মেশিন লার্নিং-এ চ্যালেঞ্জসমূহ।

মেশিন লার্নিং-এর ১০টি গুরুত্বপূর্ণ চ্যালেঞ্জ

১. ডেটা সংগ্রহ **(Data Collection)** মেশিন লার্নিং (ML) ডেটা থেকে শেখার উপর নির্ভরশীল। আপনি যখন ছোটখাটো প্রজেক্ট করেন, তখন ডেটা সহজেই CSV ফাইল বা Kaggle-এর মতো জায়গা থেকে পাওয়া যায়। কিন্তু যখন কোনো কোম্পানিতে বাস্তব ML প্রজেক্টে কাজ করা হয়, তখন ডেটা সংগ্রহ করা বা একত্রিত করা বেশ কঠিন কাজ হয়। ডেটা সংগ্রহের জন্য API ব্যবহার করা বা ওয়েব স্ক্র্যাপিং করার মতো উপায় অবলম্বন করতে হতে পারে, কিন্তু সেখানেও বিভিন্ন সমস্যা থেকে যায়।

২. অপর্যাপ্ত ডেটা ও লেবেলযুক্ত ডেটা **(Insufficient Data and Labeled Data)** এই চ্যালেঞ্জটি দুটি ভাগে বিভক্ত এবং উভয়ই ডেটা সম্পর্কিত:

- অপর্যাপ্ত ডেটা**:** বেশি ডেটা থাকলে মডেলের পারফরম্যান্স সাধারণত আরও ভালো হয়, এমনকি যদি অ্যালগরিদমটি খুব উন্নত না-ও হয়। যদি আপনার কাছে পর্যাপ্ত পরিমাণে ডেটা থাকে (huge amount), তবে আপনি কোন অ্যালগরিদম ব্যবহার করছেন তা খুব বেশি গুরুত্বপূর্ণ নয়—এই ঘটনাটিকে **"**আনরিজনেবল এফেক্টিভনেস অফ ডেটা**"** বলা হয়। তবে, সবার কাছে সবসময় এতো বেশি ডেটা থাকে না। ডেটা কম থাকলে মডেলের পারফরম্যান্স দুর্বল হতে পারে।
- লেবেলযুক্ত ডেটার অভাব**:** যদিও আপনি সহজেই প্রচুর ডেটা (যেমন ইমেজ) সংগ্রহ করতে পারেন (যেমন API বা ওয়েব স্ক্র্যাপিং ব্যবহার করে), কিন্তু প্রতিটি ডেটা সঠিকভাবে লেবেল করা (যেমন ছবিতে কোনটি বিড়াল বা কোনটি কুকুর তা চিহ্নিত করা) একটি কষ্টসাধ্য এবং সময়সাপেক্ষ কাজ।

৩. অপ্রতিনিধিত্বমূলক ডেটা **(Non-Representative Data)** যদি আপনার ট্রেনিং ডেটা সমস্যার সম্পূর্ণ চিত্র সঠিকভাবে তুলে না ধরে, তবে মডেলের সিদ্ধান্ত ভুল হতে পারে।

- উদাহরণস্বরূপ, যদি আপনি ক্রিকেট বিশ্বকাপ কে জিতবে তা জানতে চান, আর শুধুমাত্র ভারতে সার্ভে করান, তবে পক্ষপাতদুষ্ট উত্তর আসার সম্ভাবনা বেশি থাকবে (অধিকাংশ মানুষ বলবে ভারত)। কারণ আপনি একটি সীমিত জায়গা থেকে ডেটা নিয়েছেন, যা পুরো সমস্যার সঠিক প্রতিনিধিত্ব করছে না। একে স্যাম্পলিং নয়েজ **(Sampling Noise)** বলা হয়।
- যদি আপনি পর্যাপ্ত সংখ্যক ডেটা সংগ্রহ করেও সঠিক উপায়ে নমুনা সংগ্রহ না করেন (যেমন, সব দেশ থেকে ডেটা নিলেন, কিন্তু সবাই ভারতীয়), তবে সেটাকে স্যাম্পলিং বায়াস **(Sampling Bias)** বলা হয়। সঠিক ফলাফল পেতে গেলে প্রতিটি গোষ্ঠীকে সমান সুযোগ দিয়ে ডেটা সংগ্রহ করা উচিত।

৪. নিম্ন মানের ডেটা **(Poor Quality Data)** ডেটাতে অনেক ত্রুটি থাকতে পারে, যেমন গ্লাস (আবর্জনা), মিসিং ভ্যালু, আউটলায়ার বা বিভিন্ন ফরম্যাটের ভ্যালু। ডেটা বিজ্ঞানীর মোট সময়ের প্রায় ৬০**% (**যেমন এক বছরের প্রজেক্টে প্রায় আট মাস**)** ডেটা পরিষ্কার করে তার মান উন্নত করার কাজে ব্যয় হয়। ডেটার মান ভালো না হলে মেশিন লার্নিং অ্যালগরিদম ভালো ফল দিতে পারে না, কারণ এটি ডেটার উপর সম্পূর্ণ নির্ভরশীল।

৫. অপ্রাসঙ্গিক ফিচার **(Irrelevant Features)** ডেটার যে কলামগুলি আপনার ভবিষ্যদ্বাণীতে কোনো অবদান রাখে না, সেগুলিকে অপ্রাসঙ্গিক ফিচার বলা হয়। একটি বিখ্যাত উক্তি হল, **"**গার্বেজ ইন**,** গার্বেজ আউট**"**—অর্থাৎ, আপনি যদি অপ্রয়োজনীয় ফিচার ইনপুট দেন, তবে আউটপুটও খারাপ হবে।

