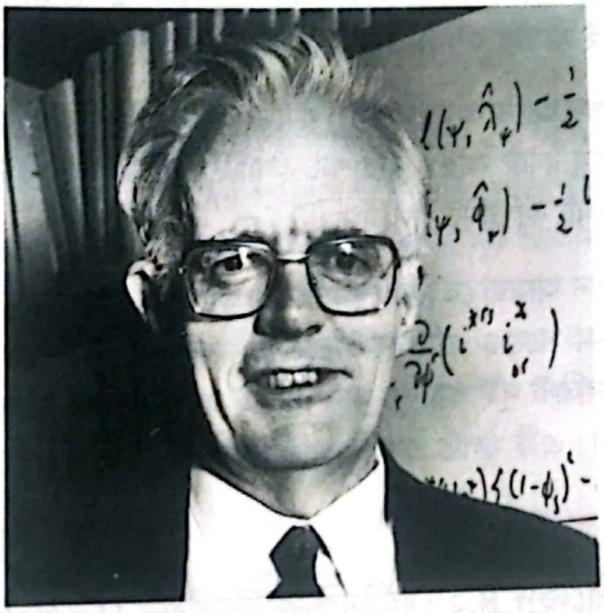
অধ্যায় ৪ : লজিস্টিক রিগ্রেশন (Logistic Regression)



ছবি 4.1 : ডেভিড কক্স (David Cox, 1924)

লজিস্টিক রিগ্রেশন (Logistic Regression) পদ্ধতিটি প্রথমবারের মতো ব্যবহার করেন পরিসংখ্যানবিদ ডেভিড কক্স (David Cox, 1924)। তিনি 1924 সালে বার্মিংহামে জন্মগ্রহণ করেন। তিনি সেইন্ট জনস কলেজ থেকে গণিত নিয়ে পড়াশোনা করেছেন এবং পরবর্তী সময়ে লিডস বিশ্ববিদ্যালয় থেকে পিএইচডি সম্পন্ন করেছেন।

তিনি সারা জীবনে অসংখ্য পুরস্কারে ভূষিত হয়েছেন। এর মধ্যে উল্লেখযোগ্য হলো 'গাই মেডাল' পুরস্কার, যেটি তিনি পেয়েছিলেন রয়্যাল সোসাইটি অব স্ট্যাটিসটিকস থেকে। সেই সঙ্গে রানি দ্বিতীয় এলিজাবেথ তাঁকে 1985 সালে নাইটহুড প্রদান করেন। তিনি ছিলেন ব্রিটিশ একাডেমির একজন ফেলো।

পরিচ্ছেদ ৪.১ : কন্টিনিউয়াস ও ডিসক্রিট ডেটা (Continuous and Discrete Data)

এই অধ্যায় শুরু করার আগে আমরা কন্টিনিউয়াস ডেটা (Continuous Data) ও ডিসক্রিট ডেটা (Discrete Data) সম্পর্কে একটু জানব। সহজ করে বলতে গেলে, কন্টিনিউয়াস ডেটা হচ্ছে একটি নির্দিষ্ট সীমার মধ্যেকার যে-কোনো সংখ্যা। যেমন, আমরা যদি চিন্তা করি যে 0 ও 3-এর ভেতরে যে-কোনো সংখ্যা এবং যদি বলে দিই আমরা কন্টিনিউয়াস মান নিয়ে কাজ করছি, তাহলে আমরা এই 0 ও 3-এর ভেতরের যে-কোনো সংখ্যা নিয়ে কাজ করতে পারি, হোক সেটি পূর্ণ সংখ্যা (শুধু 0, 1, 2, 3 ইত্যাদি), কিংবা দশমিকযুক্ত সংখ্যা (0.001, 0.0000345, 1.234434456 ইত্যাদি)। বুঝতেই পারছেন, যদি আমরা কন্টিনিউয়াস ডেটা নিয়ে কাজ করি, তাহলে 0 ও 3-এর ভেতরে অসীমসংখ্যক সংখ্যা পাওয়া সম্ভব।

আবার, আমরা যদি ডিসক্রিট ডেটা (Discrete Data) নিয়ে কাজ করতে যাই, সেটি হবে কন্টিনিউয়াস ডেটার ঠিক বিপরীত। কন্টিনিউয়াস ডেটার ক্ষেত্রে যেমন আমরা একটি সীমার মধ্যেকার যে-কোনো সংখ্যা নিতে পারতাম, ডিসক্রিট ডেটার ক্ষেত্রে আমরা তা পারব না। আমরা শুধু আমাদের সীমার ভেতরের নির্দিষ্ট কিছু সংখ্যা নিয়ে কাজ করতে পারব। যেমন, 0 ও 3-এর এই সীমায় শুধু 0, 1, 2, 3 এই চারটি পূর্ণসংখ্যা নিয়েই কাজ করতে পারব।

এখন আসি আমাদের মূল কাজে। আমরা লিনিয়ার রিগ্রেশনের ক্ষেত্রে সব সময় কন্টিনিউয়াস ডেটা নিয়ে কাজ করি। রিগ্রেশন কথাটির মানেই অনেকটা এরকম বোঝায় যে আউটপুট হবে কোনো একটি কন্টিনিউয়াস মান।

