Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

iHUB DivyaSampark, IIT Roorkee
and
Ritvij Bharat Private Limited

Led by : Shreyas Shukla

Algorithm Theory
Cost Function

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)
iHUB DivyaSampark, IIT Roorkee
and
ed
ukla

Recall we are searching for Beta values for a best-fit line.

$$\hat{y} = \sum_{i=0}^{n} \beta_i x_i$$

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)
iHUB DivyaSampark, IIT Roorkee
and

How to choose these beta coefficients?

$$\hat{y} = \sum_{i=0}^{n} \beta_i x_i$$

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)
iHUB DivyaSampark, IIT Roorkee
and
ed
ukla

We've decided to define a "best-fit" as **minimizing the squared error.**

$$\hat{y} = \sum_{i=0}^{n} \beta_i x_i$$

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

The residual error for some row ***j*** is:

iHUB DivyaSampark, IIT Roorkee
and
Ritvij Bharat Private Limited

$$y^j - \hat{y}^j$$

Led by : Shreyas Shukla

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)
iHUB DivyaSampark, IIT Roorkee
and
Ritvij Bharat Pr
Led by : Shreyas Shukla

Squared Error for some row ***j*** is then:

$$\left(y^j - \hat{y}^j\right)^2$$

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)
iHUB DivyaSampark, IIT Roorkee
Ritvij Bharat
Led by : Shreyas Shukla

Sum of squared errors for **m** rows is then:

$$\sum_{j=1}^{m} \left(y^j - \hat{y}^j\right)^2$$

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

Average squared error for **m** rows:

$$\frac{1}{2m}\sum_{j=1}^{m}\left(y^{j}-\hat{y}^{j}\right)^{2}$$

Led by : Shreyas Shukla

fining a cost function **J.**

$$J(\boldsymbol{\beta}) = \frac{1}{2m}\sum_{j=1}^{m}\left(y^j - \hat{y}^j\right)^2$$

$$= \frac{1}{2m}\sum_{j=1}^{m}\left(y^j - \sum_{i=0}^{n}\beta_i x_i^j\right)^2$$

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

A **cost function** is defined by some measure of error.

$$J(\boldsymbol{\beta})$$

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

- This means we wish to **minimize** the cost function.

$$J(\boldsymbol{\beta})$$

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

Our cost function can be defined by the squared error:

$$J(\boldsymbol{\beta}) = \frac{1}{2m} \sum_{j=1}^{m} \left(y^j - \hat{y}^j \right)^2$$

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

Error between real **y** and predicted **ŷ**

$$J(\boldsymbol{\beta}) = \frac{1}{2m}\sum_{j=1}^{m}\left(\boxed{y^j - \hat{y}^j}\right)^2$$

Squaring corrects for negative and positive errors.

$$J(\boldsymbol{\beta}) = \frac{1}{2m} \boxed{\sum_{j=1}^{m}} \left(y^j - \hat{y}^j\right)^{\boxed{2}}$$

Summing error for m rows.

$$J(\boldsymbol{\beta}) = \frac{1}{2m} \sum_{j=1}^{\boxed{m}} \left(y^j - \hat{y}^j \right)^2$$

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

Summing error for m rows.

$$J(\boldsymbol{\beta}) = \frac{1}{2m} \boxed{\sum_{j=1}^{m}} \left(y^j - \hat{y}^j\right)^2$$

Divide by **m** to get **mean**

$$J(\boldsymbol{\beta}) = \boxed{\frac{1}{2m}} \sum_{j=1}^{m} \left(y^j - \hat{y}^j \right)^2$$

Additional ½ is for convenience for derivative.

$$J(\boldsymbol{\beta}) = \frac{1}{\boxed{2}m} \sum_{j=1}^{m} \left(y^j - \hat{y}^j \right)^{\boxed{2}}$$

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

$$\begin{aligned} J(\boldsymbol{\beta}) &= \frac{1}{2m}\sum_{j=1}^{m}\left(y^j - \hat{y}^j\right)^2 \\ &= \frac{1}{2m}\sum_{j=1}^{m}\left(y^j - \sum_{i=0}^{n}\beta_i x_i^j\right)^2 \end{aligned}$$

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)
iHUB DivyaSampark, IIT Roorkee

From calculus we know that to minimize a function we can take its derivative and set it equal to zero.

$$\begin{aligned}\frac{\partial J}{\partial \beta_k}(\boldsymbol{\beta}) &= \frac{\partial}{\partial \beta_k}\left(\frac{1}{2m}\sum_{j=1}^{m}\left(y^j - \sum_{i=0}^{n}\beta_i x_i^j\right)^2\right) \\ &= \frac{1}{m}\sum_{j=1}^{m}\left(y^j - \sum_{i=0}^{n}\beta_i x_i^j\right)(-x_k^j)\end{aligned}$$

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

iHUB DivyaSampark, IIT Roorkee

$$\boxed{\frac{\partial J}{\partial \beta_k}}(\boldsymbol{\beta}) = \boxed{\frac{\partial}{\partial \beta_k}}\left(\frac{1}{2m}\sum_{j=1}^{m}\left(y^j - \sum_{i=0}^{n}\beta_i x_i^j\right)^2\right)$$

$$= \frac{1}{m}\sum_{j=1}^{m}\left(y^j - \sum_{i=0}^{n}\beta_i x_i^j\right)(-x_k^j)$$

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)
iHUB DivyaSampark, IIT Roorkee

Recall from calculus to minimize a function we can take its derivative and set it equal to zero.

$$\frac{\partial J}{\partial \beta_k}(\boldsymbol{\beta}) = \frac{\partial}{\partial \beta_k}\left(\frac{1}{\boxed{2}m}\sum_{j=1}^{m}\left(y^j - \sum_{i=0}^{n}\beta_i x_i^j\right)^{\boxed{2}}\right)$$

$$= \frac{1}{m}\sum_{j=1}^{m}\left(y^j - \sum_{i=0}^{n}\beta_i x_i^j\right)(-x_k^j)$$

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)
iHUB DivyaSampark, IIT Roorkee
and
Ritvij Bharat Private Limited
Led by : Shreyas Shukla

- Unfortunately, it is not scalable to try to get an analytical solution to minimize this cost function.
- So we use **gradient descent** to minimize this **cost function.**

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

iHUB DivyaSampark, IIT Roorkee
and
Ritvij Bharat Private Limited

Algorithm Theory
Gradient Descent

Led by : Shreyas Shukla

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)
iHUB DivyaSampark, IIT Roorkee
and
Ritvij Bharat Private Limited
Led by : Shreyas Shukla

we can describe this cost function through vectorized matrix notation and use **gradient descent** to have a computer figure out the set of Beta coefficient values that minimize the **cost/loss** function.

