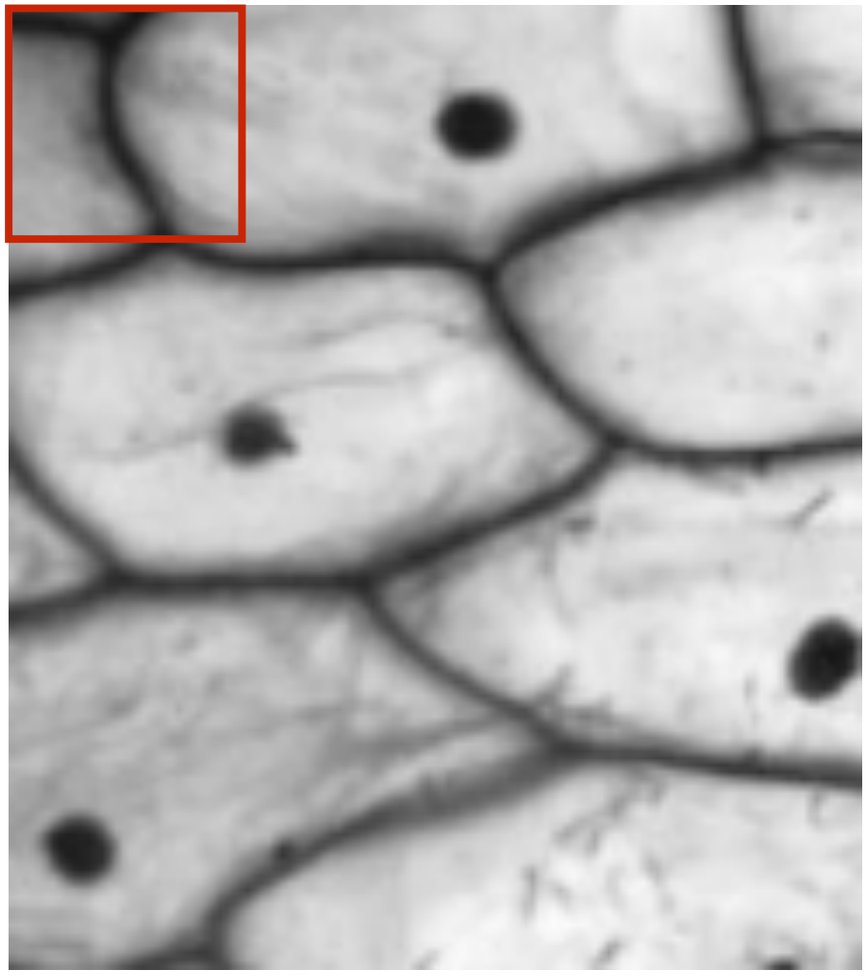
SSY097 - Image Analysis

Lecture 4 - Statistical Learning

Torsten Sattler
(slides adapted from Olof Enqvist)