এখন আমরা যে রিগ্রেশন শিখব, তার নাম লজিস্টিক রিগ্রেশন। কিন্তু, নামের মধ্যে রিগ্রেশন শব্দ থাকা সত্ত্বেও এটি কন্টিনিউয়াস মান নয়, ডিসক্রিট মান আউটপুট দেয়। এটি কন্টিনিউয়াস ও ডিসক্রিট দুই ধরনের মান নিয়েই কাজ করতে পারে, কিন্তু এর আউটপুট সব সময় হয় ডিসক্রিট মান। এটি মূলত একটি ক্লাসিফিকেশন অ্যালগরিদম।

পরিচ্ছেদ ৪.২ : লজিস্টিক রিগ্রেশন (Logistic Regression)-এর হাইপোথিসিস

লজিস্টিক রিগ্রেশনে আমাদের লক্ষ্য হবে, হাইপোথিসিস থেকে পাওয়া আউটপুট যেন 0 থেকে 1-এর ভেতরে থাকে। আউটপুট যদি 0.5-এর চেয়ে ছোটো হয়, তাহলে আমাদের লজিস্টিক রিগ্রেশনের সর্বশেষ আউটপুট হবে 0, আর যদি তা 0.5-এর সমান বা এর চেয়ে বড়ো হয়, তাহলে লজিস্টিক রিগ্রেশনের সর্বশেষ আউটপুট হবে 1।

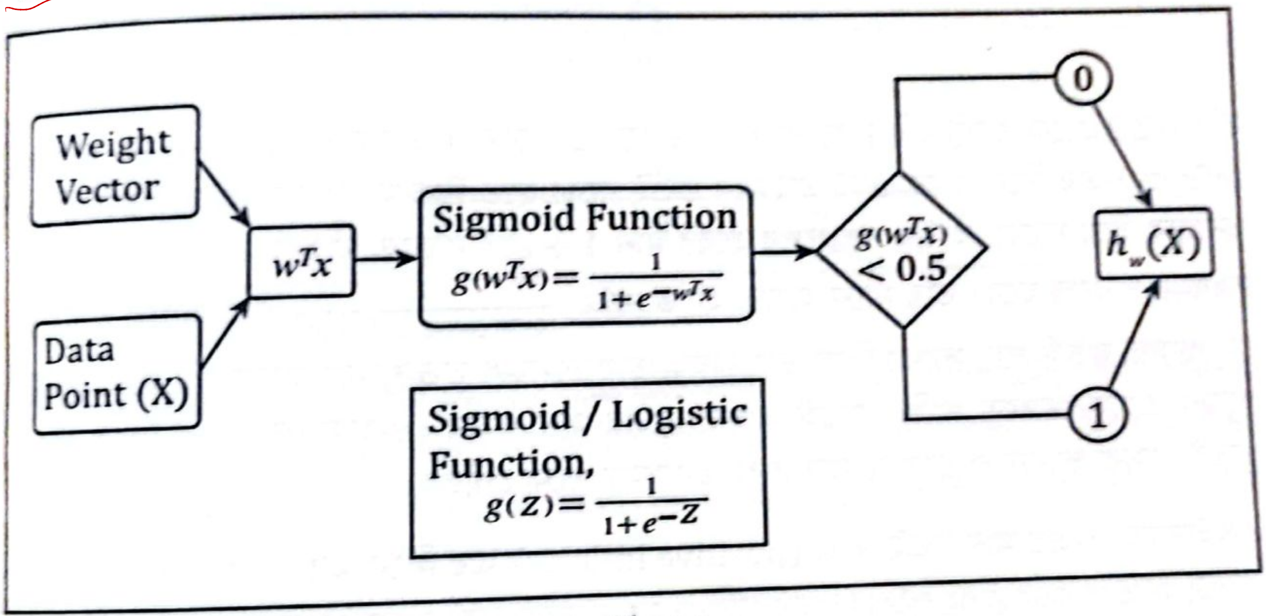
এর মান 0 থেকে 1-এর ভেতরে রাখার পেছনে যুক্তি হচ্ছে, এটি মূলত সম্ভাব্যতার মান আউটপুট হিসেবে দেয়। ধরুন, আপনি একটি প্রাণীর ছবি ইনপুট দিয়েছেন এবং লজিস্টিক রিগ্রেশনের মাধ্যমে বের করতে চাইছেন যে, সেটি একটি কুকুরের ছবি কি না। যদি সেটি কুকুরের ছবি হয়, তাহলে হিসাবনিকাশ করে হাইপোথিসিস আউটপুট দেবে 0.5 থেকে 1-এর মধ্যবর্তী কোনো একটি মান যেটি বোঝাচ্ছে যে ছবিটি কুকুরের হওয়ার সম্ভাব্যতা বেশি। আর যেহেতু, এই হাইপোথিসিসের মান 0.5 থেকে 1-এর মধ্যে আছে, সেহেতু অ্যালগরিদমের ফাইনাল আউটপুট হবে 1, অর্থাৎ ছবিটি একটি কুকুরের ছবি।

আর যদি সেটি না হয়, অর্থাৎ ছবিটি যদি কুকুরের না হয়ে অন্য কিছুর হয়, তাহলে আপনার হাইপোথিসিস হিসাবনিকাশ করে আউটপুট দেবে 0 থেকে 0.5-এর মধ্যবর্তী কোনো একটি সংখ্যা (0.5 হতে পারবে না, এর চেয়ে ছোটো হতে হবে কিন্তু) যেটি বোঝাচ্ছে যে ছবিটি কুকুরের হওয়ার