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)
iHUB DivyaSampark, IIT Roorkee
and
d by : Shreyas Shukla

Derivative of the cost function:

$$\frac{\partial J}{\partial \beta_k}(\boldsymbol{\beta}) = \frac{1}{m}\sum_{j=1}^{m}\left(y^j - \sum_{i=0}^{n}\beta_i x_i^j\right)(-x_k^j)$$

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)
iHUB DivyaSampark, IIT Roorkee
and
Private Limited
reyas Shukla

We can use a **gradient** to express the derivative of the cost function with respect to each β

$$\nabla_{\boldsymbol{\beta}} J = \begin{bmatrix} \frac{\partial J}{\partial \beta_0} \\ \vdots \\ \frac{\partial J}{\partial \beta_n} \end{bmatrix}$$

Mastering Machine Learning with Python

(27th Aug 2024 - 18th Oct 2024)

iHUB DivyaSampark, IIT Roorkee

and

$$\nabla_{\boldsymbol{\beta}} J = \begin{bmatrix} -\frac{1}{m}\sum_{j=1}^{m}\left(y^j - \sum_{i=0}^{n}\beta_i x_i^j\right)x_0^j \\ \vdots \\ -\frac{1}{m}\sum_{j=1}^{m}\left(y^j - \sum_{i=0}^{n}\beta_i x_i^j\right)x_n^j \end{bmatrix}$$

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)
iHUB DivyaSampark, IIT Roorkee
Led by : Shreyas Shukla

We know we can vectorize our data:

$$\mathbf{X} = \begin{bmatrix} 1 & x_1^1 & x_2^1 & \dots & x_n^1 \\ 1 & x_1^2 & x_2^2 & \dots & x_n^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^m & x_2^m & \dots & x_n^m \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \qquad \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_n \end{bmatrix}$$

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

iHUB DivyaSampark, IIT Roorkee
and

$$\nabla_{\beta} J = \begin{bmatrix} -\frac{1}{m}\sum_{j=1}^{m}\left(y^j - \sum_{i=0}^{n}\beta_i x_i^j\right)x_0^j \\ \vdots \\ -\frac{1}{m}\sum_{j=1}^{m}\left(y^j - \sum_{i=0}^{n}\beta_i x_i^j\right)x_n^j \end{bmatrix}$$

Mastering Machine Learning with Python

(27th Aug 2024 - 18th Oct 2024)

iHUB DivyaSampark, IIT Roorkee

and

$$\nabla_\beta J = \begin{bmatrix} -\boxed{\frac{1}{m}\sum_{j=1}^{m}}\left(y^j - \right.\boxed{\left.\sum_{i=0}^{n}\beta_i x_i^j\right) x_0^j} \\ \vdots \\ -\frac{1}{m}\sum_{j=1}^{m}\left(y^j - \sum_{i=0}^{n}\beta_i x_i^j\right) x_n^j \end{bmatrix}$$

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

iHUB DivyaSampark, IIT Roorkee

Led by : Shreyas Shukla

$$\nabla_{\beta} J = -\frac{1}{m}\begin{bmatrix} \sum_{j=1}^{m} y^j x_0^j \\ \vdots \\ \sum_{j=1}^{m} y^j x_n^j \end{bmatrix} + \frac{1}{m}\begin{bmatrix} \sum_{j=1}^{m} \sum_{i=0}^{n} \beta_i x_i^j x_0^j \\ \vdots \\ \sum_{j=1}^{m} \sum_{i=0}^{n} \beta_i x_i^j x_n^j \end{bmatrix}$$

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

We can now calculate the gradient for any set of Beta values!

iHUB DivyaSampark, IIT Roorkee

$$\nabla_{\boldsymbol{\beta}} J = -\frac{1}{m}\begin{bmatrix} \sum_{j=1}^{m} y^j x_0^j \\ \vdots \\ \sum_{j=1}^{m} y^j x_n^j \end{bmatrix} + \frac{1}{m}\begin{bmatrix} \sum_{j=1}^{m}\sum_{i=0}^{n} \beta_i x_i^j x_0^j \\ \vdots \\ \sum_{j=1}^{m}\sum_{i=0}^{n} \beta_i x_i^j x_n^j \end{bmatrix}$$

Led by : Shreyas Shukla

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)
iHUB DivyaSampark, IIT Roorkee
Led by : Shreyas Shukla

What is the best way to "guess"?

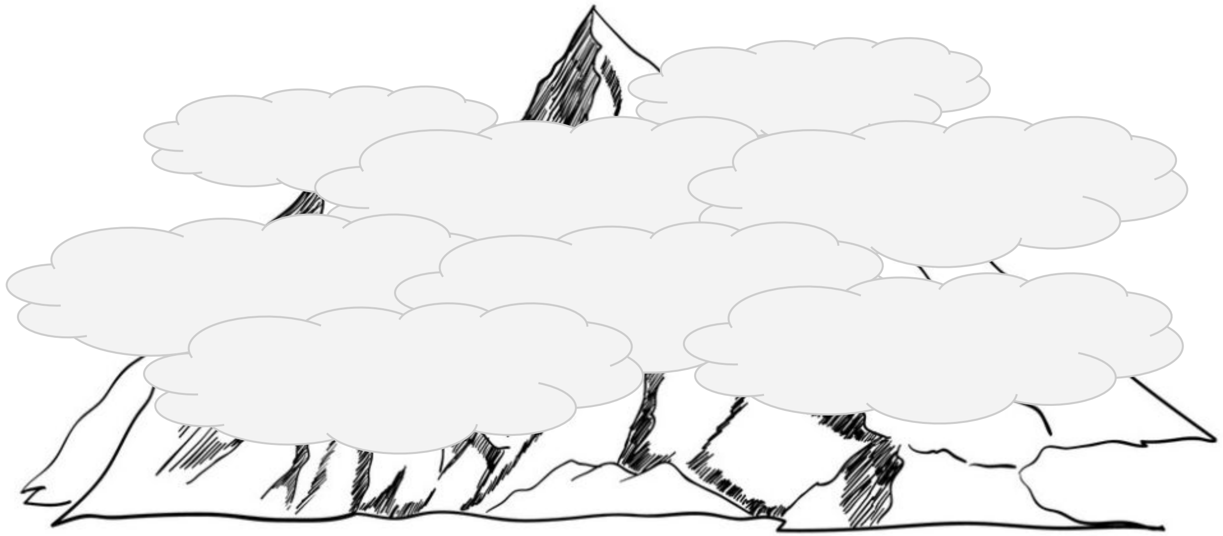
$$\nabla_{\boldsymbol{\beta}} J = -\frac{1}{m}\begin{bmatrix} \sum_{j=1}^{m} y^j x_0^j \\ \vdots \\ \sum_{j=1}^{m} y^j x_n^j \end{bmatrix} + \frac{1}{m}\begin{bmatrix} \sum_{j=1}^{m}\sum_{i=0}^{n} \beta_i x_i^j x_0^j \\ \vdots \\ \sum_{j=1}^{m}\sum_{i=0}^{n} \beta_i x_i^j x_n^j \end{bmatrix}$$

