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 মেশিন লার্নিং মডেলের তিনটি ধাপ

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 ভাল classifier এবং কোনটা খারাপ classifer সেটা বুঝার জন্যে evaluation function (objective function or risk function) ব্যবহার করা হয়। মডেল যখন classifier-দের মধ্যে পার্থক্য করতে চায়, তখন এই evaluation function একটা স্কোর বা ভ্যালু রিটার্ন করে যেটার মাধ্যমে learning algorithm ধরতে পারে কোন classifier সবচেয়ে ভাল।

2.2.1 Loss function & 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 বুঝাবে যে একটা classifer 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 classifer-কে 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) এর জন্য ভ্যালু y_hat কে প্রেডিক্ট করতে লস হবে $L(y, y_hat)$

loss function

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

- $Y \times Y$: এখানে বোঝানো হচ্ছে যে লস ফাংশন সকল সম্ভাব্য label বা output এর সেট থেকে দুইটা আর্গুমেন্ট নেয়;
  - y_i: ith ট্রেনিং এক্সাম্পলের আসল (actual) লেবেল।
  - $\hat{y}$: মডেল যে প্রেডিকশন দিয়েছে।
- $R \geq 0$: লস ফাংশনের আউটপুট হলো একটা নন-নেগেটিভ রিয়েল সংখ্যা (যেটা $R \geq 0$ দিয়ে প্রকাশ করা হয়)। এই সংখ্যাটা দেখায় কতটা এরর বা "লস" আছে actual বা আসল লেবেল y_i আর প্রেডিক্ট করা লেবেল $\hat{y}$ -এর মধ্যে। আমাদের লক্ষ্য হলো এই মানটা যতটা সম্ভব কমিয়ে আনা।

নিচে কিছু সাধারণ লস ফাংশনের উদাহরণ দেওয়া হলো:

☐ 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)) : একটা ইন্ডিকেটর ফাংশন, যেখানে যদি actual লেবেল y_i আর প্রেডিক্টেড লেবেল $f(X)$ না মিলে, তাহলে আউটপুট হবে 1, আর মিললে আউটপুট হবে 0।
- এটা খুবই সাধারণ একটা লস ফাংশন, যেটা শুধু চেক করে প্রেডিকশন ঠিক আছে নাকি ভুল, ভুলের পরিমাণ গুরুত্ব দেয় না।

☐ Quadratic loss function $L(Y, f(X)) = (Y - f(X))^2$

- $Yf(X)$ হলো আসল লেবেল Y আর প্রেডিক্টেড লেবেল $f(X)$ এর মধ্যে পার্থক্য স্কয়ার করা হচ্ছে নেগেটিভ ভ্যালুকে পরিহার করার জন্যে।
- Mean Squared Error নামেই চেনা এই লস ফাংশন regression প্রবলেমের জন্য অনেক বেশি ব্যবহৃত হয় যেখানে

---

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

আসল আর প্রেডিক্টেড ভ্যালুর মধ্যে যত বেশি পার্থক্য, তত বেশি লস।

- □ Absolute loss function $L(Y, f(X)) = |Y - f(X)|$
  - $|Y - f(X)|$ হলো আসল লেবেল y এবং আর প্রেডিক্টেড লেবেল $f(X)$ এর মধ্যে অ্যাবসোলিউট পার্থক্য।
  - এই ফাংশন আসল আর প্রেডিক্টেড ভ্যালুর মধ্যে সরাসরি পার্থক্য দেয়, স্কোয়ার না করে। যেসকল ক্ষেত্রে average loss বা error দেখার দরকার পড়ে সেখানে আমরা এই লস ফাংশন ব্যবহার করে থাকি।
- □ Logarithmic loss function $L(Y, P(Y|X)) = -\log P(Y|X)$
  - $P(Y|X)$ হলো X ইনপুট দেওয়ার পর আসল লেবেল y পাওয়ার প্রেডিক্টেড প্রোবাবিলিটি। $-\log P(Y|X)$ প্রেডিক্টেড প্রোবাবিলিটির লোগারিদম নিয়ে তার নেগেটিভ নেওয়া হয়, কারণ আমরা চাই high প্রোবাবিলিটি এর জন্যে যেন কম loss value আসে।
  - যখন $P(Y|X)$ এর probability score low, Logarithmic loss function তখন ভুল prediction কে penalize করে; এই কারণে এই loss function classifcation প্রবলেমে ব্যবহার করা হয়।

Definition 0.2. $R_{\exp}(f)$ হচ্ছে expected loss বা risk function; যা দ্বারা বোঝায় যে কোনো ফাংশন f ব্যবহার করে প্রেডিকশন করার সময় কতটা error (loss) থাকতে পারে

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

- $L(Y, f(X))$: loss function
- $E[]$:Expectation অপারেটর, probability distribution এর উপর ভিত্তি করে ফাংশনের average expected value নির্ণয় করে
- $P(x, y)$:input data X এবং তার label Y joint probaility distribution, যেটা ইনপুট-আউটপুট এর সভাব্যতা বের করে
- এক্সপেক্টেড average value নির্ণয় করার জন্যে integral $\int mathrmdx\mathrm{d}y$ ব্যবহার করা হচ্ছে; সম্ভাব্য সকল data point X এবং Y এর loss value এর average বের করছে probability distribution এর উপর ভিত্তি করে ।

