Solving Regression Problems

Swetha Kolalapudi
CO-FOUNDER, LOONYCORN
www.loonycorn.com

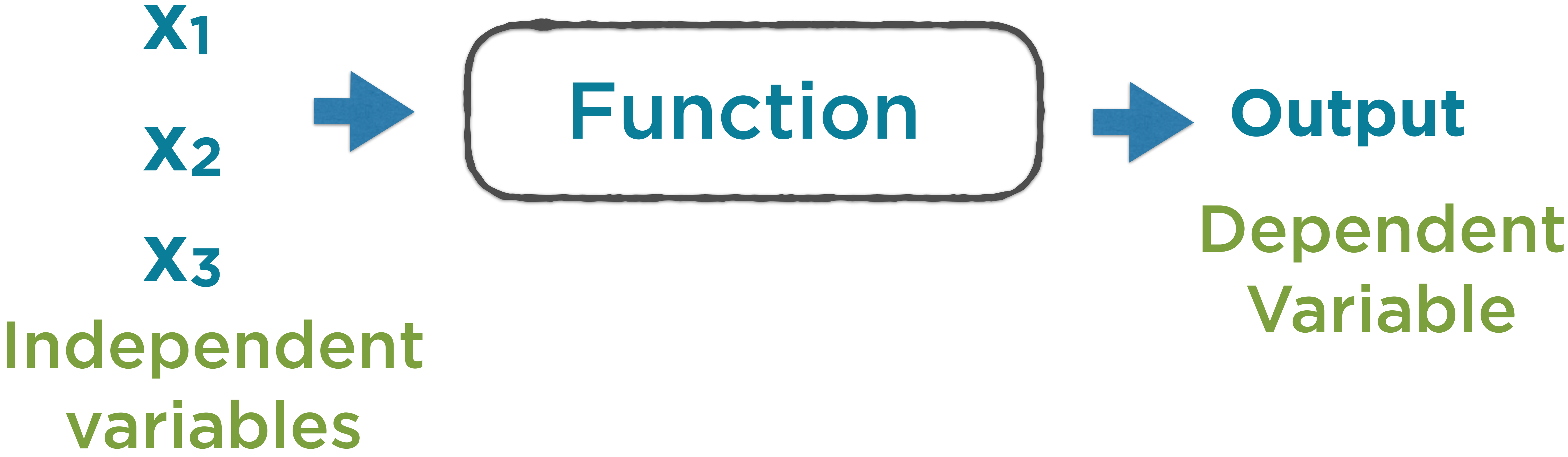
Overview

Understand how Linear Regression can be applied to find the Beta of a stock

Understand the Stochastic Gradient Method for Linear Regression

Tweak the parameters of SGD for better performance

Implement Linear Regression in Python


x_1
x_2
x_3
Independent variables
Function
Output
Dependent Variable

Function

Quantify the relationship between different variables

Function

Assume a specific form for this function

Function

If you assume the function is linear

Linear Regression

A Linear Function

x_1

x_2

x_3

Function

$$b_0 + b_1 x_1 + b_2 x_2 + \cdots$$

Linear Regression

x_1

x_2

x_3

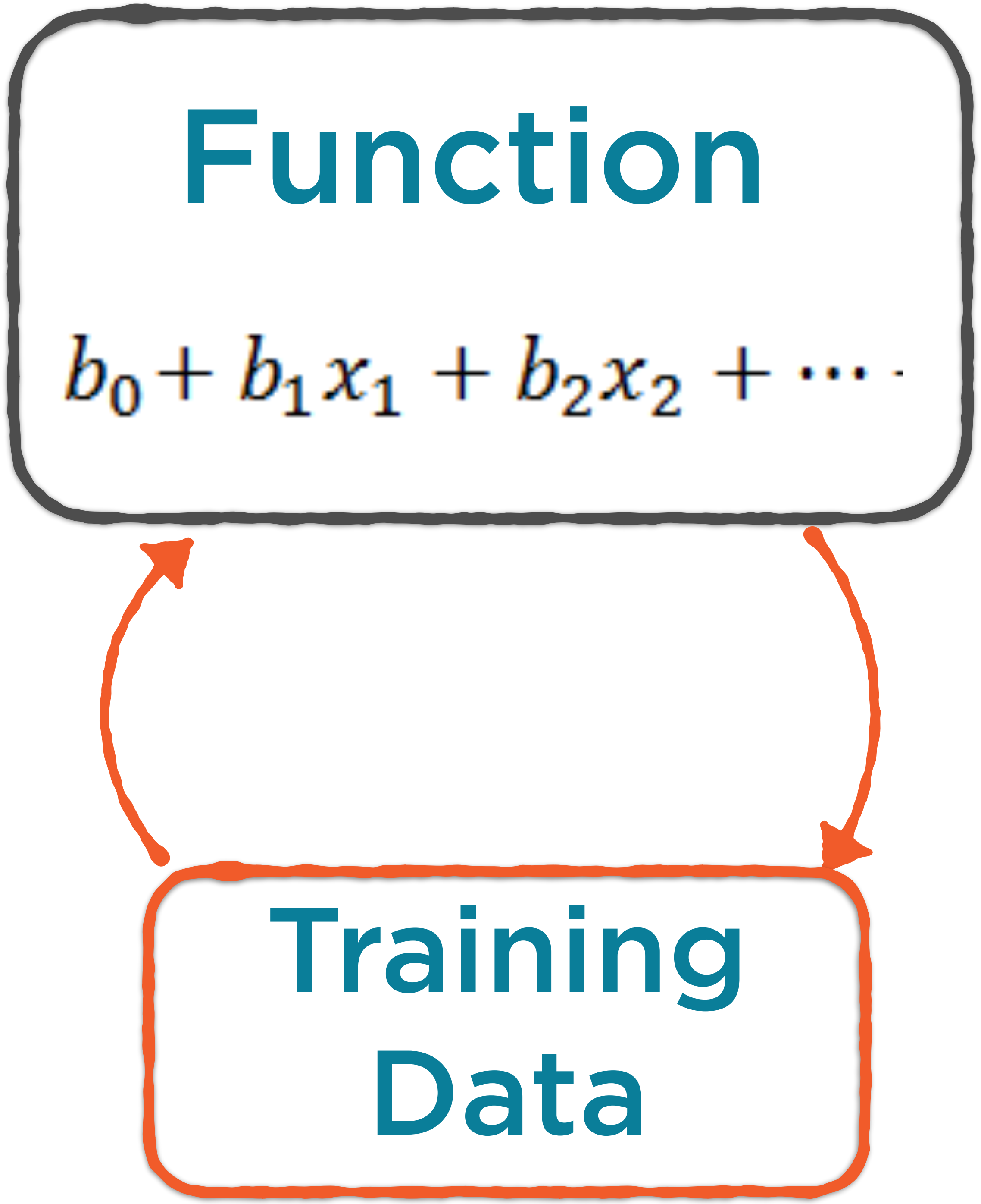
Function

$b_0 + b_1 x_1 + b_2 x_2 + \cdots$

Output

The b's are constants

Linear Regression



Solve for the values of these constants

Using past data

Function