সম্ভাব্যতা অনেক কম এবং সেটি থেকে পরবর্তী সময়ে অ্যালগরিদমের ফাইনাল আউটপুট হবে 0 অর্থাৎ ছবিটি কুকুরের নয়।

একটু খেয়াল রাখবেন, এখানে আমরা দুই ধরনের আউটপুটের কথা বলেছি, একটি হাইপোথিসিসের আউটপুট, আরেকটি লজিস্টিক রিগ্রেশনের আউটপুট। দুটোর মধ্যে পার্থক্য কী সেটি এখনই পরিষ্কার করে দিচ্ছি।

লিনিয়ার রিগ্রেশনের ক্ষেত্রে, আমাদের হাইপোথিসিসের সমীকরণে প্যারামিটার ও x-এর বিভিন্ন মান বসিয়ে হিসাব করে যে ফলাফল আসত, সেটিই ছিল আমাদের সর্বশেষ আউটপুট। লজিস্টিক রিগ্রেশনের ক্ষেত্রে তা হবে না। হাইপোথিসিসের আউটপুট ও অ্যালগরিদমের আউটপুট আলাদা হবে। নিচের ফ্লো-চার্টটি লক্ষ করুন (ফ্লোচার্ট 4.2.1):



ফ্লোচার্ট 4.2.1

লজিস্টিক রিগ্রেশনে আমরা আগের মতোই w^Tx-এর মান হিসাব করব। এর পরে, এই w^Tx-এর মান আমরা একটি সিগময়েড (Sigmoid) বা লজিস্টিক (Logistic) ফাংশনে ইনপুট হিসেবে দেব। সিগময়েড/লজিস্টিক ফাংশনটি হবে এরকম –

$$g(z) = \frac{1}{1+e^{-z}}$$

এখানে $z = w^Tx$ হবে। ছবি 4.2.2-তে একটি সিগময়েড ফাংশন দেখানো হলো। এটি থেকে আমরা যে আউটপুট পাব, সেটিই আমাদের হাইপোথিসিস $h_w(x)$-এর মান। এখন আমরা দেখব, যে এই সিগময়েড ফাংশন যে মান দিচ্ছে, সেটি 0.5-এর চেয়ে ছোটো কি না (এতটুকু নিশ্চিত থাকুন যে সিগময়েড ফাংশনের আউটপুটের মান 0 থেকে 1-এর ভেতরেই থাকবে)। যদি ছোটো

হয়, তাহলে আমাদের $h_w(x)$ হবে 0। আর যদি না হয়, অর্থাৎ যদি 0.5-এর চেয়ে বড়ো বা এর সমান হয়, তাহলে আমাদের $h_w(x)$ হবে 1।

নিচে আপনাদের বোঝার সুবিধার্থে একটি টেবিল আকারে দেওয়া হলো :

$w^T x$	$g(w^T x) = \frac{1}{1+e^{-w^T x}}$ (এটি হাইপোথিসিস থেকে প্রাপ্ত মান)	$h_w(x)$ (এটি লজিস্টিক রিগ্রেশনের ফাইনাল আউটপুট)
0	0.5	1
> 0	> 0.5	1
< 0	< 0.5	0

টেবিল 4.2.1

ছবি 4.2.2 থেকে একটু যদি বিশ্লেষণ করি, তাহলে দেখবেন, z-এর মান যতই $-\infty$ (negative infinity)-এর দিকে যাবে, e^{-z}-এর মান ততই বড়ো হতে থাকবে (ঋণাত্মকে ঋণাত্মকে মিলে ধনাত্মক হয়ে যাবে), অর্থাৎ ভগ্নাংশের হরের মান $1 + e^{-z}$ বাড়তে থাকবে এবং একটি বিশাল সংখ্যায় পরিণত হবে। তার মানে আমরা হাতে পাচ্ছি, $\frac{1}{\text{বিশাল কোনো একটি সংখ্যা}}$ এবং আমরা জানি 1 কে আমরা যত বিশাল সংখ্যা দিয়ে ভাগ করব, তার ভাগফল তত শূন্যের কাছাকাছি হবে। সেটিই আমরা গ্রাফে দেখতে পাচ্ছি। যতই আমরা z-এর মান কমাব (x-অক্ষ বরাবর বাঁ দিকে যাব) ততই আমরা শূন্যের কাছাকাছি যাব এবং একসময় শূন্যে পৌঁছব।