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

Common mountain analogy

source:pierian_data

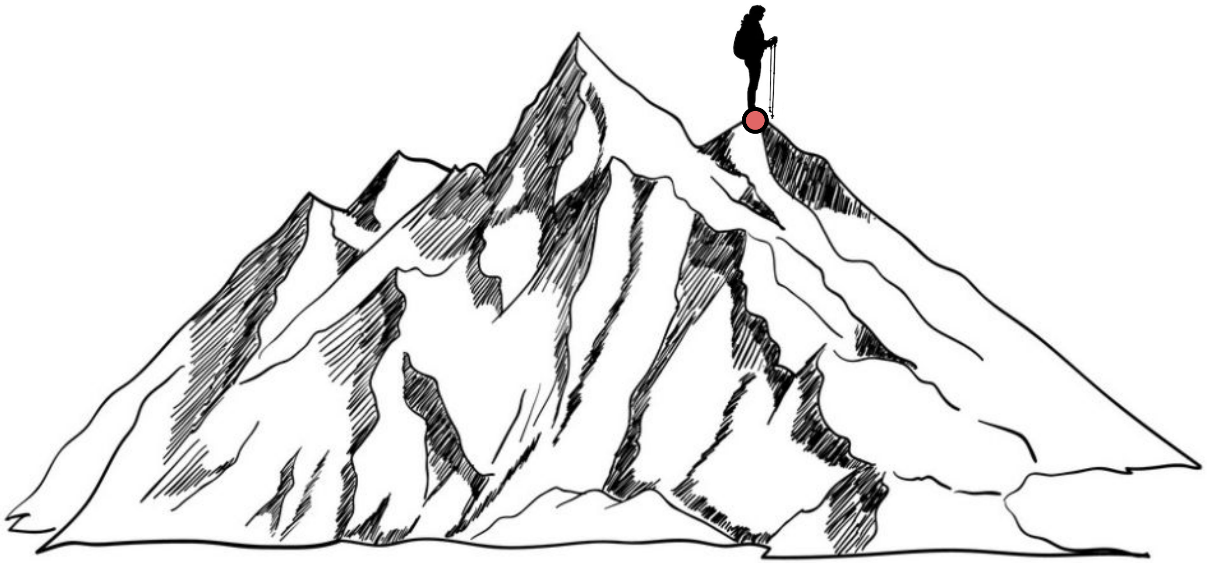
Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)



source:pierian_data

Mastering Machine Learning with Python

(27th Aug 2024 - 18th Oct 2024)

