1 Introduction

1.1 Types of Machine Learning

- Supervised Learning
  - Classification: ডেটাকে নির্দিষ্ট ক্যাটাগরিতে ভাগ করা
  - Regression: একটি কন্টিনিউয়াস রেজাল্টকে প্রেডিক্ট করা
- Unsupervised Learning
  - Clusters: এক রকমের ডেটা পয়েন্টগুলোকে একসাথে গ্রুপ করা
  - Discovering latent factors: ডেটার মধ্যে লুকানো ফ্যাক্টর খুঁজে বের করা
  - Discovering graph structure: ডেটার মধ্যে বিভিন্ন সম্পর্ক খুঁজে বের করা, যেখানে ডেটাকে নোড এবং এজ দিয়ে গ্রাফ আকারে দেখানো যায়
  - Matrix completion: কোথাও ডেটা ম্যাট্রিক্সের কিছু অংশ মিসিং থাকলে, সেটা পূরণ করার চেষ্টা করা হয়

2 Three elements of a machine learning model

Model = Representation + Evaluation + Optimization[1]

2.1 Representation

Supervised Learning-এর ক্ষেত্রে মডেলকে সবসময় তৈরী করতে হবে conditional probability distribution $P(y|\vec{x})$ আকারে অথবা decision function $f(x)$ হিসেবে। এই রিপ্রেসেন্টেশন ক্লাসিফিকেশনের ক্ষেত্রে যদি ধরি , এই কন্ডিশনাল ডিস্ট্রিবিউশনের মাধ্যমে আমরা বের করতে পারছি কোনো ইনপুট $f(x)$ দেয়ার পর কোন ক্লাস লেবেল y পাওয়ার সম্ভাবনা কতটুকু আছে. মেশিন লার্নিংয়ের ভাষায় এই ডিস্ট্রিবিউশনকে ক্লাসিফায়ার বলা হয়. এই সকল ক্লাসিফায়ারকে নিয়ে একসাথে যেই set তৈরি করা হয় তাকে hypothesis space বলে।

2.2 Evaluation

hypothesis space এর মধ্যে থাকা, কোনটা good classifier এবং konta bad classifer sheta bujhar jonno evaluation function (objective function or risk function) ব্যবহার করা হয়। মডেল যখন ক্লাসিফায়ারদের মধ্যে পার্থক্য করতে চায়, তখন এই evaluation function একটা স্কোর বা ভ্যালু রিটার্ন করে যেটার মাধ্যমে learning algorithm determine করতে পারে কোন ক্লাসিফায়ার সবচেয়ে ভাল।

2.2.1 Loss function and risk function

Definition 0.1. Loss Function

হাইপোথেসিস স্পেস (hypothesis space) ডিফাইন করার পরে এভালুয়েশন (evaluation) প্রসেস এর ক্ষেত্রে প্রথম ধাপ হচ্ছে প্রত্যেক্টা ক্লাসিফায়ারে লস ফাংশন প্রয়োগ করা, যেটা নির্দেশ করবে বা বুঝাবে যে একটা ক্লাসিফায়ার এর প্রেডিকশন কতটুকু ট্রেনিং সেট এর সাথে ম্যাচ করতে পেরেছে। এক্ষেত্রে প্রতিটা প্রেডিকশনের উপর লস ফাংশন প্রয়োগ করা হয়, লার্নিং এলগোরিদম ট্রেনিং ডাটার উপর গড় (mean) বা সম্পূর্ণ (total) লস কমিয়ে সবচেয়ে ভালো পারফর্ম করা ক্লাসিফায়ারকে খুঁজে বের করে।

একটা ফাংশন কত ভালভাবে ট্রেইনিং ডাটা এর উপর ফিট সেটা পরিমাপ করার জন্য একটা লস ফাংশন (loss function) সংজ্ঞায়িত করি। একটি ট্রেইনিং এক্সাম্পল $(x_i, y_i) y_h at L(y, y_h at)$

loss function

L:Y $\times Y \to R \geq 0$ (1)

- $Y \times Y$: label output ;
  - y_i: ith (actual)
  - $\widehat{y}$:
- $R \geq 0$: - ($R \geq 0$) " " actual y_i $\widehat{y}$ -

:

- 0-1 loss function
  $$L(Y, f(X)) = \mathbb{I}(Y, f(X)) = \begin{cases} 1, & Y = f(X) \\ 0, & Y \neq f(X) \end{cases}$$
  - I(Y,f(X)) : , actual y_i $f(X)$, 1, 0
  - , ,
- Quadratic loss function $L(Y, f(X)) = (Y - f(X))^2$
  - $Yf(X)$ Y $f(X)$
  - Mean Squared Error regression ,
- Absolute loss function $L(Y, f(X)) = |Y - f(X)|$
  - $|Y - f(X)|$ y $f(X)$

[1] Domingos, P. A few useful things to know about machine learning. Commun. ACM. 55(10):78–87 (2012).

