Theory

PAC Learning Theory, supervised concept learning. examples - m

representation complexity - H

error rate - e

failure probability - δ

Other theories for Reinforcement skill learning, semi-supervised learning active student querying.

also relating:

- of mistakes during learning
- learner's query strategy
- convergence rate
- asymptotic performance
- bias, variance

1.3.7 Machine Learning in Computer Science

Machine Learning already the preferred approach to:

1. Speech recognition, Natural language processing
2. Computer vision

3. Medical outcomes analysis
4. Robot control

This ML niche is growing, why ?

1.3.8 Function Approximation and Decision Tree Learning

have some function to some x to some y,

Function approximation:

Problem Setting:

1. set of possible instance X
2. unknown target function $f : X- > Y$
3. set of function hypotheses $h = \{h|h : X- > Y\}$

Input: - Traning examples$\{x^{(i)}\}$

Output: Hypotheses $h \in H$

1.3.9 Decision Tree Learning

Problem Setting:

-Set of possible instances S

each instanc x in X is a feature vector

-Unknown target function $f : X- > Y$

Y is discrete valued

- Set of function hypotheses

1.3.10 Decision Tree

Suppose $X =< X_1, ...X_n >$ where X_i are boolean variables.

Top-Down Induction of Decision Trees

[ID3, C4.5, Quinlan]

node = Root

Main loop:

1. $A < -$ the "best" decision attribute for next node
2. Assign A as decision attribute for node

3. For each value of A, create new descendant of *node*
4. Sort training examples to leaf nodes
5. If training examples perfectly classified, then STOP, Else iterate over new leaf nodes.

1.3.11 Entropy

Entropy $H(X)$ of a random variable X:

$H(X)$ is the expected number of bits needed to encode a randomly drawn value of X (under most efficient code)

Why? Information theory:

Sample Entropy

Entropy measures the impurity of S.

Conditional entropy $H(X|Y = v)$

Information Gain is the mutual information between input attribute A and target variable Y. Information Gain is the expected reduction in entropy of target variable Y for data sample S, due to sorting on variable A.

1.4 Review of Probability

Two-class classification probl em:

$$y(x) = w^T \phi(x) + b \tag{1.1}$$

where ϕ denotes a fixed feature-space transformation, b explicit.

Training set comprises N input vectors $x_1, \cdots, x_N$ with target $t_1, /cdots, t_N$ where $t_N \in \{-1, 1\}$ and new data points are classified according to the sign of $y(x)$.

Training data - linearly separable in feature space, so that there exits one choice of w and b such that by 1.1 satisfies $y(x_n) > 0$ for points having $t_n = +1$ and $y(x_n) < 0$ for $t_n = -1$.

In section 4, the perceptron algorithm is guaranteed to find a solution for the linearly separable problem. The solution deeps on the initial value and the order of the data points are presented - if there are multiple solutions we should try to find one with the smallest generalizaiton error. SVM approaches this problem through the concept of the **margin** - the smallest distance between the decision boundary and any of the samples.

Perpendicular distance of a point x from a hyperplane: $|y(x)|/\|w\|$. We are only interested in solutions for which all data points are correctly classified $(t_n y(x_n)) > 0$ for all n). So we have the distance of a point x_n to the decision surface is given by:

$$\frac{t_n y(x_n)}{\|w\|} = \frac{t_n(w^T \phi(x_n) + b)}{\|w\|} \tag{1.2}$$

The margin is given by the perpendicular distance to the closest point x_n from the data set, w and b need optimized in order to maximize the distance.

Then the maximum margin solution is found by

$$\arg\max_{w,b}\{\frac{1}{\|w\|}\min_n[t_n(w^T\phi(x_n)+b)]\} \tag{1.3}$$

in which $\|w\|$ does not depend on n. The direct solution of this optimization problem would be complex $\Rightarrow$ an equivalent problem easier to solve:

$$w \to \kappa w \tag{1.4}$$

$$b \to \kappa b \tag{1.5}$$

the rescaling would not change the distance. We then use this freedom to set

$$t_n(w^T\phi(x_n)+b)=1 \tag{1.6}$$

for the point that is closest to the surface.

The canonical representation of the decision hyperplane:

$$t_n(w^T\phi(x_n)+b)\geq 1, n=1,\cdots,N \tag{1.7}$$

When the equality holds, the constraints are said to be active.

There would always be at least one active constraint, because there will always be a closest point, and once the margin has been maximized there will be at leat two active constraints. The optimization problem minimizing $\|w\|^2$ is equivalent to maximize $\|w\|^{-1}$, so we need to solve the problem:

$$\arg\min_{w,b}\frac{1}{2}\|w\|^2 \tag{1.8}$$

subject to the constraints given by 1.7.

We introduce the Lagrange multipliers $a_n \geq 0$ with one multiplier a_n for each of the constraints in1.7 giving the Lagrangian function:

$$L(w,b,a)=\frac{1}{2}\|w\|^2-\sum_{n=1}^{N}a_n\{t_n(w^T\phi(x_n)+b)-1\} \tag{1.9}$$

where $a=(a_1,\cdots,a_N)^T$.