$$b_0 + b_1x_1 + b_2x_2 + \cdots$$

The b’s are called co-efficients

Demand Forecasting

Sales = 2* Marketing Spend + 0.5* Sales of Last week

Demand Forecasting

Sales = 2* Marketing Spend + 0.5* Sales of Last week

Dependent Variable

Demand Forecasting

Sales = 2* **Marketing Spend** + 0.5* **Sales of Last week**

Independent Variables

Demand Forecasting

Sales = 2* Marketing Spend +
0.5* Sales of Last week

Co-efficients
found using Linear
Regression

The CAPM Model

$$R_i = R_f + \beta_i (R_m - R_f)$$

$$R_i - R_f = \beta_i (R_m - R_f)$$

If we rearrange this equation

The CAPM Model

$$R_i - R_f = \beta_i (R_m - R_f)$$

Linear Regression

The CAPM Model

$$R_i - R_f = \beta_i (R_m - R_f)$$

Dependent Variable

The CAPM Model

$$R_i - R_f = \beta_i (R_m - R_f)$$

Independent Variable

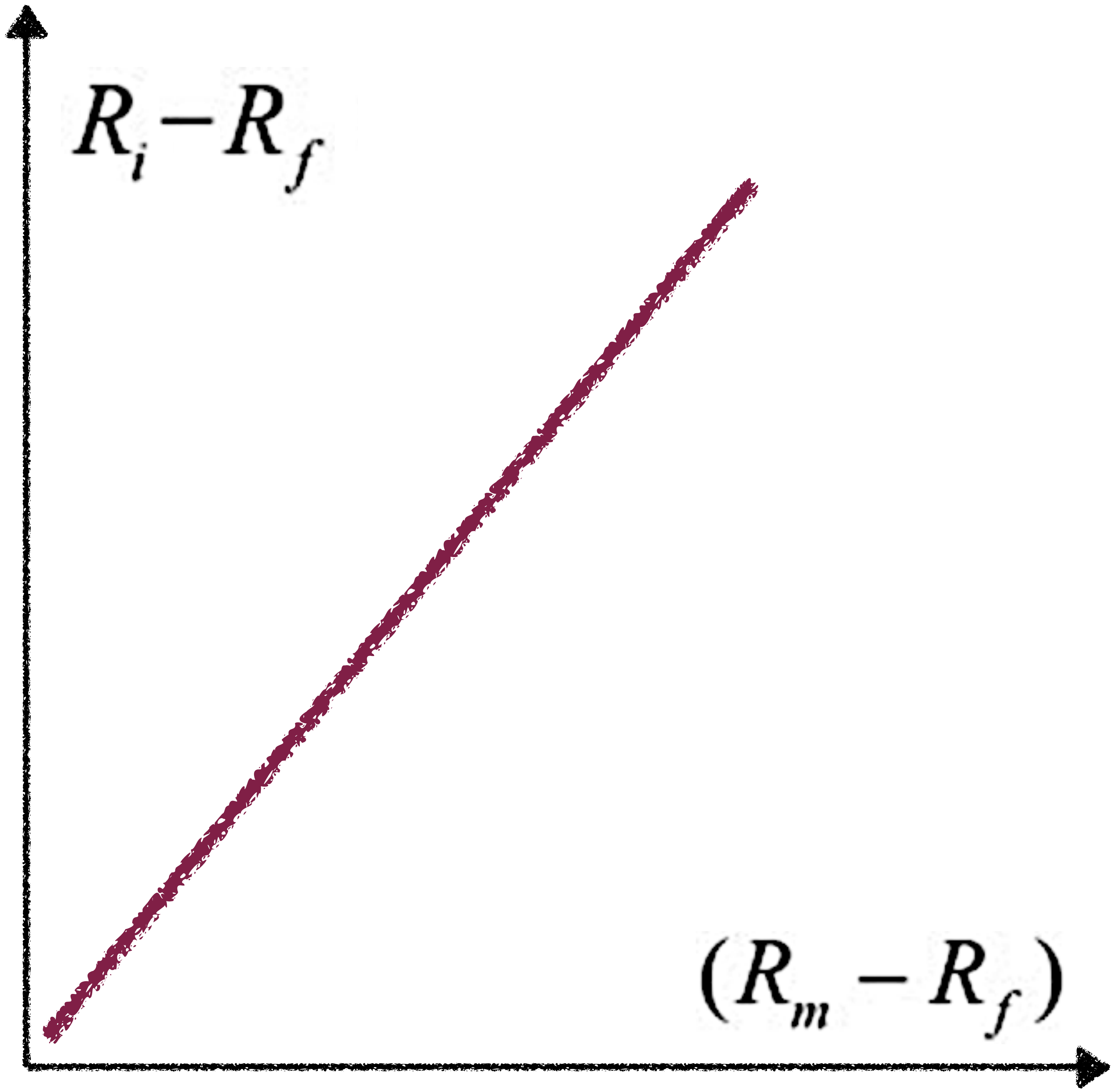
The CAPM Model

$$R_i - R_f = \beta_i (R_m - R_f)$$

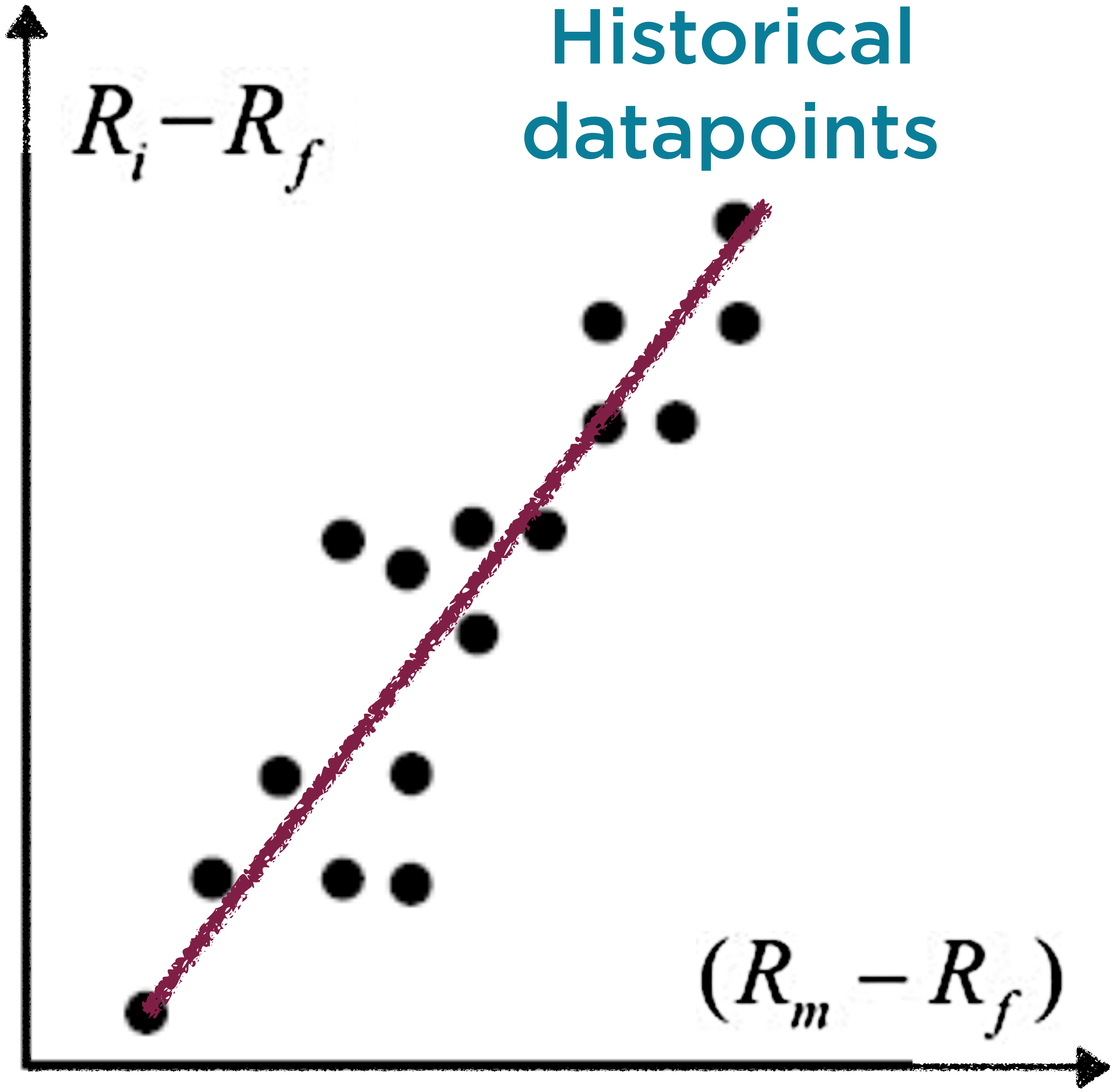
Equation of a line passing through the origin