– ,
average loss error

- Logarithmic loss function
$L(Y, P(Y|X)) = -\log P(Y|X)$
  - $P(Y|X)$ X y
$-\log P(Y|X)$
, high loss
value
  - $P(Y|X)$ probability score low, Logarithmic
loss function prediction penalize ;
loss function classifcation

Definition 0.2. $R_{\exp}(f)$ expected loss risk function; f error
(loss)

$$R_{\exp}(f) = E\left[L(Y, f(X))\right] = \int L(y, f(x))\, P(x, y)\mathrm{d}x\mathrm{d}y \tag{2}$$

- $L(Y, f(X))$: loss function
- $E[]$: Expectation , probability distribution
average expected value
- $P(x, y)$: input data X label Y joint probaility
distribution, -
- average value integral
$\int mathrmdx\mathrm{d}y$; data point
X Y loss value average probability
distribution

Definition 0.3. training data The risk function
$R_{\exp}(f)$ -

$$R_{\text{emp}}(f) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)) \tag{3}$$

- $R_{\text{emp}}(f)$,
f
- total loss average
- $L(y_i, f(x_i))$ data y_i, x_i loss function apply
, x_i y_i , $f(x_i)$

empirical loss empirical risk- .

আমরা কিন্তু চাইলে আমাদের নিজেদের মতো করেও লস ফাংশন ডিফাইন করতে পারি; কিন্তু শুরুর দিকে শেখার অবস্থায় লিটারেচর থেকে থেকে একটি ব্যবহার করা আমাদের জন্য ভালো হবে। লস ফাংশন ডিফাইন করার সময় অবশ্যই কিছু বিষয় মাথায় রাখতে হবে-[2]

1. মডেল যেই আসল লস(actual loss) কমানোর চেষ্টা করছে, সেই লসকে কাছাকাছি আনাই লস ফাংশনের কাজ। উদাহরণসরূপ, ক্লাসিফিকেশনের জন্য একটি সাধারণ লস ফাংশন হল জিরো-ওয়ান লস, যেটা শুধু কতগুলো ভুল ক্লাসিফিকেশন হয়েছে সেই হিসাব রাখে; একটি ভুল প্রেডিকশনের জন্য ১ এবং সঠিক প্রেডিকশনের জন্য ০ দেয়
2. আমরা যেই নির্দিষ্ট অপটিমাইজেশন ব্যবহার করতে চাই তাকে অবশ্যই মানানসই হতে হবে লস ফাংশনকে অবশ্যই জন্যই জিরো-ওয়ান লস সরাসরি ব্যবহার করা হয় না, কারণ এটা গ্রেডিয়েন্ট-ভিত্তিক অপটিমাইজেশন মেথড এর সাথে কাজ করে না
The main algorithm that optimizes the zero-one-loss directly is the old perceptron algorithm(chapter □??).

2.2.2 ERM and SRM

ERM(ERM (Empirical Risk Minimization)) এর লক্ষ্য হল প্রত্যেকটা ট্রেইনিং ডাটা থেকে প্রাপ্ত লস ফাংশনের এভারেজ ভ্যালু বের করা । এই পদ্ধতিতে আমরা হাইপথিসিস স্পেইস থেকে এমন একটি ফাংশন f (মডেল বা ক্লাসিফায়ার) খুঁজে পাই যা ট্রেনিং ডেটায় error-কে কমায়ে রাখে।
SRM স্ট্রাকচারাল রিস্ক মূলত এম্পিরিক্যাল রিস্কের সাথে একটি অতিরিক্ত পেনাল্টি টার্ম $\lambda J(f)$ যোগ করে যখনই মডেলের কমপ্লেক্সিটি বাড়তে থাকে। এখন প্রশ্ন আসে, মডেলের কমপ্লেক্সিটি বাড়লে কি সমস্যা? এক্ষেত্রে মডেল ডেটার প্যাটার্নের পাশাপাশি অপ্রয়োজনীয় প্যাটার্নও ধরতে থাকবে জেগুলো মূলত নয়েস (noise)। পেনাল্টি টার্ম এক ধরনের ব্যালেন্স তৈরি করে যাতে মডেলটি ওভারফিট না করে। ERM এর মতো মডেল ট্রেনিং ডেটায় বেশি ফিট করার পাশাপাশি মডেলটি যেন বেশি জটিল না হয় তা নিশ্চিত SRM ।

Definition 0.4. ERM(Empirical risk minimization)

$$\min_{f\in\mathcal{F}} R_{\text{emp}}(f) = \min_{f\in\mathcal{F}} \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)) \tag{4}$$

- N data
R_{emp}

Definition 0.5. Structural risk

$$R_{\text{smp}}(f) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)) + \lambda J(f) \tag{5}$$

