**به نام خدا**

**توضیح کد :**

*# افزودن ضریب اریب (bias)*

X\_b = np.c\_[np.ones((len(X), 1)), X] *# تبدیل X به شکل [1, X]*

مفهوم **ضریب اریب (Bias Term)** و دلیل اضافه کردن ستون ۱ به داده‌ها در یادگیری ماشین را بررسی می‌کنیم.

## **۱. ضریب اریب (Bias) چیست؟**

در مدل‌های خطی (مثل رگرسیون)، رابطه بین ورودی (X) و خروجی (y) به صورت زیر تعریف می‌شود:

y=θ0+θ1x1+θ2x2+...+θnxn*y*=*θ*0​+*θ*1​*x*1​+*θ*2​*x*2​+...+*θn*​*xn*​

* θ₀**(Bias Term)**: مقدار پایه‌ای که وقتی تمام xها صفر هستند، y = θ₀ خواهد بود.  
  (مثال: اگر مدل پیش‌بینی قیمت خانه بر اساس متراژ باشد، θ₀ هزینه‌های ثابت مثل زمین است، حتی اگر متراژ صفر باشد!)
* θ₁**تا**θₙ: ضرایب مربوط به هر ویژگی.
* برای ساده‌تر کردن محاسبات ماتریسی، تمام پارامترها (شامل θ₀) را در یک بردار θ قرار می‌دهیم. اما برای این کار، باید یک ستون **۱** به ماتریس X اضافه کنیم تا θ₀ در ضرب ماتریسی دخیل شود.

### مثال عددی:

* فرض کنید داده‌های ما (X) و خروجی (y) به صورت زیر هستند:

X = [[2], y = [4,

[3], 6,

[4]] 8]

مدل خطی ما:

*y*=*θ*0​+*θ*1​*x*

پس از اضافه کردن ستون ۱:

X\_b = [[1, 2],

[1, 3],

[1, 4]]

حالا می‌توانیم مدل را به صورت ماتریسی بنویسیم:

*y*=*Xb*​⋅*θ*

که در آن:

*θ*=[*θ*0​

*θ*1​​]

## **۳. محاسبات با مثال واقعی**

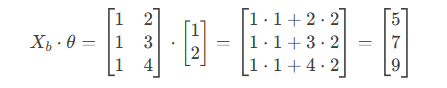
فرض کنید مدل نهایی ما پارامترهای زیر را یاد گرفته است:

*θ*0​=1,*θ*1​=2

پیش‌بینی مدل برای x = 3:

*y*=*θ*0​+*θ*1​⋅*x*=1+2⋅3=7

به صورت ماتریسی:



## **4 . چرا این کار در هوش مصنوعی مهم است؟**

* **انعطاف مدل**: بدون θ₀، مدل نمی‌تواند داده‌هایی که رابطه‌شان از مبدأ عبور نمی‌کند را توصیف کند.
* **یکپارچه‌سازی در محاسبات**: با اضافه کردن ستون ۱، تمام پارامترها (θ₀ تا θₙ) را می‌توان با روش‌های بهینه‌سازی (مثل Gradient Descent) به صورت همزمان آموزش داد

## **جمع‌بندی نهایی**

* **ستون ۱** در X\_b برای گنجاندن **عرض از مبدأ (Bias)** در مدل است.
* بدون آن، مدل در توصیف رواقعی که از مبدأ نمی‌گذرند **ناتوان** خواهد بود.
* این تکنیک در **تمام مدل‌های خطی** (رگرسیون، شبکه‌های عصبی، ...) استفاده می‌شود.

**توضیح کد :**

*# محاسبه پارامترهای مدل (θ) با معادله نرمال*

theta = np.linalg.inv(X\_b.T.dot(X\_b)).dot(X\_b.T).dot(y)

حتماً! این خط کد، یک روش ریاضی برای پیدا کردن بهترین پارامترهای مدل رگرسیون خطی (θ) است که **معادله نرمال (Normal Equation)** نام دارد. بیایید آن را قدم به قدم و با مثال ساده توضیح دهیم:

### ۱. ****هدف چیست؟****

ما می‌خواهیم ضرایب مدل (θ) را طوری پیدا کنیم که خط پیش‌بینی مدل، کمترین فاصله (خطا) را با داده‌های واقعی داشته باشد.  
مدل رگرسیون خطی:

*y*=*Xb*​⋅*θ*

که در آن:

* X\_b: ماتریس ویژگی‌ها + ستون ۱ (برای ضریب اریب).
* θ: پارامترهای مدل (شامل θ₀ و θ₁).
* y: مقادیر واقعی خروجی.

### ۲. ****معادله نرمال چیست؟****

این معادله، یک فرمول ریاضی برای محاسبه مستقیم  **θ** است **بدان نیاز به روش‌های تکراری مثل Gradient Descent نیست**:

* X\_b^T: ترانهاده (Transpose) ماتریس X\_b.
* (X\_b^T X\_b)^{-1}: معکوس ماتریس حاصل از ضرب.

### ۳. ****تجزیه کد به قدم‌های کوچک****

theta = np.linalg.inv(X\_b.T.dot(X\_b)).dot(X\_b.T).dot(y)

#### **قدم‌ها:**

۱. **محاسبه**X\_b.T**(ترانهاده**X\_b**)**:

اگر X\_b این باشد:

[[1, 2],

[1, 3],

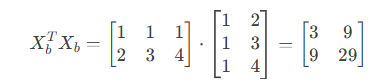
[1, 4]]

ترانهاده آن (X\_b.T):

[[1, 1, 1],

[2, 3, 4]]

۲. **ضرب**X\_b.T**در**X\_b:



* عدد ۳: جمع ستون اول X\_b (۱+۱+۱).
* عدد ۹: جمع ضرب ستون اول و دوم X\_b (۱×۲ + ۱×۳ + ۱×۴).

۳. **معکوس ماتریس حاصل (**inv**)**:

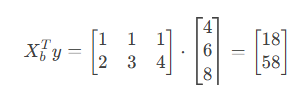
* معکوس [[3, 9], [9, 29]] می‌شود:

[[ 4.83, -1.5 ],

[-1.5 , 0.5 ]]

۴. **ضرب در**X\_b.T**و سپس**y:

* ابتدا X\_b^T y را محاسبه می‌کنیم:



* عدد ۱۸: جمع yها (۴+۶+۸).
* عدد ۵۸: (۲×۴ + ۳×۶ + ۴×۸).

سپس نتیجه را در معکوس ماتریس ضرب می‌کنیم:



* این یعنی0  θ₀ = و 2 θ₁ = .

### ۴. ****چرا این معادله کار می‌کند؟****

* این معادله، **کمینه‌سازی خطای مربعات (MSE)** را انجام می‌دهد.
* از مشتق‌گیری برای یافتن نقطه کمینه استفاده می‌کند (به صورت تحلیلی).

### ****جمع‌بندی****

* X\_b.T.dot(X\_b): ماتریس کوواریانس ویژگی‌ها.
* inv(...): معکوس این ماتریس برای حل معادله.
* X\_b.T.dot(y): همبستگی بین ویژگی‌ها و خروجی.
* **نتیجه نهایی (**θ**)**: ضرایب بهینه مدل.

این روش برای داده‌های **کوچک** عالی است، اما برای داده‌های بزرگ از **Gradient Descent** استفاده می‌شود (چون معکوس ماتریس هزینه محاسباتی بالایی دارد).