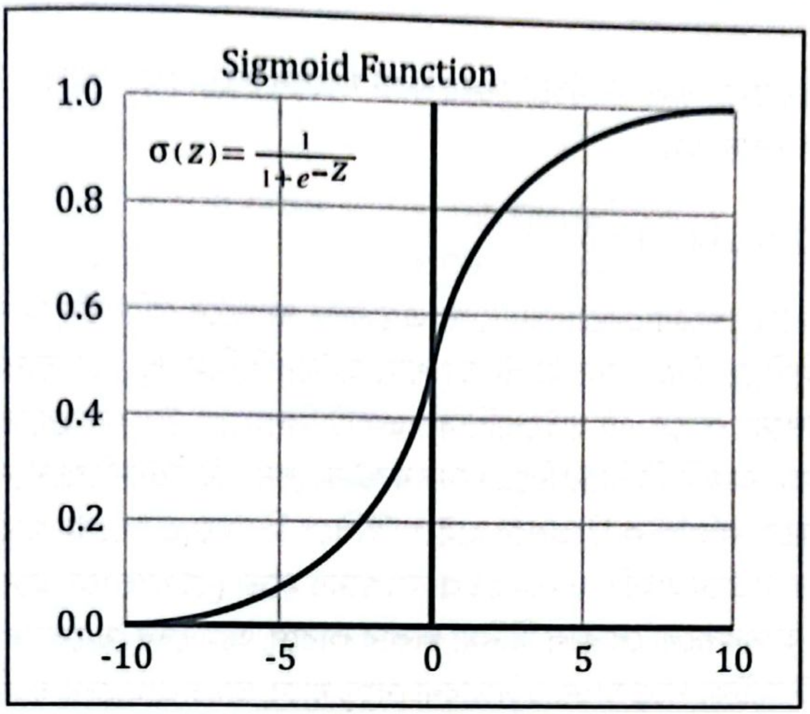
একইভাবে, z-এর মান যতই $+\infty$ (positive infinity)-এর দিকে যাবে, e^{-z}-এর মান ততই ছোটো হতে থাকবে এবং শূন্যের কাছাকাছি যেতে থাকবে, অর্থাৎ ভগ্নাংশের হরের মান $1 + e^{-z}$ তখন 1-এর কাছাকাছি যেতে থাকবে যেহেতু,

$$1 + \text{খুব ক্ষুদ্র কোনো সংখ্যা (শূন্যের কাছাকাছি)} \cong 1$$

তার মানে এবারে আমরা হাতে পাচ্ছি, $\frac{1}{\text{1-এর খুব কাছাকাছি কোনো সংখ্যা (যেমন 1.00001)}}$ যার ফলে শেষমেশ আমাদের ভাগফল দাঁড়াচ্ছে 1-এর খুব কাছাকাছি কোনো মান অর্থাৎ, 1-ই ধরে নেওয়া যায়। সেটিই আমরা গ্রাফে দেখতে পাচ্ছি। যতই আমরা z-এর মান বাড়াব (x-অক্ষ বরাবর ডানদিকে যাব), ততই আমরা 1-এর কাছাকাছি যাব এবং একসময় 1-এ পৌঁছব।

এখন শেষ কথা হচ্ছে – এটাকে তাহলে লজিস্টিক ক্লাসিফিকেশন কেন বলা হলো না, কেন রিগ্রেশনই বলা হলো একে? এর উত্তর খুব সহজ। একটু খেয়াল করলে দেখবেন, আমরা আমাদের হাইপোথিসিস থেকে যে মান পাচ্ছি, সেটি কিন্তু একটি কন্টিনিউয়াস মান, তাই না? আর আমরা

আউটপুট হিসেবে কন্টিনিউয়াস মান কোথায় পেতাম? রিগ্রেশনের ক্ষেত্রে, ঠিক? আর তা-ই একে লজিস্টিক রিগ্রেশন বলা হয়। কিন্তু এটি মূলত একটি ক্লাসিফিকেশন অ্যালগরিদম।



গ্রাফ 4.2.2

লজিস্টিক রিগ্রেশনের আরেকটি বৈশিষ্ট্য হচ্ছে এটি একটি টু-ক্লাস ক্লাসিফিকেশন অ্যালগরিদম (Two-class classification algorithm)। অর্থাৎ, এটি কোন ছবি – কুকুরের কিংবা কুকুরের নয় – এ দুটোর মধ্যে তফাত বের করতে পারবে। কিংবা, ছবিটি কুকুরের নাকি বিড়ালের – এরকম দুটি ক্লাসের মধ্যে ক্লাসিফিকেশন করতে পারবে।

এখন যদি বলা হতো কুকুর, বিড়াল ও হরিণ –এই তিনটি ক্লাসের মধ্যে ক্লাসিফিকেশন করতে, তখন কিন্তু এই সাধারণ লজিস্টিক রিগ্রেশন সেটি পারত না। তার জন্য দরকার হতো মাল্টিনোমিয়াল লজিস্টিক রিগ্রেসন (Multinominal Logistic Regression), যেটি এই সাধারণ লজিস্টিক রিগ্রেশনেরই আরেকটি ভার্শন। সেটি আপাতত আমাদের বইয়ের এখতিয়ারের বাইরে থাকছে।

লজিস্টিক রিগ্রেশনের আরেকটি বৈশিষ্ট্য এখানে বলে রাখি। লজিস্টিক রিগ্রেশন তখনই কাজ করে, যদি আমাদের ডেটা Higher Dimension-এ নিয়ে গেলে সেটি লিনিয়ারলি সেপারেবল (Linearly Seperable) হয়। এর বিস্তারিত নিয়ে আর এখানে বলব না।