$$R_i - R_f = \beta_i (R_m - R_f)$$

Slope of the line

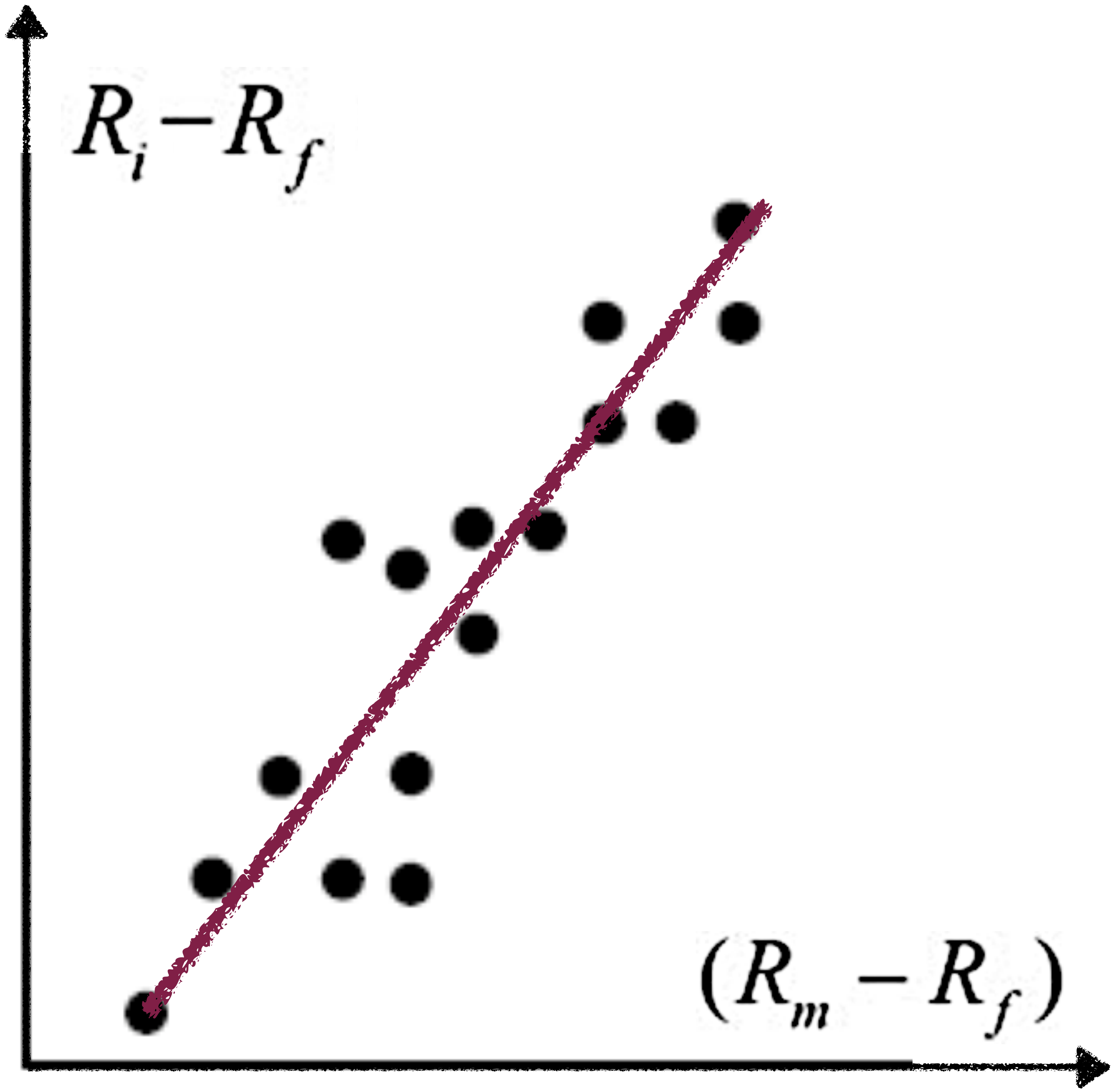


Linear Regression will find the line that is the best fit



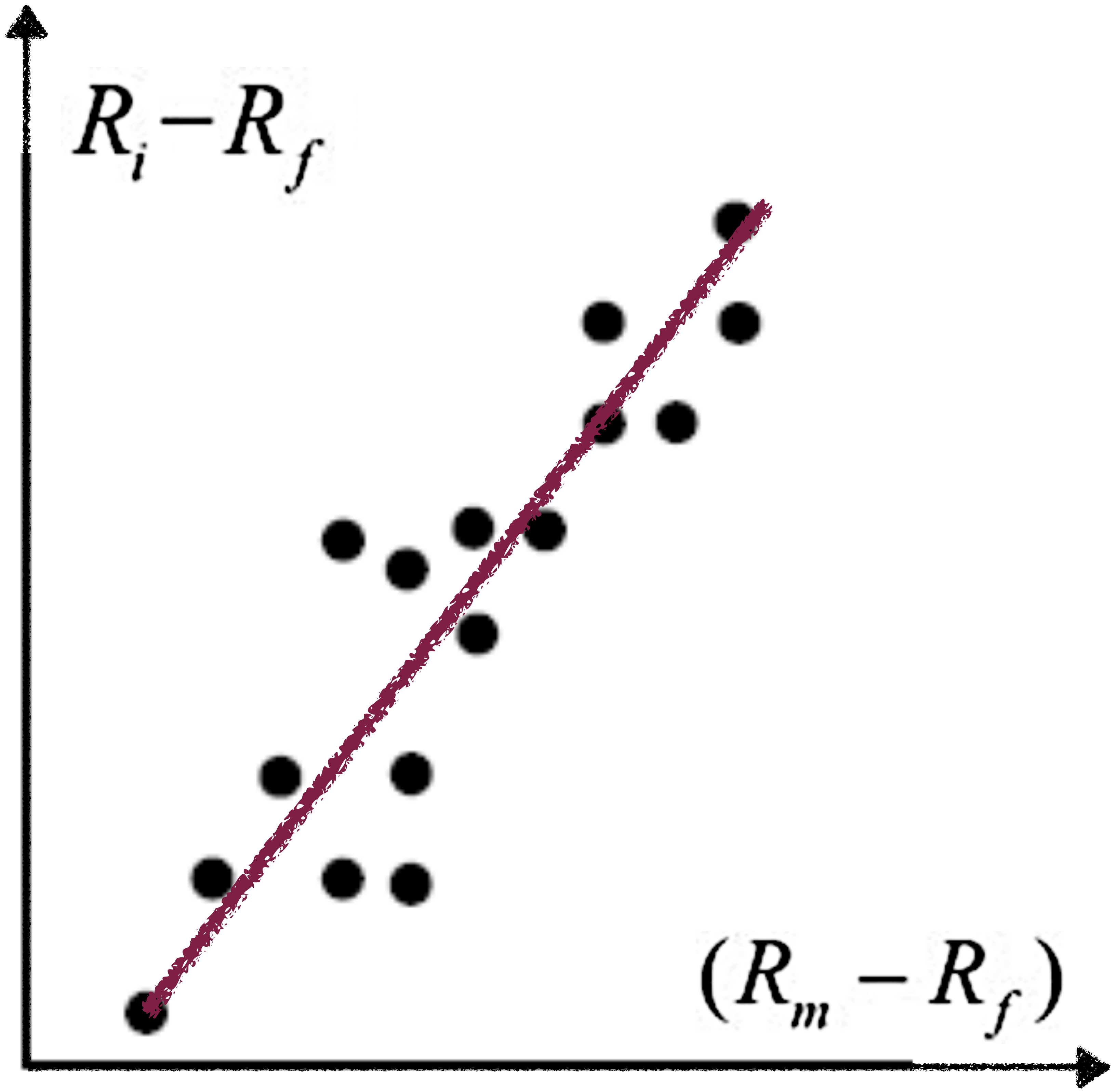
$$R_i - R_f = \beta_i (R_m - R_f)$$

The slope of that line will be our Beta