- উদাহরণস্বরূপ, ম্যারাথনে কে অংশগ্রহণ করবে তা ভবিষ্যদ্বাণী করার সময়, বয়স, ওজন, উচ্চতা প্রাসঙ্গিক হলেও 'লোকেশন' বা অবস্থান সাধারণত অপ্রাসঙ্গিক হয় এবং এটিকে বাদ দেওয়াই শ্রেয়।

- অনেক সময় দুটি ফিচারকে একত্রিত করে একটি নতুন একক ফিচার তৈরি করা হয় (যেমন ওজন ও উচ্চতা থেকে BMI তৈরি করা)। এটিকে ফিচার ইঞ্জিনিয়ারিং **(Feature Engineering)** বলা হয়।

৬. ওভারফিটিং **(Overfitting)** ওভারফিটিং তখনই ঘটে যখন একটি মেশিন লার্নিং মডেল ট্রেনিং ডেটা খুব ভালোভাবে মুখস্থ করে নেয়, কিন্তু সেই ডেটার পিছনের আসল গল্প বা ধারণা বুঝতে পারে না। ফলে যখন নতুন ডেটা আসে, তখন মডেলের পারফরম্যান্স খারাপ হয়। একটি মডেল যখন ট্রেনিং ডেটার প্রতিটি পয়েন্টকে খুব কাছ থেকে অনুসরণ করে একটি জটিল ফাংশন তৈরি করে, তখন এটি ওভারফিটিং নির্দেশ করে।

৭. আন্ডারফিটিং **(Underfitting)** এটি ওভারফিটিং-এর ঠিক বিপরীত। এখানে মডেল ডেটার অন্তর্নিহিত গঠনকে ধরতে ব্যর্থ হয় এবং খুবই সরল একটি মডেল তৈরি করে। এর ফলে মডেলটি ট্রেনিং ডেটা বা নতুন ডেটা, কোনোটির উপরেই ভালো ফলাফল দিতে পারে না।

৮. সফটওয়্যার ইন্টিগ্রেশন **(Software Integration)** একটি ML প্রজেক্টের চূড়ান্ত লক্ষ্য হলো এটিকে একটি সফটওয়্যারের অংশ হিসেবে ব্যবহারকারীর কাছে পৌঁছে দেওয়া (যেমন সুপারিশ সিস্টেম বা পূর্বাভাস অ্যালগরিদম)। ML মডেলকে বিভিন্ন প্ল্যাটফর্মে (উইন্ডোজ, অ্যান্ড্রয়েড, লিনাক্স, সার্ভার) ইন্টিগ্রেট করা কঠিন। বর্তমানে ML একটি নতুন প্রযুক্তি হওয়ায়, অনেক সফটওয়্যার প্ল্যাটফর্ম এখনও স্থিতিশীলভাবে ML মডেল সমর্থন করে না (যেমন জাভা বা পুরোনো জাভাস্ক্রিপ্ট)। সকল প্ল্যাটফর্মে মডেলটি সঠিকভাবে কাজ করছে তা নিশ্চিত করা একটি বড় চ্যালেঞ্জ।

৯. অফলাইন লার্নিং এবং ডিপ্লয়মেন্ট **(Offline Learning and Deployment)**

- অফলাইন লার্নিং**:** এই প্রক্রিয়ায় মডেল একবার ট্রেনিং নিয়ে সার্ভারে আপলোড করা হয় এবং স্বয়ংক্রিয়ভাবে আপডেট হয় না। মডেলকে আপডেট করতে হলে তাকে অফলাইনে এনে নতুন ডেটা দিয়ে পুনরায় ট্রেনিং করাতে হয় এবং আবার সার্ভারে আপলোড করতে হয়। এর বিপরীতে, ক্রমাগত মডেল আপডেট হওয়ার প্রক্রিয়াকে অনলাইন লার্নিং বলে, যা তুলনামূলকভাবে কঠিন।
- ডিপ্লয়মেন্ট**:** মডেলকে প্রোডাকশনে নিয়ে যাওয়া এবং স্থাপন করা (deployment) খুবই কঠিন কাজ, এমনকি AWS-এর মতো শীর্ষস্থানীয় পরিষেবা প্রদানকারী থাকলেও। রিয়েল টাইমে পর্যবেক্ষণ এবং প্রোডাকশন কাজ করার সময় দেখা যায় যে সফটওয়্যার সাইডের তুলনায় ML ডিপ্লয়মেন্টে আরও উন্নতির প্রয়োজন।

১০. খরচ বা লুকানো ব্যয় **(Cost/Hidden Costs)** যখন একটি বড় স্কেলে (যেমন, ১ লক্ষ ব্যবহারকারীর জন্য) ML মডেল ডিপ্লয় করা হয়, তখন অপ্রত্যাশিতভাবে বিপুল পরিমাণ লুকানো খরচ আসতে পারে। যেহেতু সার্ভারে ML প্রযুক্তির অপটিমাইজেশন এখনও পুরোপুরি হয়নি, তাই এই খরচ নিয়ন্ত্রণ করা একটি বড় চ্যালেঞ্জ। এই খরচ নিয়ে "The Cost of AI" নামে একটি পেপার রয়েছে।

এই চ্যালেঞ্জগুলি মোকাবিলার জন্য একটি নতুন ক্ষেত্র তৈরি হয়েছে, যার নাম **MLOps (Machine Learning Operations)**। এই ক্ষেত্রটি সফটওয়্যার প্রোডাক্টকে সার্ভারে স্থাপন করা এবং তার খরচ ইত্যাদি পরিচালনা করার সাথে জড়িত এবং এটি বর্তমানে একটি দ্রুত বর্ধনশীল ক্ষেত্র।