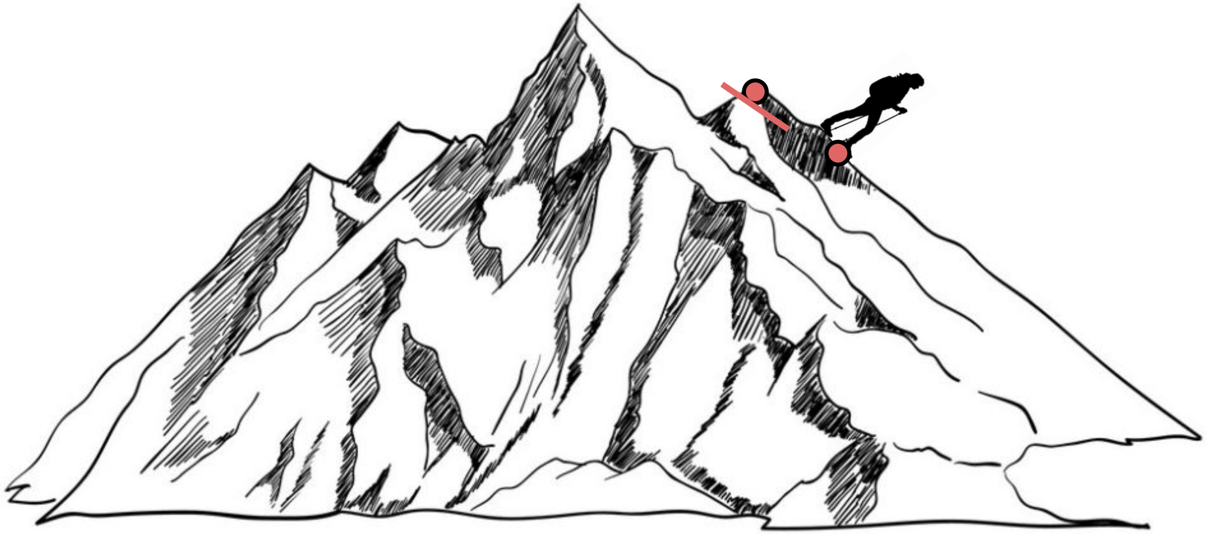


source:pierian_data

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

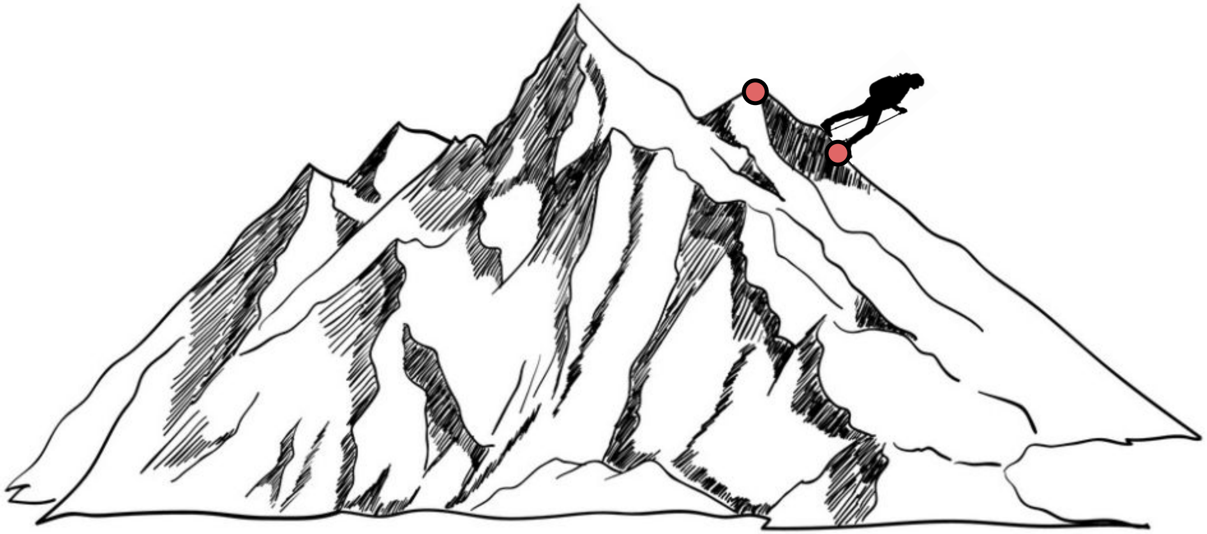
source:pierian_data

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)



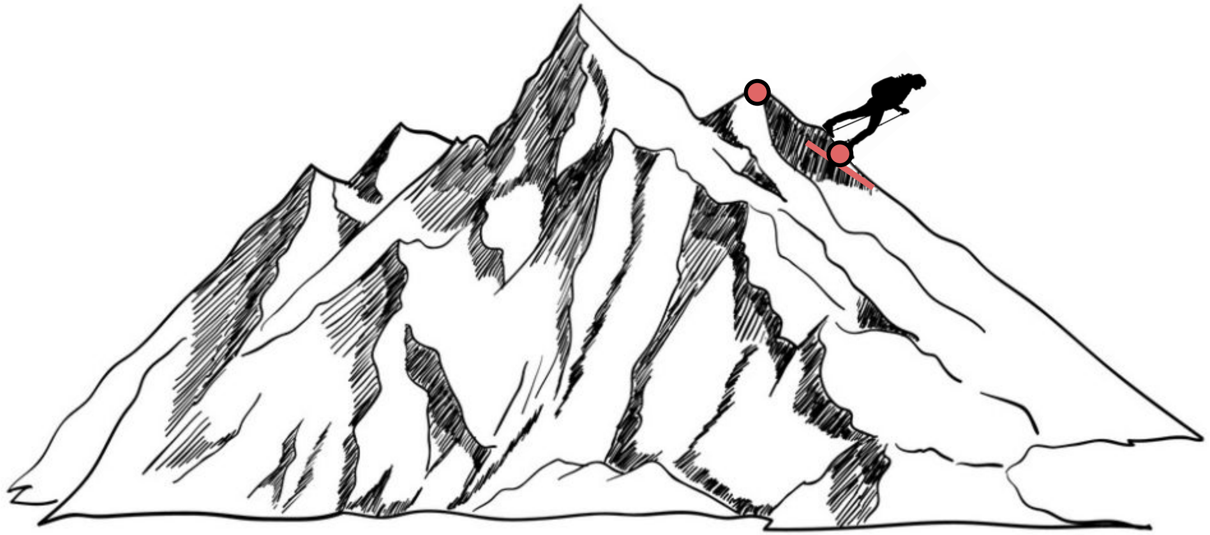
source:pierian_data

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)



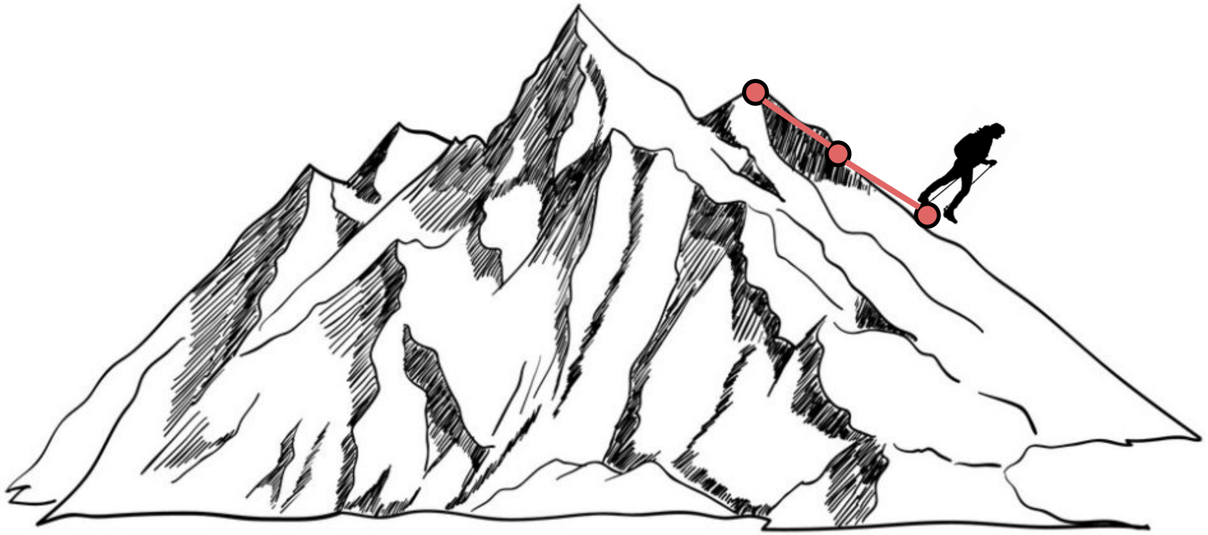
source:pierian_data

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)