$$R_i - R_f = \beta_i (R_m - R_f)$$

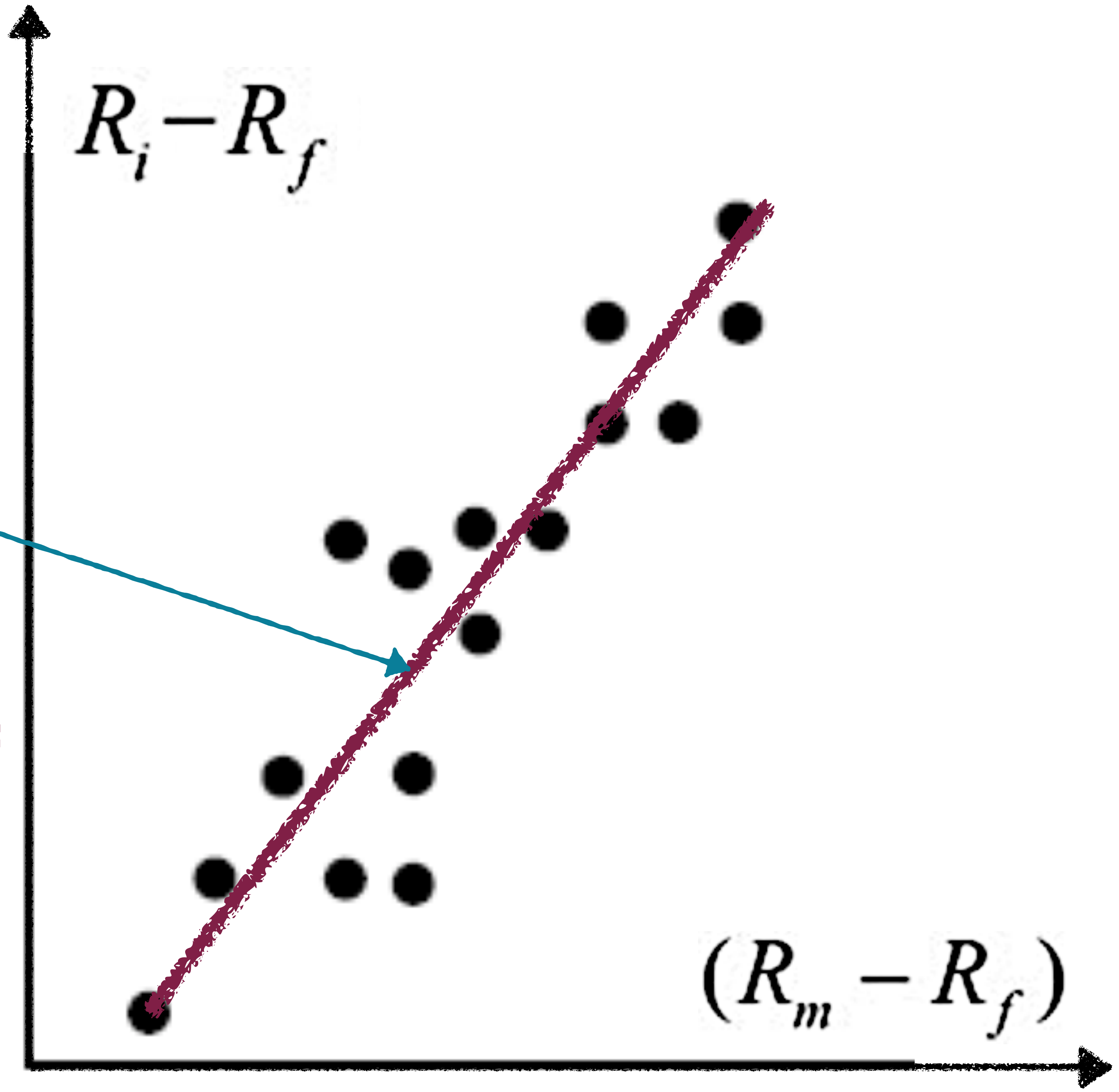
Simple Linear Regression with 1 Variable



Once we find Beta

$$R_i - R_f = \beta_i (R_m - R_f)$$

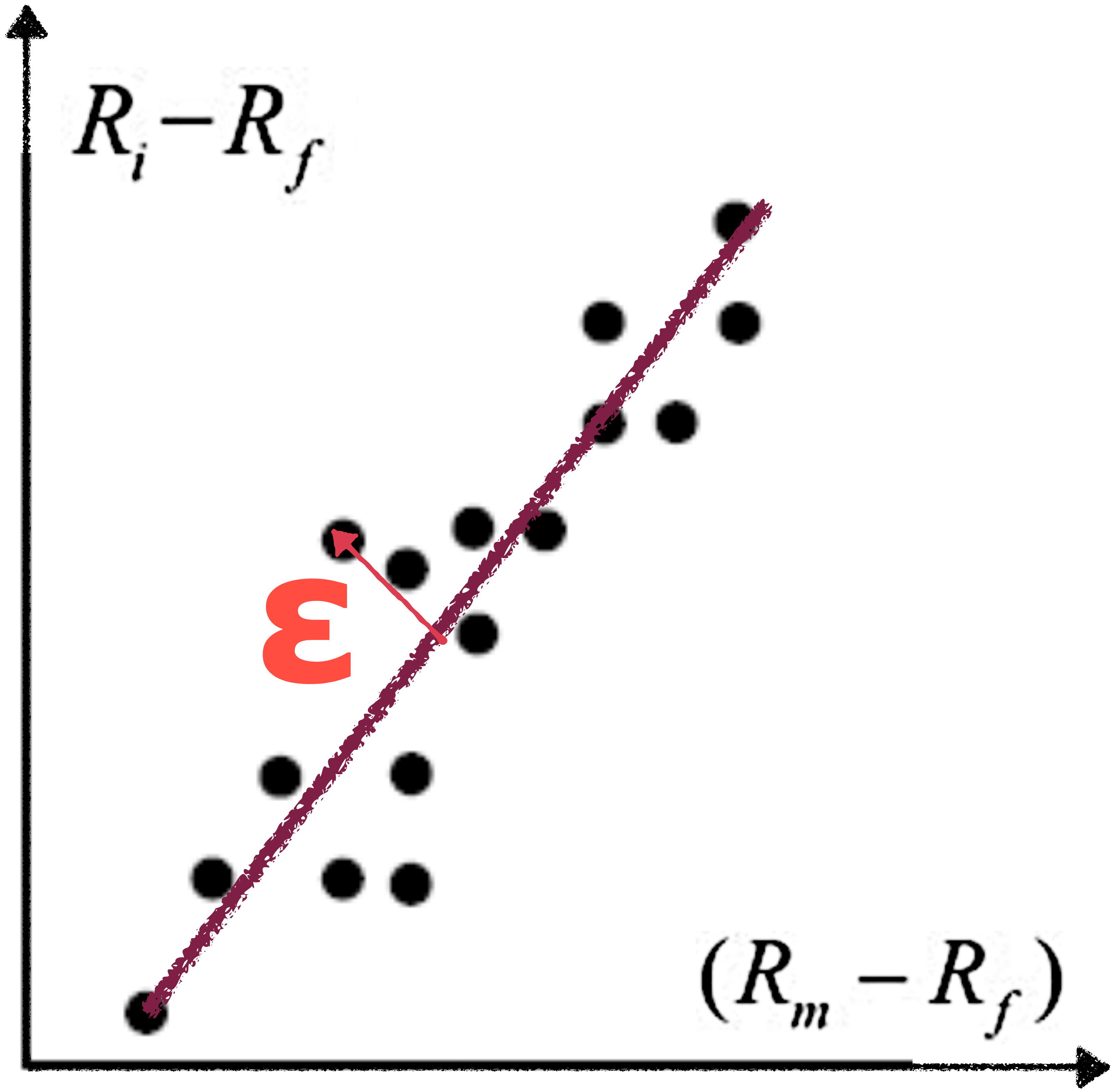
The predicted value of returns using the line