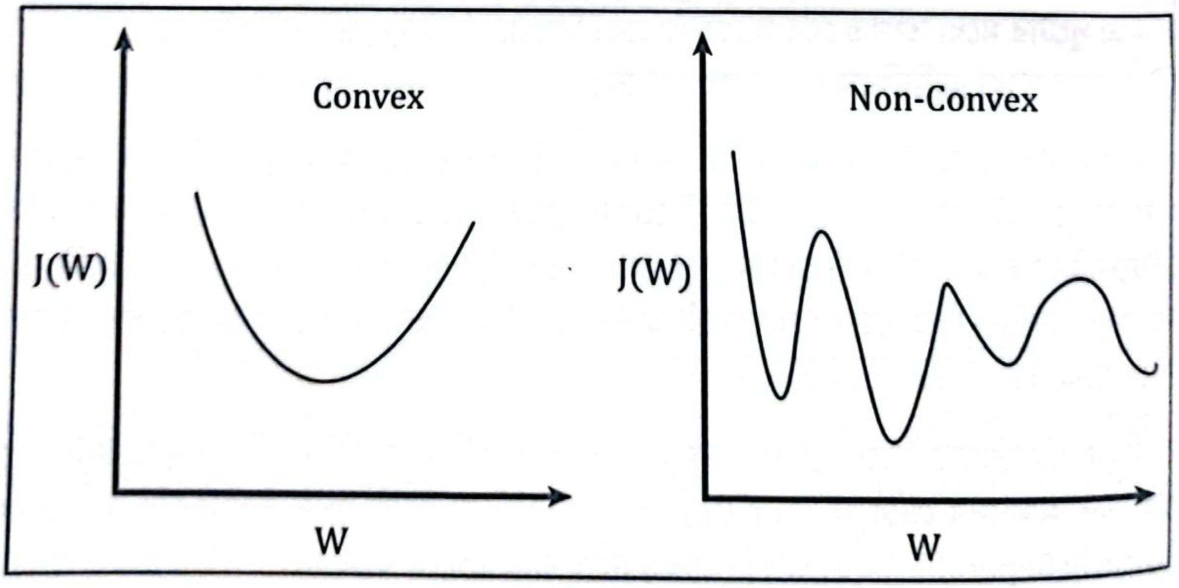
পরবর্তী অধ্যায়েই (সাপোর্ট ভেক্টর মেশিন – Support Vector Machine) পরিচ্ছেদ ৫.৩-এ এই সম্পর্কে বিস্তারিত বলা হয়েছে।

পরিচ্ছেদ ৪.৩ : লজিস্টিক রিগ্রেশনের কস্ট ফাংশন (Cost Function) ও প্রয়োগ পদ্ধতি

এখন আমরা আমাদের লজিস্টিক রিগ্রেশনের কস্ট ফাংশন নিয়ে কথা বলব। লিনিয়ার রিগ্রেশনে আমাদের কস্ট ফাংশন ছিল –

$$\frac{1}{2m}\sum_{i=1}^{m}\{h_w(x^{(i)}) - y^{(i)}\}^2$$

কিন্তু, লজিস্টিক রিগ্রেশনের ক্ষেত্রে আমাদের এই কস্ট ফাংশনে পরিবর্তন আসবে। এর কারণ হচ্ছে, আমরা লিনিয়ার রিগ্রেশনের কস্ট ফাংশনে হাইপোথিসিস $h_w(x)$ ব্যবহার করছি এবং লজিস্টিক রিগ্রেশনের ক্ষেত্রে এই হাইপোথিসিস একটি সিগময়েড ফাংশন হয়, তাই না (ফ্লো-চার্ট দ্রষ্টব্য)? এখন, আমরা যদি লিনিয়ার রিগ্রেশনের কস্ট ফাংশন লজিস্টিক রিগ্রেশনেও ব্যবহার করি এর ফলে যেটি হবে, লজিস্টিক রিগ্রেশনে হাইপোথিসিসে সিগময়েড ফাংশন থাকার কারণে কস্ট ফাংশনের চেহারা দাঁড়াবে একটি নন-কনভেক্স ফাংশনের মতো (যে ফাংশনে অনেকগুলো লোকাল মিনিমা আছে)। অনেকগুলো লোকাল মিনিমা থাকার কারণে, আমাদের যে-কোনো একটি লোকাল মিনিমাম পয়েন্টে আটকা পড়ে যাওয়ার সম্ভাবনা বেড়ে যাবে, অথচ আমাদের লক্ষ্য হলো সব সময় কস্ট ফাংশনের সর্বনিম্ন মান অর্থাৎ গ্লোবাল মিনিমামে পৌঁছানোর। সেজন্য আমরা লজিস্টিক রিগ্রেশনে লিনিয়ার রিগ্রেশনের কস্ট ফাংশন ব্যবহার না করে নতুন একটি কস্ট ফাংশন ব্যবহার করব। ব্যাপারটি বোঝার জন্য ছবি 4.3.1-এর গ্রাফ দুটি দেখুন।