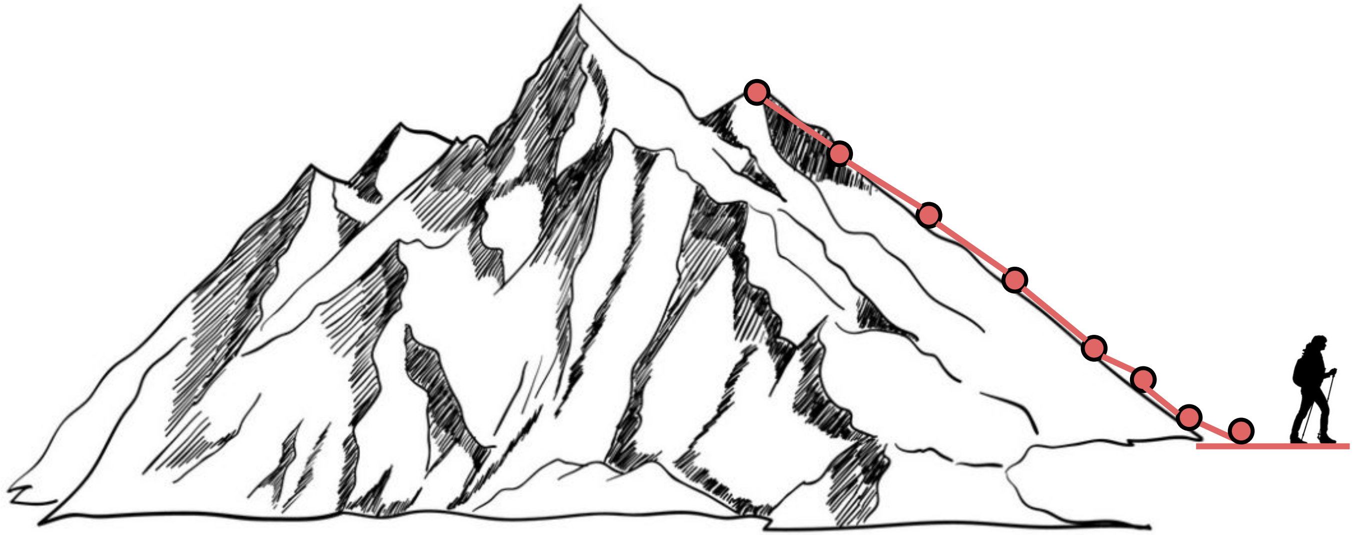
source:pierian_data

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)



source:pierian_data

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

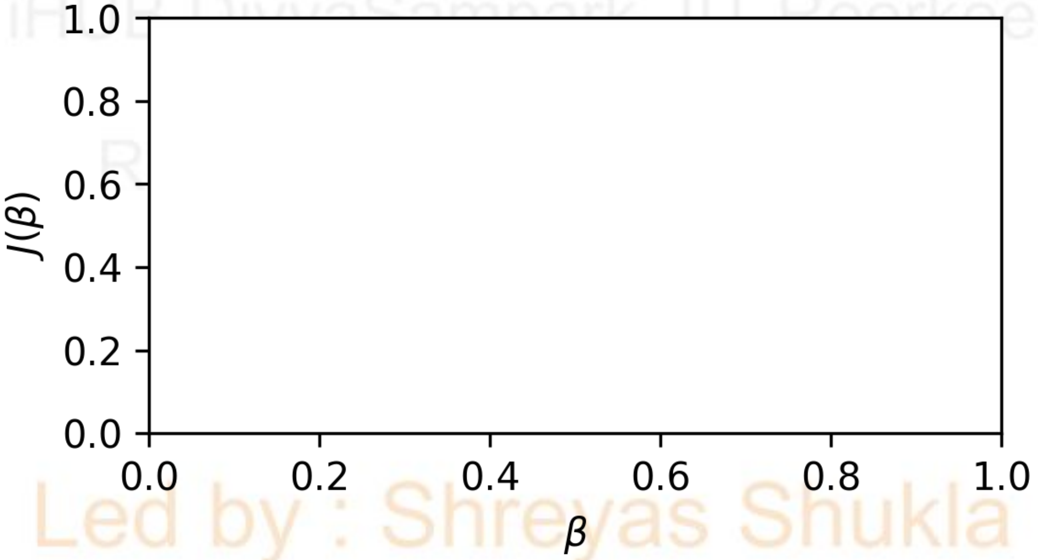


source:pierian_data

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

1 dimensional cost function (single Beta)