Setting the derivatives of $L(w,b,a)$ with respect to w abd b equal to zero, we obtain the following conditions:

$$w=\sum_{n=1}^{N}a_nt_n\phi(x_n) \tag{1.10}$$

$$0=\sum_{n=1}^{N}a_nt_n \tag{1.11}$$

Eliminating w and b from $L(w,b,a)$ using these conditions then gives the dual representation of the **maximum margin problem** in which we maximize

$$\tilde{L}(a)=\sum_{n=1}^{N}a_n-\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}a_na_mt_nt_mk(x_n,x_m) \tag{1.12}$$

with respect to a subject to the constraints

$$a_n \geq 0, n = 1, \cdots, N \tag{1.13}$$

$$\sum_{n=1}^{N} a_n t_n = 0 \tag{1.14}$$

Here the kernel function is defined by $k(x, x') = \phi(x)^T \phi(x')$. This takes the form of a **quadratic programming problem**.

Computational complexity - $O(M^3)$.

With dual formulation, the original optimization problem involved minimizing over M variables into the dual problem with N variables.

For fixed set of basis functions M is smaller than the data points number N, dual representation seems useless. However, it allows the model to be reformulated using kernels, and the maximum margin classifier can be applied efficiently to feature spaces whose dimensionality M exceeds the number of data points N, including infinite feature spaces.

The kernel formulation also makes clear the role of the constraint that the kernel function $k(x, x')$ be **positive definite** - this ensures the Lagrangian function $\tilde{J}(a)$ is bounded below, giving rise to a well-defined optimization problem.

So new data point can be evaluated by:

$$y(x) = w^T \phi(x) + b = \sum_{n=1}^{N} a_n t_n k(x, x_n) + b \tag{1.15}$$

For the Karush-Kuhn-Tucker KKT condtion, which in this case require that the following three properties hold:

$$a_n \geq 0 \tag{1.16}$$

$$t_n y(x_n) - 1 \geq 0 \tag{1.17}$$

$$a_n \{t_n y(x_n) - 1\} = 0 \tag{1.18}$$

---

check the Appendix E

For every data point, either $a_n = 0$ or $t_n y(x_n) = 1$, any data point for which $a_n = 0$ will not appear in the sum in1.15 thus play no role in making predictions for new data points. **The remaining data points are called support vectors** - and they satisfy $t_n y(x_n) = 1$, they lie on the maximum margin hyperplanes in feature space – once the model is trained only the support vectors retained.

Solving the quadratic programming problem for a, determine b by

$$t_n(\sum_{m \in \mathcal{S}} a_m t_m k(x_n, x_m) + b) = 1 \tag{1.19}$$

where $\mathcal{S}$ denotes the set of indices of the support vectors.

A more stable solution for b is:

$$b = \frac{1}{N_{\mathcal{S}}} \sum_{m \in \mathcal{S}} (t_n - \sum_{m \in \mathcal{S}} a_m t_m k(x_n, x_m)) \tag{1.20}$$

where $N_{\mathcal{S}}$ is the total number of support vectors.

1.4.1 Overlapping class distributions

1.4.2 Relation to logistic regression

1.4.3 Multiclass SVMs

1.4.4 SVMs for regression

1.4.5 Computational learning theory

1.5 Relevance Vector Machines

1.5.1 RVM for regression

1.5.2 Analysis of sparsity

1.5.3 RVM for classification

We now delve right into the proof.

Lemma 1.1 *This is the first lemma of the lecture.*

Proof: The proof is by induction on .... For fun, we throw in a figure.

Figure 1.1: A Fun Figure

This is the end of the proof, which is marked with a little box. ■

1.5.4 A few items of note

Here is an itemized list:

- this is the first item;
- this is the second item.

Here is an enumerated list:

1. this is the first item;
2. this is the second item.

Here is an exercise:

Exercise: Show that P $\neq$ NP.

Here is how to define things in the proper mathematical style. Let f_k be the $AND - OR$ function, defined by

$$f_k(x_1, x_2, \ldots, x_{2^k}) = \begin{cases} x_1 & \text{if } k = 0; \\ AND(f_{k-1}(x_1, \ldots, x_{2^{k-1}}), f_{k-1}(x_{2^{k-1}+1}, \ldots, x_{2^k})) & \text{if } k \text{ is even;} \\ OR(f_{k-1}(x_1, \ldots, x_{2^{k-1}}), f_{k-1}(x_{2^{k-1}+1}, \ldots, x_{2^k})) & \text{otherwise.} \end{cases}$$

Theorem 1.2 *This is the first theorem.*

Proof: This is the proof of the first theorem. We show how to write pseudo-code now.

Consider a comparison between x and y:

```
if $x$ or $y$ or both are in $S$ then
    answer accordingly
else
    Make the element with the larger score (say $x$) win the comparison
    if $F(x) + F(y) < \frac{n}{t-1}$ then
        $F(x) \leftarrow F(x) + F(y)$
        $F(y) \leftarrow 0$
    else
        $S \leftarrow S \cup \{x\}$
        $r \leftarrow r + 1$
    endif
endif
```

This concludes the proof. ■

1.6 Next topic

Here is a citation, just for fun [CW87].

References

[CW87] D. Coppersmith and S. Winograd, "Matrix multiplication via arithmetic progressions," *Proceedings of the 19th ACM Symposium on Theory of Computing*, 1987, pp. 1–6.