ছবি 4.3.1

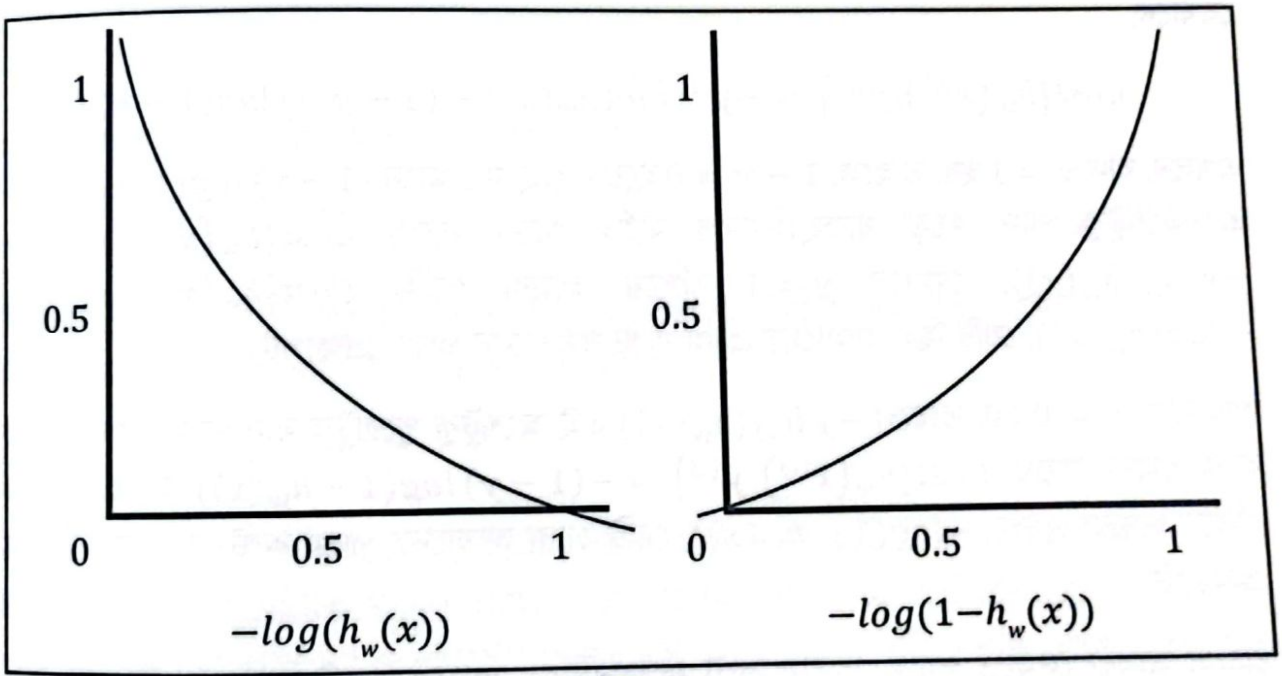
সুতরাং, লজিস্টিক রিগ্রেশনের জন্য নতুন কস্ট ফাংশন ব্যবহার করব,

$$J(w_0, w_1, w_2, \dots, w_n) = \frac{1}{m}\sum_{i=1}^{m} Cost\{h_w(x^{(i)}), y^{(i)}\}$$

যেখানে,

$$Cost\{h_w(x^{(i)}), y^{(i)}\} = \begin{cases} -log(h_w(x)), & if\ y = 1 \\ -log(1 - h_w(x)), & if\ y = 0 \end{cases}$$

প্রথমেই বলে রাখি, এই y হচ্ছে আগের মতোই, আমাদের আউটপুটের প্রকৃত মান যেটি আমরা আমাদের ট্রেনিং ডেটা থেকে পাব। এখানে দেখুন, যদি $y = 1$ হয়, তাহলে $Cost\{h_w(x^{(i)}), y^{(i)}\} = -log(h_w(x))$ হবে, আবার যদি $y = 0$ হয় তাহলে, $Cost\{h_w(x^{(i)}), y^{(i)}\} = -log(1 - h_w(x))$ হবে। ছবি 4.3.2-এর গ্রাফ থেকে ফাংশনগুলো দেখতে কী রকম তার একটি ধারণা পাওয়া যাবে।



ছবি 4.3.2

Y	$h_w(x)$	Cost
0	0	0
	$\cong 1$	$\cong \infty$
1	1	0
	$\cong 0$	$\cong \infty$

টেবিল 4.3.1

আপনারা ছবি 4.3.2-এর কস্ট ফাংশনের গ্রাফ দুটি পর্যবেক্ষণ করলে, টেবিল 4.3.1-এর অ... বুঝতে পারবেন। গ্রাফ ও চার্ট অনুযায়ী, আমাদের প্রকৃত আউটপুট যদি 1 হয় এবং আমা... হাইপোথিসিস থেকে প্রেডিকশন করেও যদি 1 পাই, তাহলে আমাদের কস্ট হবে শূন্য; যেহে... প্রকৃত মান এবং আমাদের অনুমান করা মান মিলে গেছে। যতই হাইপোথিসিসের মান শূন্যে... কাছাকাছি যেতে থাকবে, ততই কস্ট-এর মান অসীমের দিকে ধাবিত হবে। একই পদ্ধতি $y = 0$ এর ক্ষেত্রেও প্রযোজ্য।

আমরা চাইলে আমাদের কস্ট ফাংশনটিকে আরেকটু সহজ করে, দুটি শর্তই একটি সমীকরণে... মধ্যে নিয়ে এসে প্রকাশ করতে পারি। সে ক্ষেত্রে, কস্ট ফাংশনটি হবে এরকম :

$$J(w_0, w_1, w_2, \dots, w_n) = \frac{1}{m}\sum_{i=1}^{m} Cost\{h_w(x^{(i)}), y^{(i)}\}$$