$$R_i - R_f = \beta_i (R_m - R_f)$$

Error

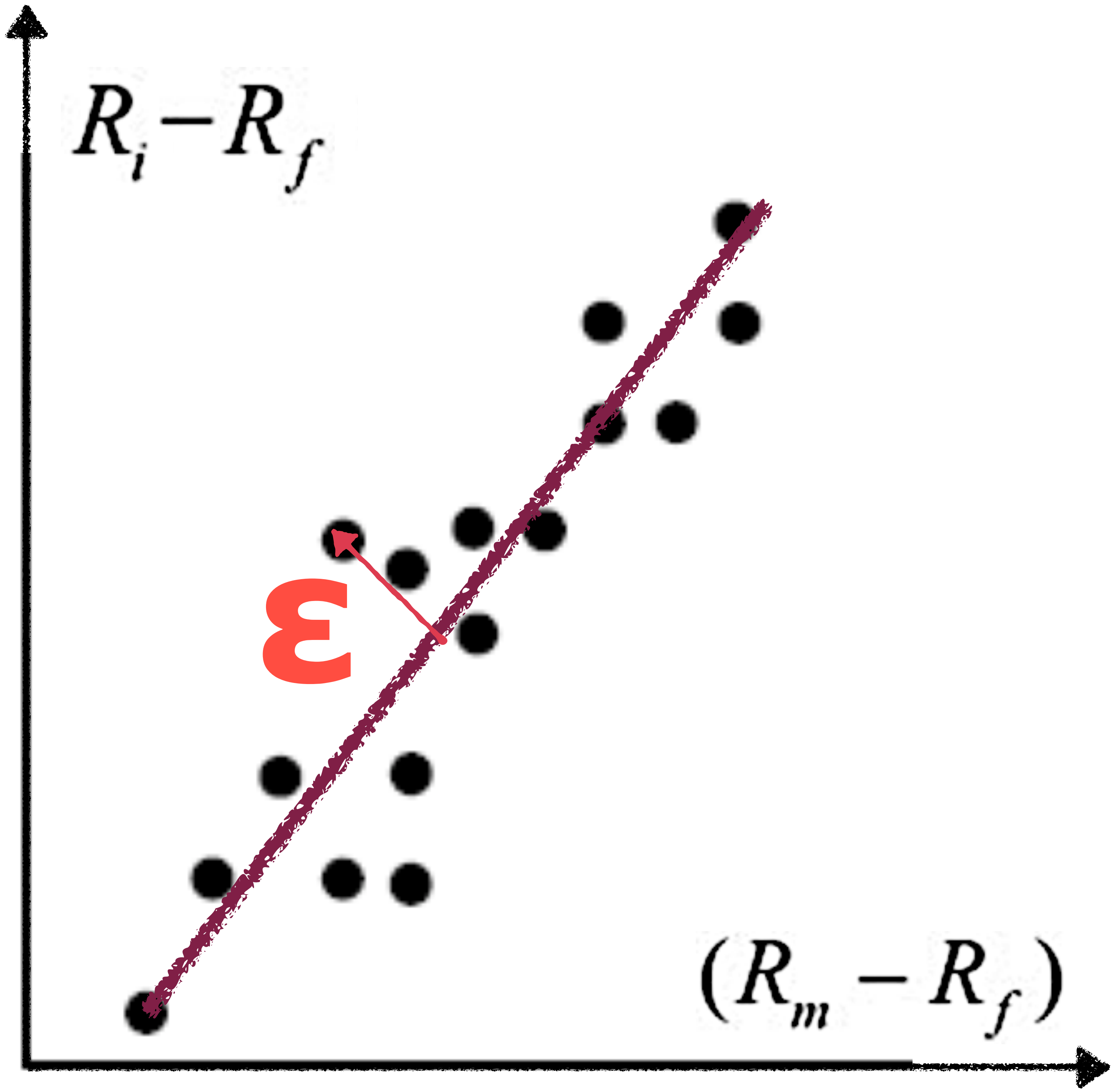
(Distance between the actual point and the line)



Error

The left over parts of the dependent variable, not explained by the independent variables