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

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

- $R_{\mathrm{emp}}(f)$ হচ্ছে এমপিরিকাল রিস্ক বা এমপিরিকাল লস, যেটার মাধ্যমে জানা যায় একটা মডেল f শুধু ট্রেনিং ডেটায় কতোটা ভালো কাজ করেছে।
- গড় মান পাবার জন্যে total loss এর average বের করছি
- $L(y_i, f(x_i))$ data পয়েন্ট y_i, x_i এর উপর loss function apply করা হচ্ছে, x_i যেখানে ইনপুট এবং y_i হলো আসল আউটপুট, আর $f(x_i)$ হলো প্রেডিক্টেড আউটপুট।

এই ফাংশনকে empirical loss বা empirical risk-ও বলা হয়ে থাকে.

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

2.2.2 ERM 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)

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

- N সংখ্যক data থেকে প্রাপ্ত লস ভ্যালু এর অ্যাভারেজ ভ্যালু গুলোর মধ্যে মিনিমাম যেটা পাব হাইপথিসিস স্পেইস থেকে সেটা হবে R_{emp}

Definition 0.5. Structural risk

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

- □ $J(f)$ এমন একটি টার্ম যা বেশি জটিল মডেলকে শাস্তি (penalty) দেয়

---

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

- λ দ্বারা নির্ধারিত হয় কতটুকু শাস্তি বা পেনলাইজ করা হবে কমপ্লেক্সিটি লেভেল ঠিক রাখার জন্যে

Definition 0.6. SRM(Structural risk minimization) SRM-এর লক্ষ্য হল সমস্ত সম্ভাব্য ফাংশন F থেকে এমন একটি ফাংশন f খুঁজে বের করা যা এম্পিরিকাল রিস্ক এবং মডেল জটিলতার সমষ্টিকে সর্বনিম্ন করে।

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

- $R_{\mathrm{srm}}(f)$ মডেলের স্ট্রাকচারাল রিস্ক, যা এম্পিরিকাল রিস্ক এবং মডেলের জটিলতার সমন্বয়ে গঠিত।
- $\frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i))$: এম্পিরিকাল রিস্ক, যা ট্রেনিং ডেটায় মডেলের পারফরম্যান্সের গড় ত্রুটি মাপা হয়।
- $\lambda J(f)$ রেগুলারাইজেশন টার্ম, যা মডেলের জটিলতাকে শাস্তি দেয় এবং ওভারফিটিং এড়াতে সাহায্য করে।

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 ব্যাসিক কনসেপ্ট

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 একটি সহজ নন-প্যারামেট্রিক ক্লাসিফায়ার: k nearest algorithm

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

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।

3.2.2 Evaluation

কোনো ট্রেনিং এর প্রয়োজন হয় না।

3.2.3 Optimization

কোনো ট্রেনিং এর প্রয়োজন হয় না।

3.3 Overfitting

ওভারফিটিং হয় যখন একটি মডেল ট্রেইনড ডেটাতে খুব ভালো কাজ করে কিন্তু নতুন ডেটাতে খারাপ করে। এটি খুব জটিল মডেলগুলিতে ঘটে (complex model) যা ডেটার noise-ও শিখে ফেলে।

3.4 Cross validation

Definition 0.7. Cross validation (অনেক সময় যাকে rotation estimation বলা হয়) হল একটি model validation পদ্ধতি, যা একটি statistical analysis ফলাফল অন্য ডেটাসেটে কতটা ভালোভাবে প্রয়োগ করা যায় তা নির্ধারণ করতে ব্যবহৃত হয় [3]।

সাধারণ কিছু cross-validation এর ধরন:

a. K-fold cross-validation: এই পদ্ধতিতে, মূল sample-কে এলোমেলোভাবে k সমান আকারের subsample এ ভাগ করা হয়। এই k টি subsample এর মধ্যে একটি subsample মডেলটিতে test করার জন্য validation ডেটা হিসেবে ব্যবহৃত হয়, আর বাকি k - 1 subsample গুলো মডেলটি training এর জন্য ব্যবহৃত হয়।

---

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

b. 2-fold cross-validation: simple k-fold cross-validation বলা হয় , যেখানে k=2; holdout method ও বলা হয়।
c. Leave-one-out cross-validation(LOOCV): এখানে k = M, অর্থাৎ মূল sample এর সংখ্যা যত।

3.5 Model selection

যখন আমাদের হাতে বিভিন্ন কমপ্লেক্স (যেমন: ভিন্ন ভিন্ন degree এর polynomials সহ linear বা logistic regression মডেল, বা ভিন্ন ভিন্ন K এর মান সহ KNN classifiers) মডেলের অনেকগুলো বিকল্প থাকে, তখন আমরা সঠিক মডেলটি কীভাবে নির্ধারন করব? একটি সাধারণ উপায় হল, প্রতিটি পদ্ধতির জন্য training set এ misclassification rate হিসাব করা।