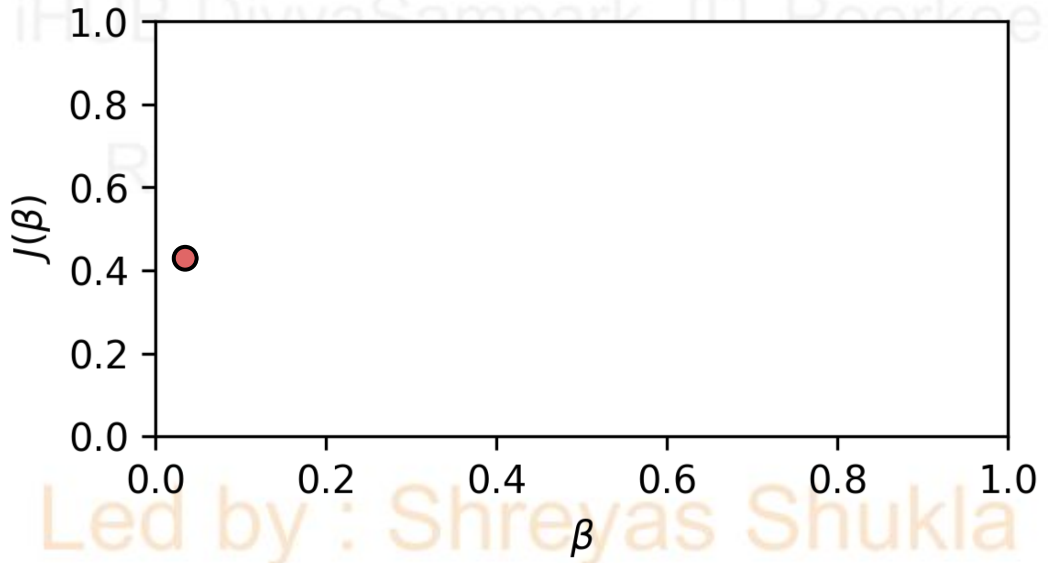
iHUB DivyaSampark, IIT Roorkee
R



Led by : Shreyas Shukla

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

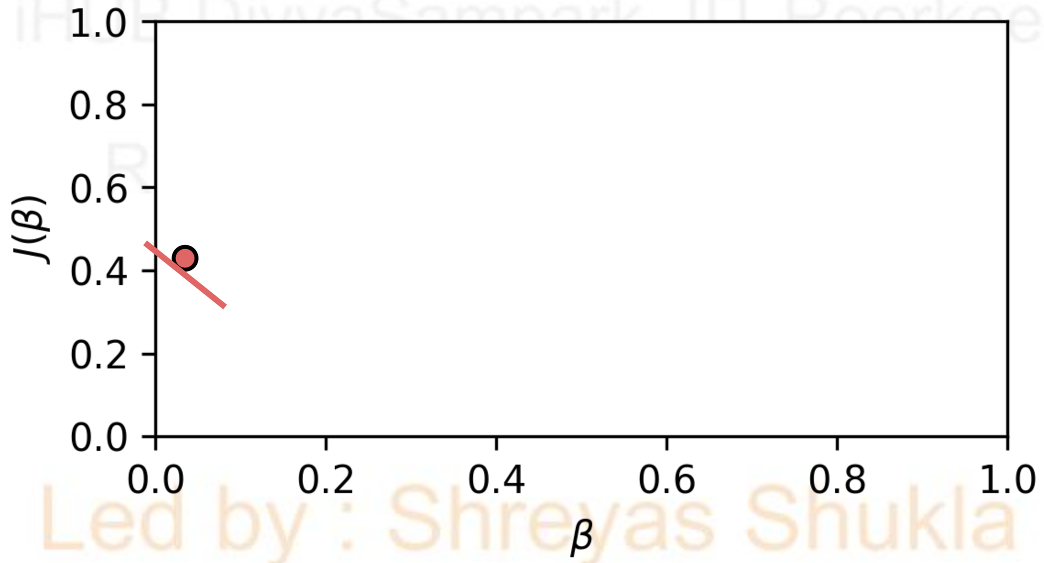
Choose a starting point



iHUB DivyaSampark, IIT Roorkee
Led by : Shreyas Shukla

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

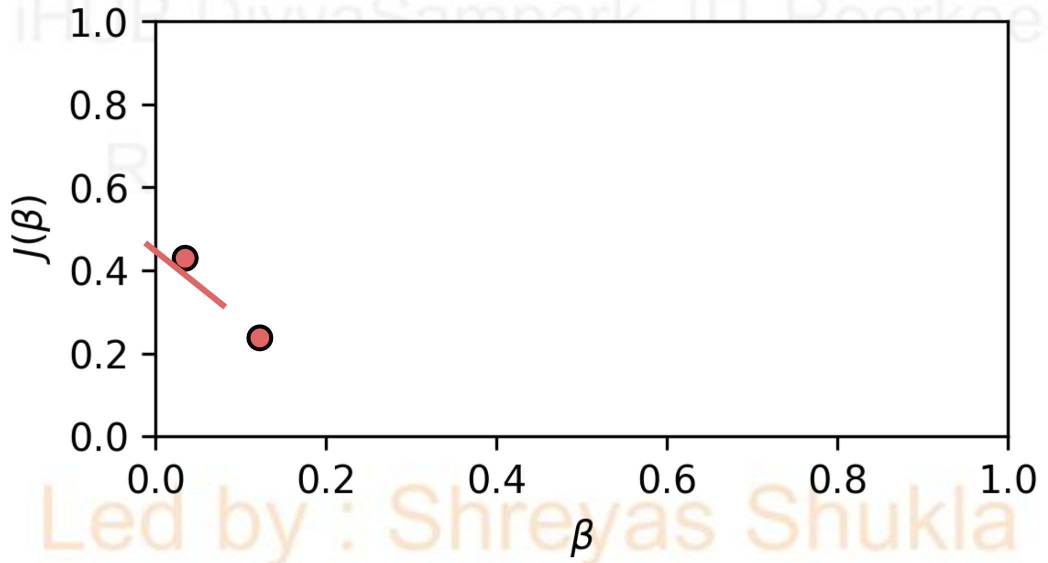
Calculate gradient at that point



iHUB DivyaSampark, IIT Roorkee
Led by : Shreyas Shukla

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

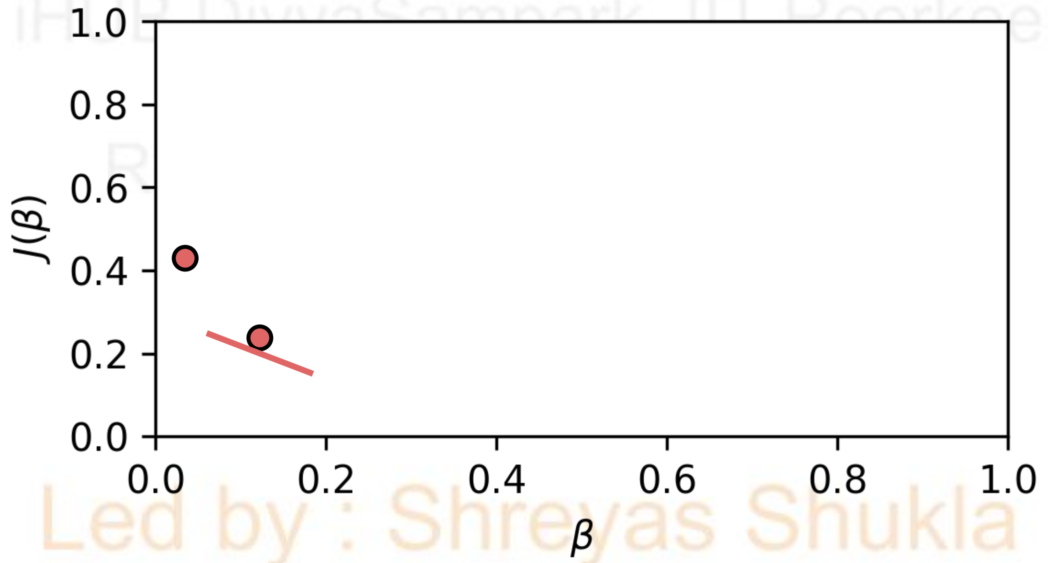
Step forward proportional to **negative** gradient