Residuals



Minimizing Error

Linear Regression tries to minimize this error for the training data

Minimizing Error

One of the techniques to help with this

Stochastic Gradient Descent

Stochastic Gradient Descent

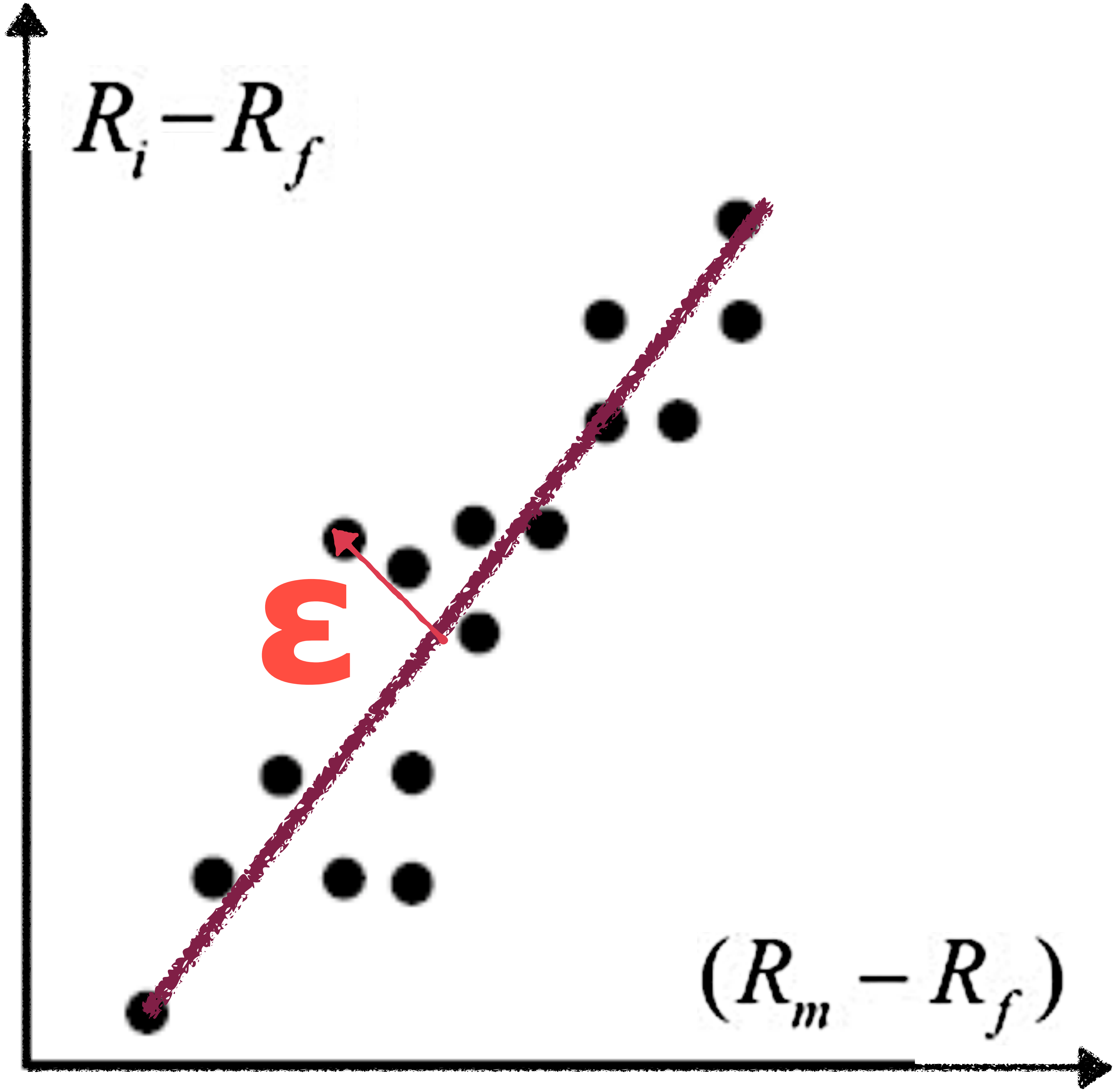
The goal is to minimize error

$$\text{Error} = \frac{1}{N}\sum_{i=1}^{N} \varepsilon_i^2$$

N -> number of historical datapoints

$$\text{Error} = \frac{1}{N}\sum_{i=1}^{N} \varepsilon_i^2$$

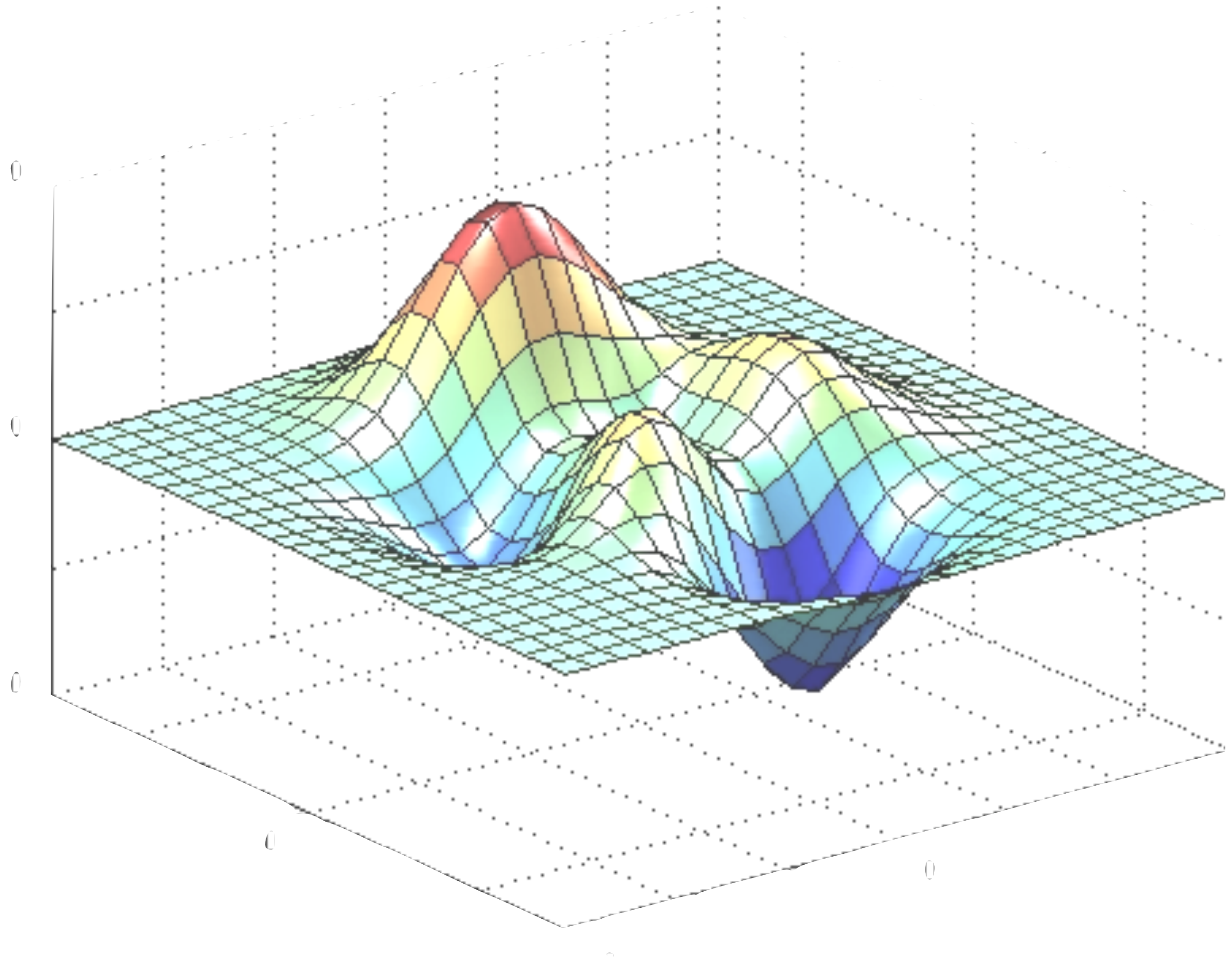
A function of the slope and intercept of the line



Stochastic Gradient Descent

$$\text{Error} = \frac{1}{N}\sum_{i=1}^{N} \varepsilon_i^{2}$$

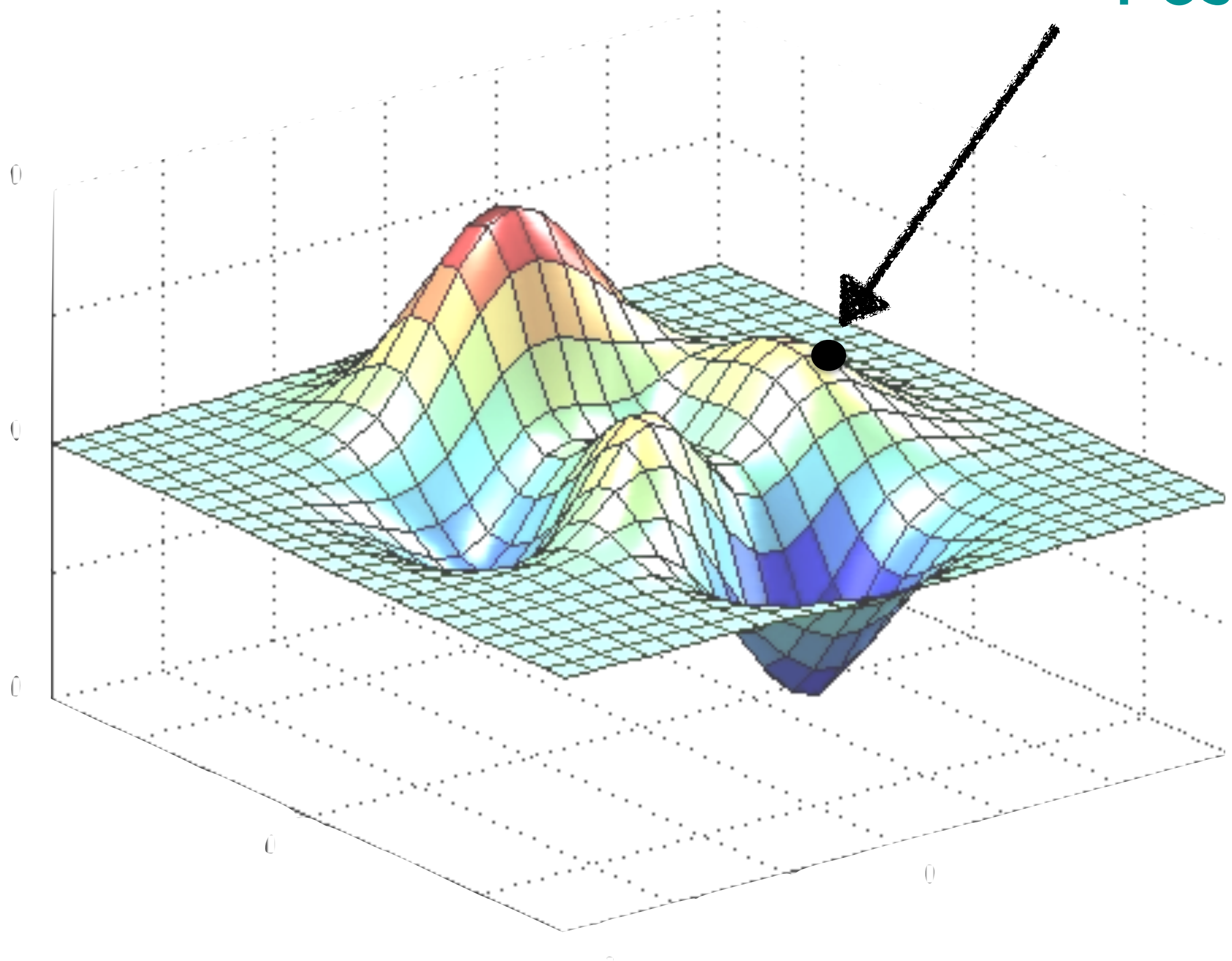
The graph represents the error for different values of the slope and intercept



