04-10-2025

Agenda:

practical linear Regression → gradient descent → Batch gradient descent

→ SGD → one data point at a time

Train

1. train.csv features + Target column → Train, test
2. test.csv ~~features + Target column~~

→ prediction → wrong

y $\hat{y}$

→ data → EDA → FE

1 stage → ML → 1.0

Highe

→ Endocing → OHE TE

→ outliers → ML → good, Bad → remove outliers

whole

data → M → 90 → M → 70, 30

M → 10

method — M

↓

2 part

split with Train_test_split

Train

x1

↓

$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$ Blue, green, Red → OHE

→ 0 0 1 → Blue
0 1 0 → green
1 0 0 → Red

OHE

↓

Train

Test

x1

Blue
green
Red
violet → error → ignore → [0,0,0]
yellow → [0,0,0]

weather → FE → google → Real

App!!

32°C 33° → FE → scaler → MC

A →

B

B A → MC → prediction

A

Train

unknown →

violet → [0 0 0]
yellow →

MC

unknown
Test

fit_transform
transform

fit tries StandardScaler(data)

↓

α = update α ← train data

fit → learn
transform → apply

test
↓
train → ML → predict (transform)
↓ ↓
fit apply

fit
data → scaling → transform ← train → scaled_train data
(train)
↓
parameters

.fit()
.transform()
.fit_transform() → .fit() + .transform()

fit & transform

Min Max Scaling → $\alpha = 200$ → ML

train → 200 → 400

leaking your question of final exam
↓
True exam

α → test → testing
→ production
→ application

100, 410

error → gradient → β
update

1 epoch
1000, 2000

error ≈ 0

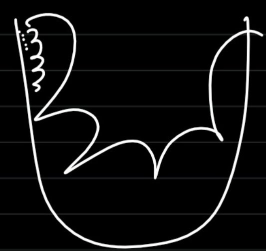
random_state ↔
seed ↔ numpy

python → random value → [1, 2]

seed = 2 → → [100, 200] → 60%. (A)

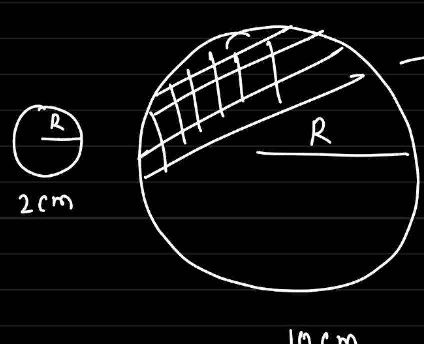
seed = 1 → → [34, 96] → 98%. (B)

random_state = 42



$$\beta^{new} = \beta^{old} - \alpha \text{ gradient}$$

α ↓ learning rate



→ cookie → children

β^{old} 0.1 $\qquad$ -300

$$\beta^{new} = \beta_{old} - \alpha \; gradi$$
$$= 0.1 - ((-300))$$
$$\alpha = 0.01$$

$\alpha = 0.00001$

Data

train test split

Scaling →

Encoding →

Model training (SGD) → save locally file

Model prediction (SGD trained model) → answers

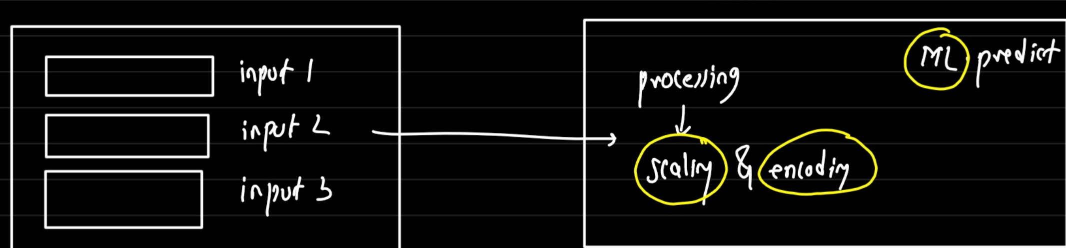
↑

test, processed
(scale & encode)

Height, weight, x, y $\xrightarrow{.predict}$ prediction

Human readable

3 components



→ DS → ML model

→ DS 2 | AI Engineer → ML model → production

Linear Regression

Gradient Descent
↓
parameter update in iteration
↓
medium-large dataset

→ low size dataset
→ computation (disadvantage)
↓
sklearn . SGD Regressor

OLS
(Ordinary least squares)
↓
good for small-medium size dataset
↓
→ Finds the optimal parameter in a single pass

→ calculus

OLS → Ordinary Least Squares

GD : $J(\beta) = \frac{1}{2m} \sum_{i=1}^{m} (y_i - \hat{y}_i)^2$ → Modified MSE

OLS : $J(\beta) = \sum_{i=1}^{m} (\underline{y_i - \hat{y}_i})^2$ → Sum of squared Errors we want to minimize:

$$J(\beta) = (y - X\beta)^2$$

$$= (y - X\beta)^T \cdot (y - X\beta)$$

$$= y^T y - 2\beta^T X^T y + \beta^T X^T X \beta$$

Derivative w.r.t β

$$\frac{\partial}{\partial \beta}(y^T y) = 0 \qquad \frac{\partial}{\partial \beta}(-2\beta^T X^T y) = \underline{-2X^T y}$$

$$\frac{\partial}{\partial \beta}(\beta^T X^T X \beta) = 2X^T X \beta$$

$$\frac{\partial J}{\partial \beta} = -2X^T y + 2X^T X \beta$$

set derivative $= 0$

$$-2X^T y + 2X^T X \beta = 0$$

$= 0$

<u>solve for β:</u>

$\beta =$ np.random ( )
np.zeros ( )

$$\cancel{2}X^T X \beta = \cancel{2}X^T y$$

$$\beta = \frac{X^T y}{X^T X} = X^T y (X^T X)^{-1}$$

this is OLS closed-form solution

Gradient descent:

$$\beta^{new} = \beta^{old} - \alpha \cdot \frac{1}{m} X^T (X\beta - y)$$

multiple times

iteration

OLS:

$$\beta^{new} = (X^T X)^{-1} X^T y$$