iHUB DivyaSampark, IIT Roorkee
Led by : Shreyas Shukla

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

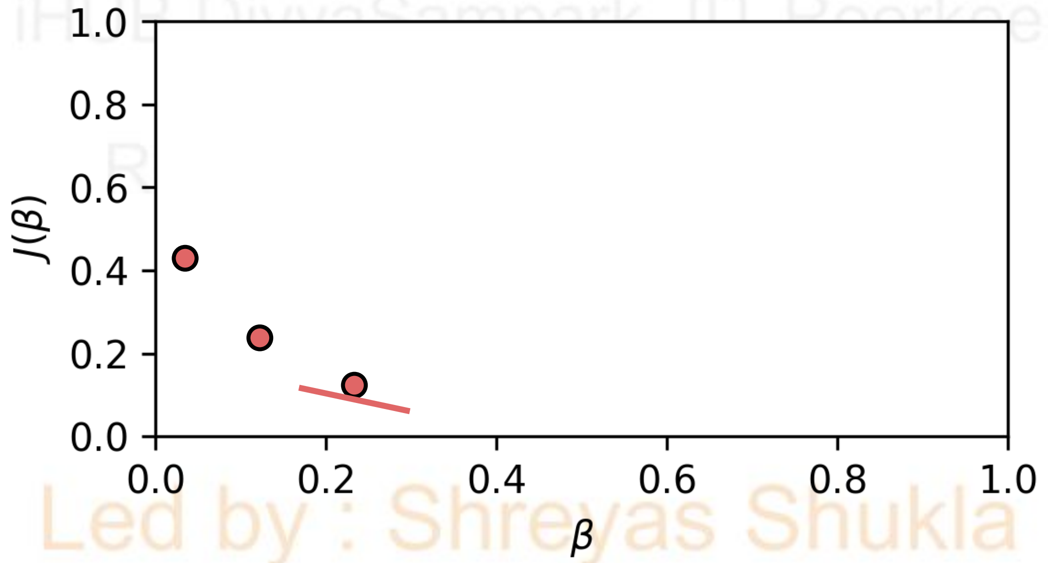
Repeat



iHUB DivyaSampark, IIT Roorkee
Led by : Shreyas Shukla

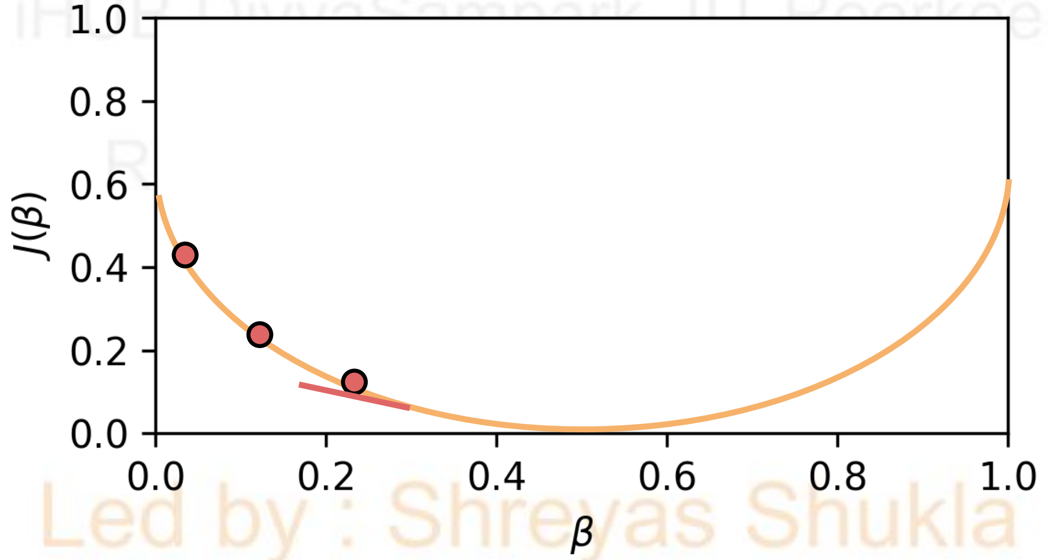
Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

Repeat



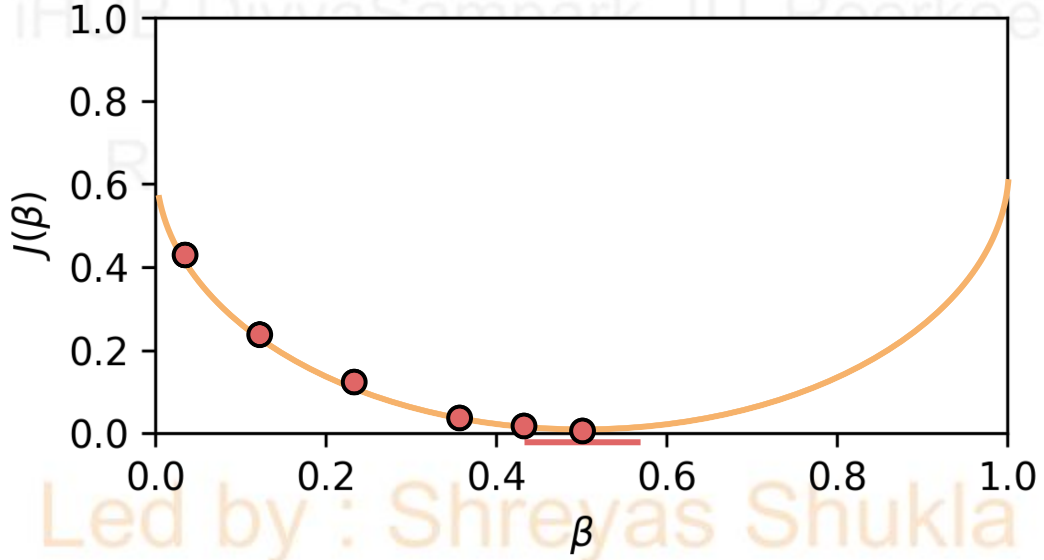
iHUB DivyaSampark, IIT Roorkee
Led by : Shreyas Shukla

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)
iHUB DivyaSampark, IIT Roorkee
Led by : Shreyas Shukla



Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

Eventually we will find the Beta that minimizes the cost function!

