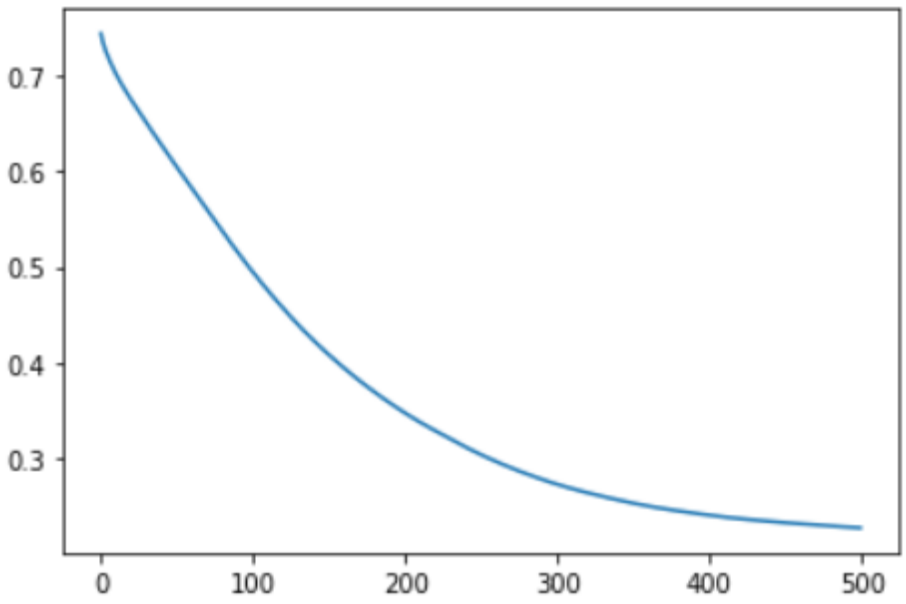
Gradient Descent in Neural Networks

Batch Gradient Descent

- **Batch GD** uses the entire data to calculate slope in this equation.
  - **Fast**

```
model.fit(X_scaled,y,epochs=500,batch_size=n,validation_split=0.2)
```

- For batch Batch GD, `batch_size = n` → **No. of rows**
  - Here, you're creating a single batch of n rows
- Loss decreases **smoothly** in Batch GD



Stochastic Gradient Descent

- **Stochastic Gradient Descent** takes 1 row & calculates values for each row

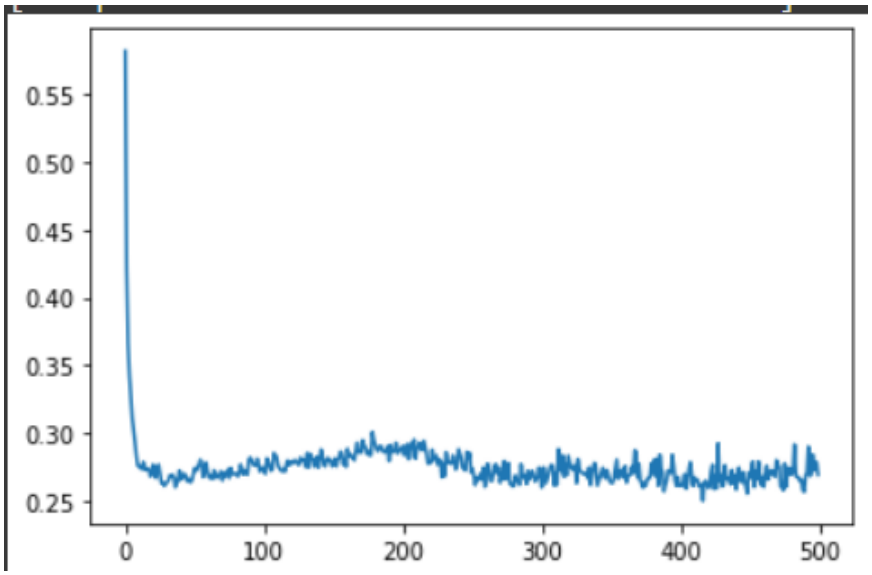
- **Slow**

```
model.fit(X_scaled,y,epochs=500,batch_size=1,validation_split=0.2)
```

- For batch SGD, `batch_size = 1`

> 💡 **Stochastic GD** gives **better accuracy** with **same no. of epochs.**

- Loss decreases jaggedly in SGD



- 👆 This jaggedness helps SGD to move out of local minima.
- But the solution is also approximate.

Mini-Batch Gradient Descent

- This balances the Batch and Stochastic GD
- You define a batch size - eg. 30

- It gets updated when it goes through 30 rows
- If there are 300 rows, **Mini batch Descent** it will update **10 times**
- **Stochastics GD** would update → **300 times**
- **Batch GD** would update → **1 Time**

💡 **MOSTLY USED GD→ Mini-Batch GD**

Interview Question

Q. Why batch-size is provided in multiples of 2?

- RAM is designed to handle binary values.
- It's done for effective use of RAM.
- Optimization technique