যেখানে,

$$Cost\{h_w(x^{(i)}), y^{(i)}\} = -y^{(i)}\,log(h_w(x)) - (1 - y^{(i)})\,log(1 - h_w(x))$$

এখানে, যদি $y = 1$ হয়, তাহলে, $1 - y = 0$ হয়ে যাচ্ছে, যার মানে $(1 - y)\,log(1 - h_w(x))$ এ অংশটুকু বাদ পড়ে যাচ্ছে। ফলে বাকি থেকে যাচ্ছে $Cost\{h_w(x^{(i)}), y^{(i)}\} = -y\,log(h_w(x))$, যেখানে $y = 1$ বসিয়ে আমরা পাচ্ছি $Cost\{h_w(x^{(i)}), y^{(i)}\} = -\,log(h_w(x))$, যেটি ছিল আমাদের প্রথম কস্ট ফাংশনের প্রথম প্রেক্ষাপট।

আর যদি, $y = 0$ হয়, তাহলে $-y\,log(h_w(x))$ এই অংশটুকু পুরোটুকু বাদ পড়ে যাচ্ছে। ফলে বাকি থেকে যাচ্ছে $Cost\{h_w(x^{(i)}), y^{(i)}\} = -(1 - y)\,log(1 - h_w(x))$ যাতে $y = 0$ বসিয়ে আমরা পাচ্ছি $-\,log(1 - h_w(x))$, যেটি হলো আমাদের প্রথম কস্ট ফাংশনের দ্বিতীয় প্রেক্ষাপট।

তাহলে আমরা দেখতে পাচ্ছি, দ্বিতীয় কস্ট ফাংশনটিতে আসলে একটি সমীকরণ দিয়েই প্রথম কস্ট ফাংশনের দুটি প্রেক্ষাপটই আমরা পেয়ে যাচ্ছি। তাই, আমরা কস্ট ফাংশন হিসেবে দ্বিতীয়টি ব্যবহার করব।

যাহোক, আমরা আমাদের কস্ট ফাংশন পেয়ে গেলাম। আর বাকি থাকল এখন শুধু গ্রেডিয়েন্ট ডিসেন্ট অ্যালগরিদম কিংবা নরমাল ইকুয়েশন (Normal Equation) পদ্ধতিতে প্যারামিটার আপডেট করা যেটি হুবহু আগের মতোই হবে, যেভাবে আমরা মাল্টিভ্যারিয়েট লিনিয়ার রিগ্রেশনের ক্ষেত্রে পড়ে এসেছি (পরিচ্ছেদ ৩.৪)।

গ্রেডিয়েন্ট ডিসেন্ট আসলেই কীভাবে কাজ করে সেটি উদাহরণের সাহায্যে বুঝতে পরিচ্ছেদ ৩.৮ পড়ে নিতে পারেন। সেখানে লিনিয়ার রিগ্রেশনের ক্ষেত্রে কীভাবে গ্রেডিয়েন্ট ডিসেন্ট অ্যালগরিদম

কাজ করে, সেটি একেবারে হাতে কলমে দেখানো হয়েছে। এখানেও সেটি একইভাবে কাজ করবে।

পরিচ্ছেদ ৪.৪ : লজিস্টিক রিগ্রেশনে রেগুলারাইজেশনের ব্যবহার

লজিস্টিক রিগ্রেশনের প্রাথমিক বিষয়গুলো আমরা ইতিমধ্যেই পড়ে ফেলেছি। এটি খুবই জনপ্রিয় একটি ক্লাসিফিকেশন অ্যালগরিদম। লজিস্টিক রিগ্রেশনের কস্ট ফাংশন এরকম :

$$J(W) = \frac{1}{m}\sum_{i=1}^{m} Cost\{h_w(x^{(i)}), y^{(i)}\}$$

যেখানে,

$$Cost\{h_w(x^{(i)}), y^{(i)}\} = -y\, log(h_w(x)) - (1-y)\, log(1-h_w(x))$$

সুতরাং, পুরোটুকু একসঙ্গে মিলিয়ে লিখলে দাঁড়ায়,

$$J(W) = -\frac{1}{m}\left[\sum_{i=1}^{m}\{\, y^{(i)}\, log(h_w(x^{(i)})) + (1-y^{(i)})\, log(1-h_w(x^{(i)}))\}\right]$$

আমরা ইতিমধ্যেই পরিচ্ছেদ ৩.৭ থেকে জানি, L2 রেগুলারাইজেশনের জন্য আমাদেরকে এখানে অতিরিক্ত একটি টার্ম যোগ করতে হবে, সেটি হচ্ছে –

$$\frac{\lambda}{2m}\sum_{j=1}^{n} {w_j}^2$$

এখানে λ হচ্ছে আমাদের রেগুলারাইজেশন প্যারামিটার এবং আমরা রেগুলারাইজ করব w-কে অর্থাৎ আমাদের ওয়েইটগুলোকে। আমরা আরো জানি, আমরা যে পদ্ধতিতে রেগুলারাইজেশন করলাম, তার নাম হচ্ছে L2 রেগুলারাইজেশন। এটি ছাড়াও, আরেকটি পদ্ধতি আছে, যাকে বলে L1 রেগুলারাইজেশন। সেখানে আমাদেরকে যোগ করতে হতো –

$$\frac{\lambda}{2m}\sum_{j=1}^{n} |w_j|$$

আমরা আমাদের এই বইতে L2 রেগুলারাইজেশন ব্যবহার করব, কেননা ওভারফিটিং এড়ানোর জন্য L1 রেগুলারাইজেশনের চেয়ে L2 রেগুলারাইজেশন ভালো কাজ করে। এর আরো বিস্তারিত গাণিতিক হিসাবকিতাব আছে, যেগুলো সংগত কারণেই আমরা আপাতত এড়িয়ে যাব, পরবর্তী সময়ে যদি কাজ করতে করতে জানার প্রয়োজন হয় তখন জেনে নেব।