Stochastic Gradient Descent

1. Initialize some value for the slope and intercept

2. Find the current value of the error function

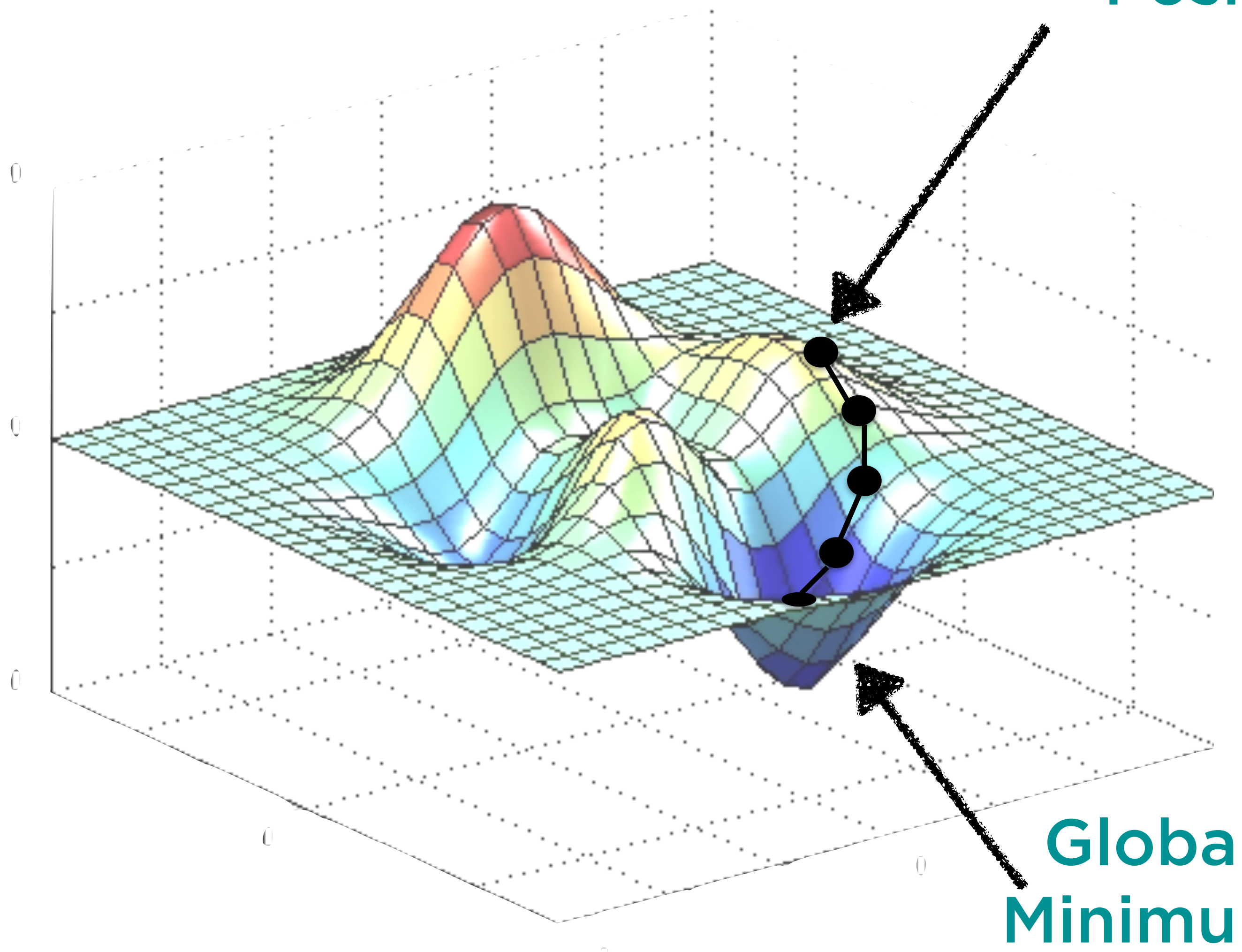


Stochastic Gradient Descent

3. Find the slope at the current point and move slightly downwards in that direction

4. Repeat until you reach a minimum

(or) stop after certain number of iterations

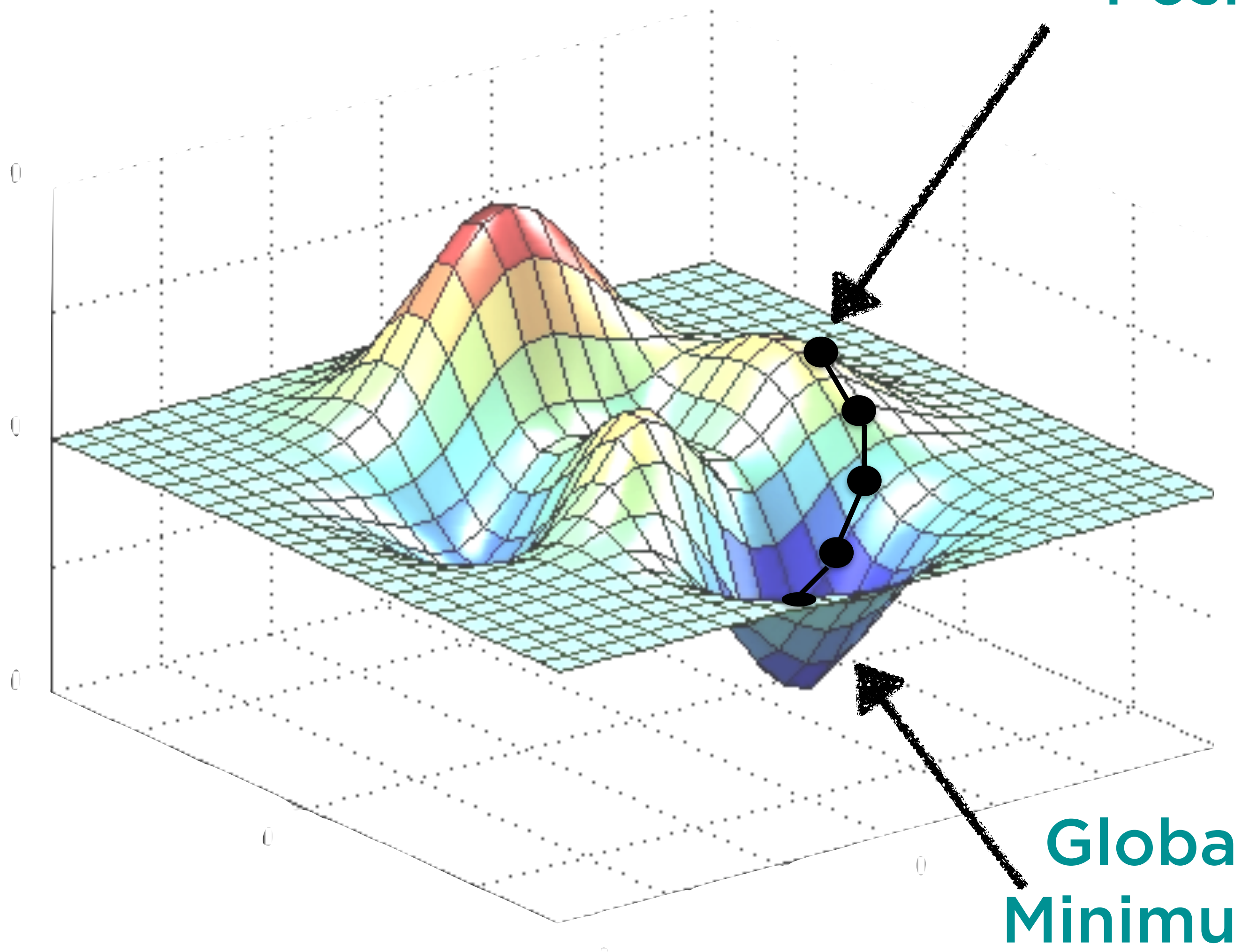


Stochastic Gradient Descent

When you implement Stochastic Gradient Descent

Step Size, Number of Iterations

are parameters to experiment with



The CAPM Model for Google

$$R_i - R_f = \beta_i (R_m - R_f)$$

Find the Beta for Google

The CAPM Model for Google

$$R_i - R_f = \beta_i(R_m - R_f)$$

Returns of Google
- Risk Free Rate

The CAPM Model for Google

$$R_i - R_f = \beta_i(R_m - R_f)$$

Returns of an index that represents the market

The CAPM Model for Google

$$R_i - R_f = \beta_i(R_m - R_f)$$

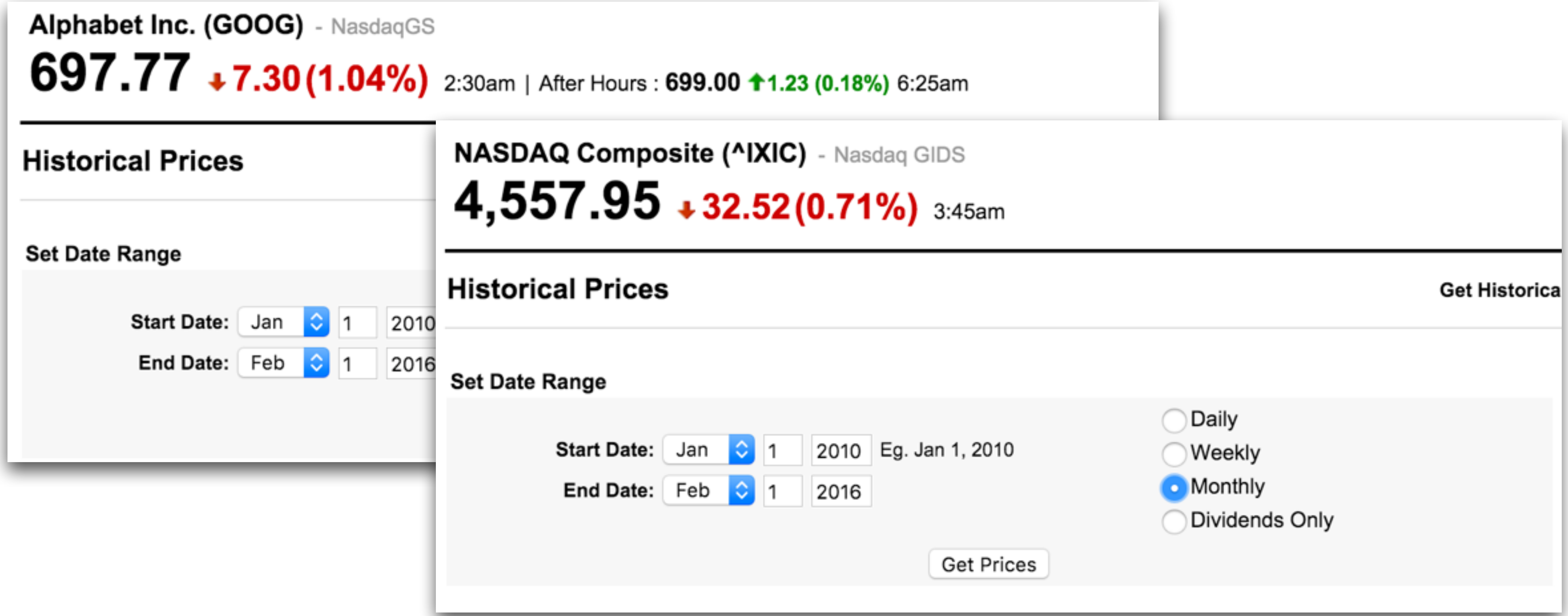
Returns of NASDAQ

The CAPM Model for Google

$$R_i - R_f = \beta_i (R_m - R_f)$$

Linear Regression in Python

Step 1: Download Historical Prices for Google and Nasdaq from a financial site (Yahoo Finance)



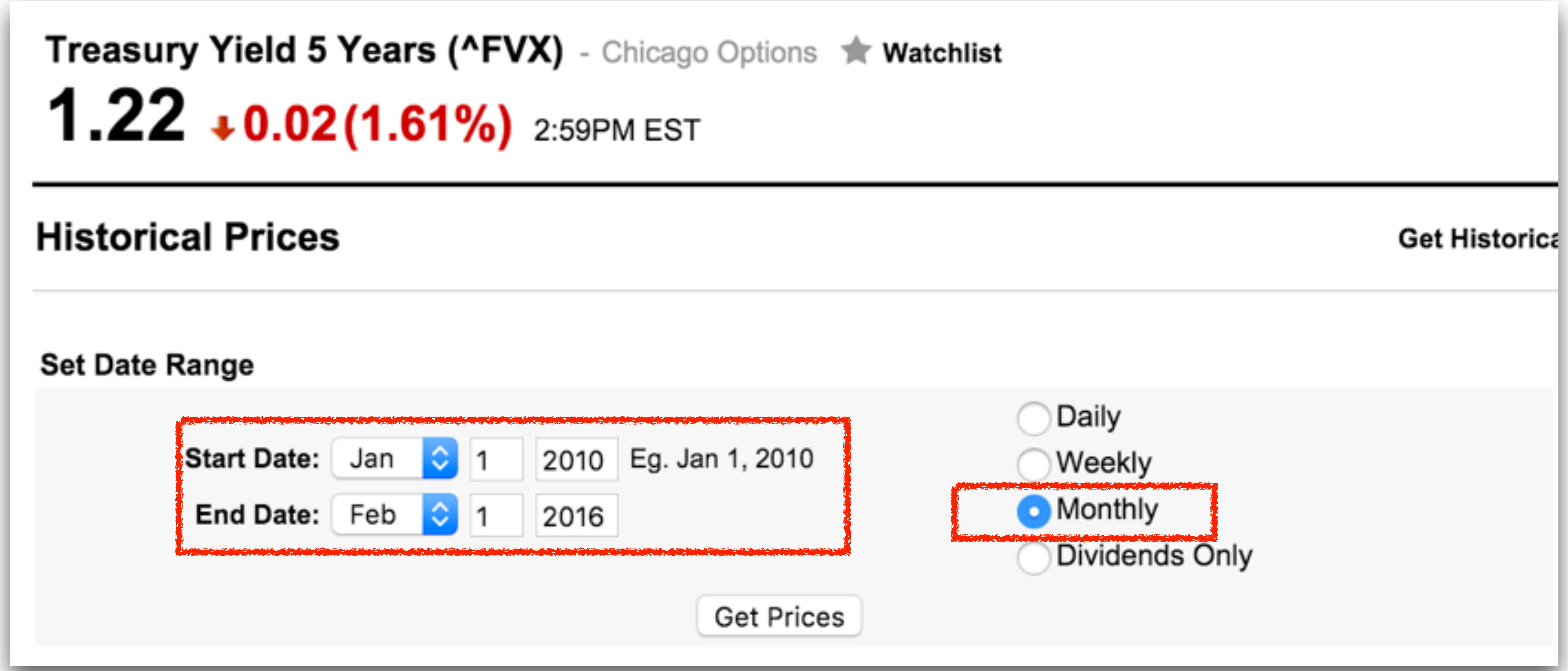
Step 2: Convert the prices to Returns

Date	Open	High	Low	Close	Volume	Adj Close
2/1/2016	750.46	757.86	743.27	752	10278400	752
1/4/2016	743	752	673.26	742.95	2632600	742.95
12/1/2015	747.11	779.98	724.17	758.88	2026100	758.88
11/2/2015	711.06	762.708	705.85	742.6	1801600	742.6
10/1/2015	608.37	730	599.85	710.81	2333600	710.81