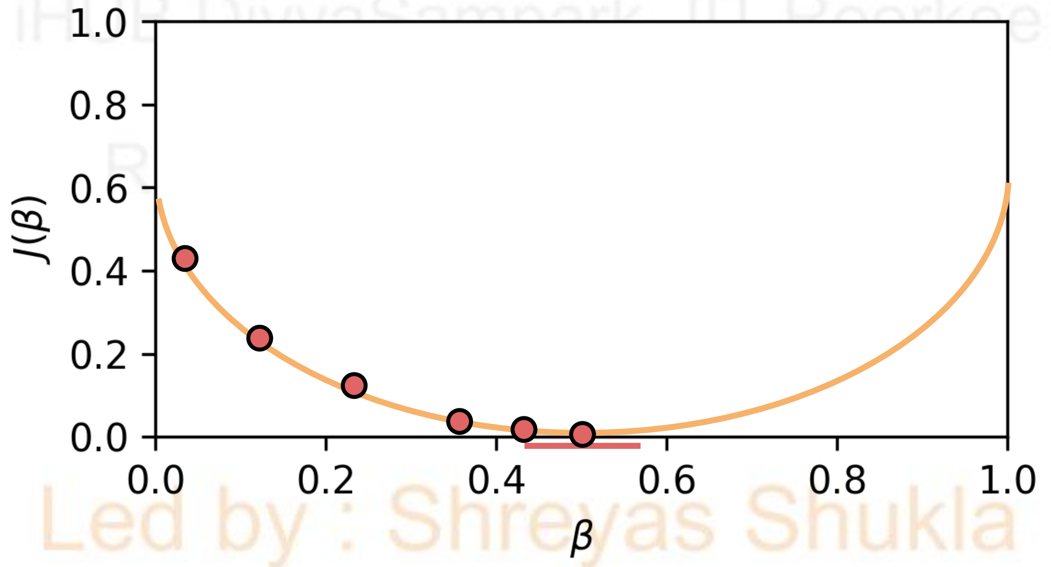


iHUB DivyaSampark, IIT Roorkee
Led by : Shreyas Shukla

Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

Steps are proportional to negative gradient!
Steeper gradient at start gives larger steps.
Smaller gradient at end gives smaller steps.

iHUB DivyaSampark, IIT Roorkee



Led by : Shreyas Shukla

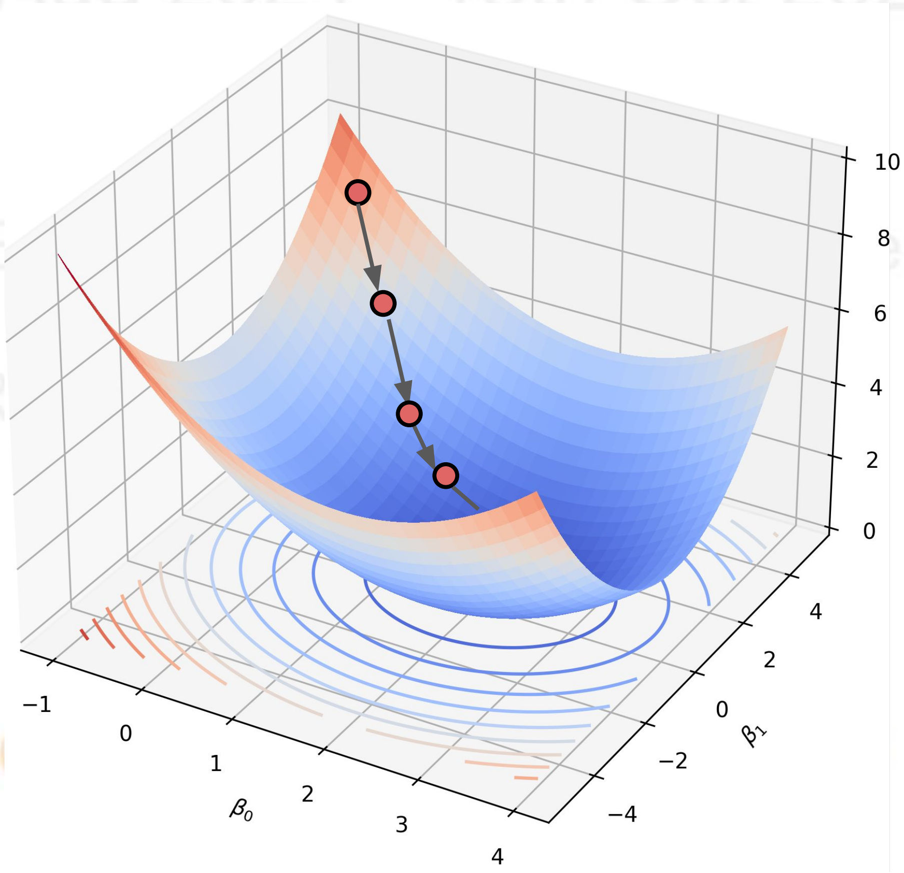
Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)
iHUB DivyaSampark, IIT Roorkee
and
Ritvij Bharat Private Limited
Led by : Shreyas Shukla

To further understand this, let's visualize this gradient descent search for two Beta values.
Process is still the same:

- Calculate gradient at point.
- Move in a step size proportional to negative gradient.
- Repeat until minimum is found.

Mastering Machine Learning with Python

(27th Aug 2024 - 18th Oct 2024)



Mastering Machine Learning with Python
(27th Aug 2024 - 18th Oct 2024)

Cost Function/Loss Function: It basically ***determines how well a machine learning model performs for a given dataset.*** It calculates the difference between the expected value and predicted value and represents it as a single term.

iHUB DivyaSampark, IIT Roorkee

Gradient Descent: "Gradient Descent is an optimization algorithm which is used for optimizing the cost function or error in the model." It enables the models to take the gradient or direction to reduce the errors by reaching to least possible error. Gradient descent is an iterative process where the model gradually converges towards a minimum value, and if the model iterates further than this point, it produces little or zero changes in the loss.

Led by : Shreyas Shukla