তাহলে L2 রেগুলারাইজেশন ব্যবহার করে আমাদের সর্বশেষ কস্ট ফাংশনটি দাঁড়াচ্ছে :

$$J(W) = -\frac{1}{m}\left[\sum_{i=1}^{m}\left\{\begin{array}{c} y^{(i)}\log\left(h_w\left(x^{(i)}\right)\right) \\ +\left(1-y^{(i)}\right)\log\left(1-h_w\left(x^{(i)}\right)\right)\end{array}\right\}\right] + \frac{\lambda}{2m}\sum_{j=1}^{n} w_j^2$$

দেখে হয়তো মনে হচ্ছে অনেক জটিল, কীভাবে কী করব? কীভাবে এটি যোগ করার পরে আমাদের গ্রেডিয়েন্ট ডিসেন্ট পদ্ধতি ব্যবহার করে প্যারামিটার আপডেট হবে?

আমরা গ্রেডিয়েন্ট ডিসেন্ট ব্যবহার করে কীভাবে লিনিয়ার রিগ্রেশনের প্যারামিটার বা ওয়েইটগুলো খুঁজে বের করেছিলাম, আশা করি সবার মনে আছে। তবু এখানে আমি আরেকবার লিখে দিচ্ছি :

প্রথম প্যারামিটারের জন্য :

$$w_0 = w_0 - \alpha\frac{\partial}{\partial w_0}J(W)$$

$$\frac{\partial}{\partial w_0}J(W) = \frac{\partial}{\partial w_0}\left[\frac{1}{2m}\sum_{i=1}^{m}\left\{h_w\left(x^{(i)}\right) - y^{(i)}\right\}^2\right]$$

$$= \frac{1}{m}\sum_{i=1}^{m}\left\{h_w\left(x^{(i)}\right) - y^{(i)}\right\}\cdot {x_0}^{(i)}$$

পরবর্তী সব প্যারামিটার ($j = 1, 2, 3, \ldots, n$) জন্য –

$$w_j = w_j - \alpha\frac{\partial}{\partial w_j}J(W)$$

$$\frac{\partial}{\partial w_j}J(W) = \frac{\partial}{\partial w_j}\left[\frac{1}{2m}\sum_{i=1}^{m}\left\{h_w\left(x^{(i)}\right) - y^{(i)}\right\}^2\right]$$

$$= \frac{1}{m}\sum_{i=1}^{m}\left\{h_w\left(x^{(i)}\right) - y^{(i)}\right\}\cdot {x_j}^{(i)}$$

এখন, এটি ছিল সাধারণ গ্রেডিয়েন্ট ডিসেন্ট, কোনোরকম রেগুলারাইজেশন ছাড়াই। এখন, আমরা যদি রেগুলারাইজড কস্ট ফাংশন ব্যবহার করি, যেটি কিনা এরকম :

$$J(W) = -\frac{1}{m}\left[\sum_{i=1}^{m}\left\{\begin{array}{c} y^{(i)}\log\left(h_w\left(x^{(i)}\right)\right) \\ +\left(1-y^{(i)}\right)\log\left(1-h_w\left(x^{(i)}\right)\right)\end{array}\right\}\right] + \frac{\lambda}{2m}\sum_{j=1}^{n} w_j^2$$

সে ক্ষেত্রে, আমাদের প্যারামিটার আপডেটের ক্ষেত্রে সামান্য কিছু পরিবর্তন আসবে। প্রথম প্যারামিটার w_0-এর জন্য আগে যা ছিল তা-ই থাকবে –

$$w_0 = w_0 - \alpha\frac{\partial}{\partial w_0}J(W)$$

$$\frac{\partial}{\partial w_0}J(W) = \frac{1}{m}\sum_{i=1}^{m}\left\{h_w\left(x^{(i)}\right) - y^{(i)}\right\}\cdot {x_0}^{(i)}$$

পরবর্তী প্যারামিটার ($j = 1, 2, 3, \dots, n$)-গুলোর জন্য পরিবর্তিত হয়ে হবে –

$$w_j = w_j - \alpha \frac{\partial}{\partial w_j} J(W)$$

$$\frac{\partial}{\partial w_j} J(W) = \frac{1}{m} \sum_{i=1}^{m} \{h_w(x^{(i)}) - y^{(i)}\} \cdot x_j^{(i)} - \frac{\lambda}{m} w_j$$

ব্যস, শুধু এটুকুই পরিবর্তন। বাকি সবকিছু আমরা যে লজিস্টিক রিগ্রেশন পড়েছি, সে পদ্ধতিতেই করতে হবে।