□ $J(f)$ এমন একটি টার্ম যা বেশি জটিল মডেলকে শাস্তি দেয়
□ λ দ্বারা নির্ধারিত হয় কতটুকু শাস্তি বা পেনলাইজ করা হবে কমপ্লেক্সিটি লেভেল ঠিক রাখার জন্যে

Definition 0.6. SRM(Structural risk minimization)
SRM- F f

$$\min_{f\in\mathcal{F}} R_{\text{srm}}(f) = \min_{f\in\mathcal{F}} \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)) + \lambda J(f) \tag{6}$$

[2] http://t.cn/zTrDxLO

- $R_{\text{srm}}(f)$,
- $\frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i))$: ,
- $\lambda J(f)$,

2.3 Optimization

মেশিন লার্নিং মডেল ডেভেলপমেন্টের সর্বশেষ ধাপ হচ্ছে অপটিমাইজেশন (gradient descent), যার মধ্যমে হাইপথিসিস স্পেইস থেকে সেরা ক্লাসিফায়ার সার্চ স্পেইস থেকে কত কার্যকরীভাবে

3 Some basic concepts

3.1 Parametric vs non-parametric models

প্যারামেট্রিক মডেল: এগুলির নির্দিষ্ট সংখ্যক প্যারামিটার থাকে। মডেলটি একবার ট্রেইনড হয়ে গেলে, প্যারামিটারগুলি নির্দিষ্ট হয়ে যায় এবং মডেলের কমপ্লেক্সিটি বাড়েনা। যেমন লিনিয়ার রিগরেশন, লজিস্টিক রিগরেশন

নন-প্যারামেট্রিক মডেল: এক্ষেত্রে মডেলের নির্দিষ্ট সংখক প্যারামিটার থাকে এবং ডাটাসেট বৃদ্ধির সাথে সাথে মডেলের কমলেক্সিটি বা জটিলতা বাড়তে থাকে। এগুলি আরও ফ্লেক্সিবল এবং ডেটার পরিমাণ বেশি থাকা লাগে।

3.2 A simple non-parametric classifier: K-nearest neighbours

3.2.1 Representation

KNN একটি নন-প্যারামেট্রিক ক্লাসিফায়ার যেখানে একটি পয়েন্টের আউটপুট হয় তার সবচেয়ে কাছের টি প্রতিবেশীর সাধারণ শ্রেণী।

$$y = f(\vec{x}) = \arg\min_{c} \sum_{\vec{x}_i \in N_k(\vec{x})} \mathbb{I}(y_i = c) \qquad (7)$$

যেখানে $N_k(\vec{x})$ k পয়েন্টের একটি সেট যারা $\vec{x}$ পয়েন্টের কাছাকাছি।

- $N_k(\vec{x})$ হচ্ছে পয়েন্ট X এর আশেপাশের k-nearest neighbor
- $I(y_i = c)$ ইন্ডিকেটর ফাংশন যদি y_i c ক্লাসের মধ্যে পড়ে তবে 1 রিটার্ন করবে আর যদি না হয় তবে 0 রিটার্ন করে

Usually use k-d tree to accelerate the process of finding k nearest points.

উদাহরণ: যদি =3 হয় এবং x -এর সবচেয়ে কাছের 3 জন প্রতিবেশীর মধ্যে দুটি শ্রেণী -তে এবং একটি শ্রেণী B-তে থাকে, তাহলে -এর আউটপুট শ্রেণী হবে, কারণ এটি A-এর প্রতিবেশীদের মধ্যে সবচেয়ে সাধারণ।

Example: If k=3 and among the 3 nearest neighbors of point x, two belong to class A and one belongs to class B, then the output class y will be A, because A is the most common class among the neighbors.

3.2.2 Evaluation

No training is needed.

3.2.3 Optimization

No training is needed.

3.3 Overfitting

ওভারফিটিং হয় যখন একটি মডেল ট্রেইনড ডেটাতে খুব ভালো কাজ করে কিন্তু নতুন ডেটাতে খারাপ করে। এটি খুব জটিল মডেলগুলিতে ঘটে যা ডেটার গোলমালও শিখে ফেলে।

3.4 Cross validation

Definition 0.7. Cross validation, sometimes called rotation estimation, is a model validation technique for assessing how the results of a statistical analysis will generalize to an independent data set[3].

Common types of cross-validation:

1. K-fold cross-validation. In k-fold cross-validation, the original sample is randomly partitioned into k equal size subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining k − 1 subsamples are used as training data.
2. 2-fold cross-validation. Also, called simple cross-validation or holdout method. This is the simplest variation of k-fold cross-validation, k=2.
3. Leave-one-out cross-validation(LOOCV). k=M, the number of original samples.

[3] http://en.wikipedia.org/wiki/Cross-validation_(statistics)

3.5 Model selection

When we have a variety of models of different complexity (e.g., linear or logistic regression models with different degree polynomials, or KNN classifiers with different values of K), how should we pick the right one? A natural approach is to compute the misclassification rate on the training set for each method.