9/1/2015	602.36	650.9	589.38	608.42	2398400	608.42
8/3/2015	625.34	674.9	565.05	618.25	2661500	618.25
7/1/2015	524.73	678.64	515.18	625.61	2955500	625.61
6/1/2015	536.79	543.74	520.5	520.51	1660600	520.51
5/1/2015	538.43	544.19	521.085	532.11	1723100	532.11

Step 2: Convert the prices to Returns

$$\text{Monthly Return} = \frac{(\text{New Price - Old Price})}{\text{Old Price}}$$

Step 3: Compute the risk free rate of return using the yields of 5 year Treasury bonds



Step 3: Compute the risk free rate of return using the yields of 5 year Treasury bonds

Tbond
Adj Close
1.383
1.335
1.758
1.654

Tbond
Yield
1%
1%
2%
2%

The Adj.Close column represents the yield %

Divide by 100 to compute the Yield

Step 4: Subtract the yields from Google and Nasdaq Returns

	GOOG	NASDAQ
Date	Returns	Returns
2/1/2016	1%	0%
1/4/2016	-2%	-8%
12/1/2015	2%	-2%
11/2/2015	4%	1%
10/1/2015	17%	9%
9/1/2015	-2%	-3%
8/3/2015	-1%	-7%
7/1/2015	20%	3%
6/1/2015	-2%	-2%
5/1/2015	-1%	3%

-

Tbond
Yield
1%
1%
2%
2%
2%
1%
2%
2%
2%
1%

=

GOOG	Nasdaq
r-rf	rm-rf
0%	-1%
-3%	-9%
0%	-4%
3%	-1%
15%	8%
-3%	-5%
-3%	-8%
19%	1%
-4%	-3%
-2%	1%

Step 5: Regress the adjusted Google returns against adjusted Nasdaq returns

`SGDRegressor`

from the Scikit-Learn Package

Demo

Compute the Returns of Google, Nasdaq and 5 year treasury bonds

Implement Linear Regression with SGD in Python

Summary

Understand how Linear Regression can be applied to find the Beta of a stock

Understand the Stochastic Gradient Method for Linear Regression

Tweak the parameters of SGD for better performance

Implement Linear Regression in Python