CMPUT 367

Roderick Lan

Contents

1 Lecture 1 - Jan 9

1.1 What is ML

Machine = automation.
ML = Learning[1] from experience spacing

$$\text{Training/Learning experience: } \xrightarrow{\text{ML Algo}} \text{ML model}$$

$$\text{Inference/Prediction}^2\text{: } x_* \xrightarrow{\text{ML Model}} \hat{y}^*$$

Input features d-dimensional real vectors ($\vec{x}^{(m)} \in \mathbb{R}^d$)
$y^{(m)} \in \mathbb{R}$ regression problem
$y^{(m)} \in \{\}$ classification problem spacing

$k = 2 \rightarrow$ binary; $k \geq 2 \rightarrow$ multi class

categories mutually exclusive

Explain 1.1.1: Multilabel Classification

Naive: Solve as separate task (structured prediction)

If $y^{(m)}$ has internal structure: Treat as separate tasks; Structured prediction (PGM)

1.2 Supervised Learning

Labeled datasets used (ie. given labeled training data)
Experience is a tuple $\{(x^{(m)}, y^{(m)})\}_{m=1}^{M}$

2 Lecture 2 - Jan 11

2.1 Supervised Learning

1. **Training/Learning:**
   Experience $\xrightarrow{\text{ML Training}}$ ML Model
2. **Inference/Prediction:**

2.2 Unsupervised Learning

$t^{(m)}$ is not given in training

Labels are not given; samples are unlabeled. Patterns in data are present (ie. from clustering). Task is ambiguous; less well defined.

[1]"Learning" from instructions = programming; from experience = from dataset
[2]x_* = new data

Example 2.2.1: Types of Unsupervised Learning

Clustering - similar things close together
Outlier Detection - datapoints outside of trend
Representation Learning - extract meaningful patterns to create representations that are easier to understand/process

2.3 Reinforcement Learning

Well defined. Gives feedback on actions through rewards, doesn't give 'answer' like supervised learning.

2.3.1 SL vs RL

Supervised Learning - taught on exact steps
Reinforcement Learning - taught on feedback from actions (no reference solution)

2.4 Regression

2.4.1 Linear Regression

Input = $x^{(m)} \in \mathbb{R}^d$
Output = $y^{(m)} \in \mathbb{R}$
Data = $\{(x^{(m)}, t^{(m)})\}_{m=1}^{M}$

Explain 2.4.1

"Can I learn a function from $\{h : \mathbb{R}^d \to \mathbb{R}\}$"
No, set too powerful.

To make set meaningful, we need to restrict set of functions (**Hypothesis set**). Only consider functions in hypothesis set.

$\mathcal{H}_{\textsf{linear}} = \left\{ h : \mathbb{R}^d \to \mathbb{R} | h(x) = \sum_{i=1}^{d} w_i x_i + b, w_i, b \in \mathbb{R} \right\}$

Explain 2.4.2: Visualizing Higher Dim Space

Hypotheis set is a hyperplane. Vertical hyperplane invalid (same as low dim)

Training loss function/objective = $J(h, \mathcal{D}_{\text{train}})$
$h^* = \min_{h \in \mathcal{H}} J(h, \mathcal{D}_{\text{train}})$
Linear Regression Objective[1]:
$(|h(x^{(m)}; w, b) - t^{(m)}|)^2$

[1]Squared to give more weight to larger errors; Probabilistic interpretation

3 Lecture 3 - Jan 16

$J(w) = \frac{1}{2M} \|Xw - t\|^2$

$\frac{\partial J}{\partial W} = \frac{1}{M}[X^\top Xw - X^\top t] = 0$

$w = (X^\top X)^{-1} X^\top t$ when $X^\top X$ is invertible

When is $X^\top X$ non-invertible (not full rank)

$X \in \mathbb{R}^{M \times (d+1)}$

If rank is not $d+1$, then

- $M < d+1$; more features than samples (underdetermined)
  - pseudoinverse works mathematically but may not give a meaningful ML **model**
  - fix by simplifying model, refine feature selection
  - "sparse" models automatically select relevant features
- Duplicate features (linearly dependent); rank of $X^\top$ less than $d+1$
  - can use pseudoinverse

Problem of closed-form sol. for MSE:

calculating inverse not fun

slow $O(d^3)$

can be numerically unstable

3.1 Gradient Based Methods

Algorithm 1: Gradient Descent

Randomly initialize $w^{(0)}$;
for $e = 1, 2, \cdots$ *until satisfied* **do**
  $w^{(e)} = w^{(e-1)} - \alpha \nabla_w J(w)\Big|_{w=w^{(e-1)}}$
end for

Big problem of gradient: might not point to desired direction
gradient always in direction orthogonal to contour

3.1.1 Batch Gradient

Use a couple samples to approx gradient (very cheap), reach optimimum faster than full batch.

3.2 Closed Form Sol. vs Iterative Methods

Use closed form solution when it **exists**, is **cheap** and is **numerically stable**
Use iterative method otherwise.

3.3 Probabilistic Interpretation:

For **linear regression**:

Assume $\epsilon \sim \mathcal{N}(0, \sigma^2)$

Assume target $t^{(m)} = w^\top X^{(m)} + \epsilon^{(m)}$ where w is unknown constant

Target $t^{(m)} \sim \mathcal{N}(w^\top x^{(m)}, \sigma^2)$
(target is gaussian since error/noise gaussian)
(gaussian chosen because it occurs naturally + CLT)
(gaussian tail decreases exponentially/quadratically)

Non-Gaussian:

Uniform

Laplace (similar to gaussian)

Power law distr / zipf distr. (similar to gaussian, but very long tail)

Poisson (converges to gaussian $x \to \infty$)

$\mathcal{D}_{\mathsf{train}} = \left\{ (x^{(m)}, t^{(m)})_{m=1}^{M} \right\}$
Discrete and Generative Product