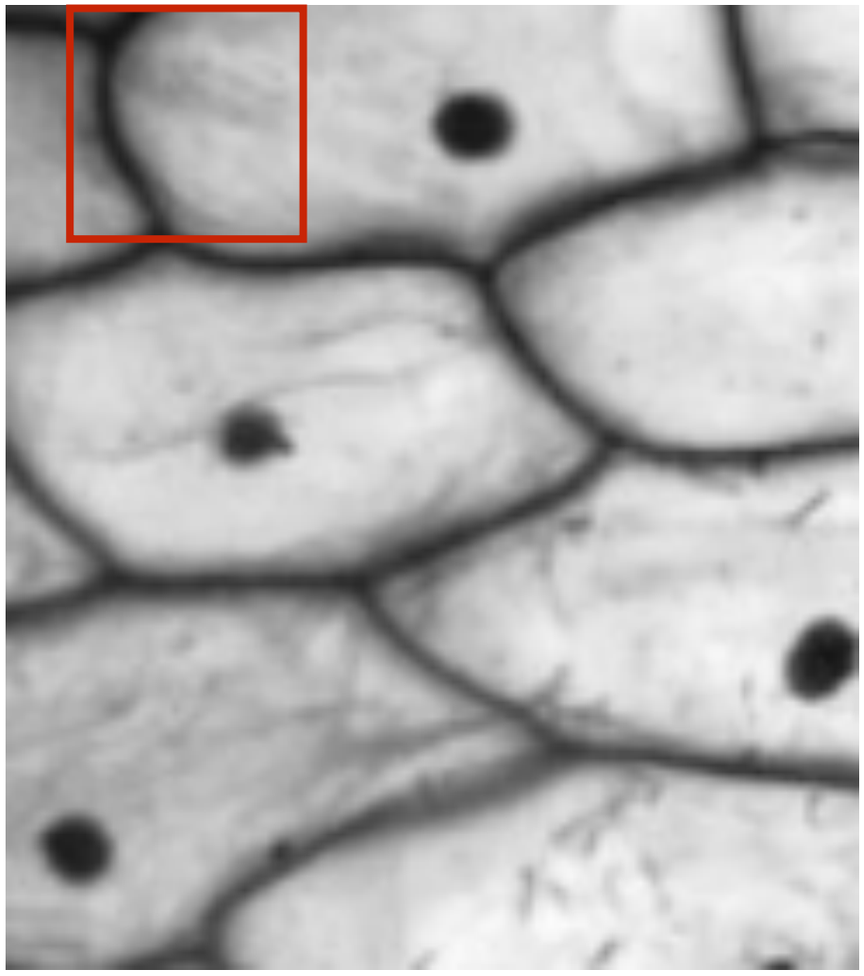
First Lecture



$$\cdot w < \tau$$

Sliding window classification

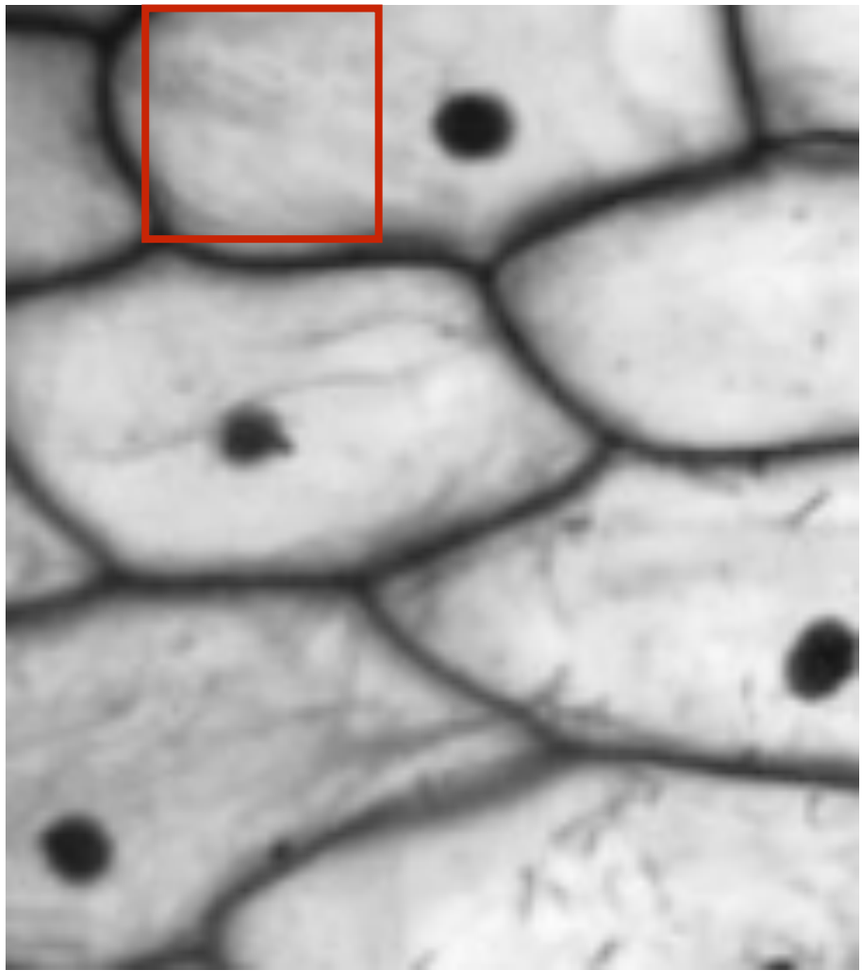
First Lecture



$\cdot w < \tau$

Sliding window classification

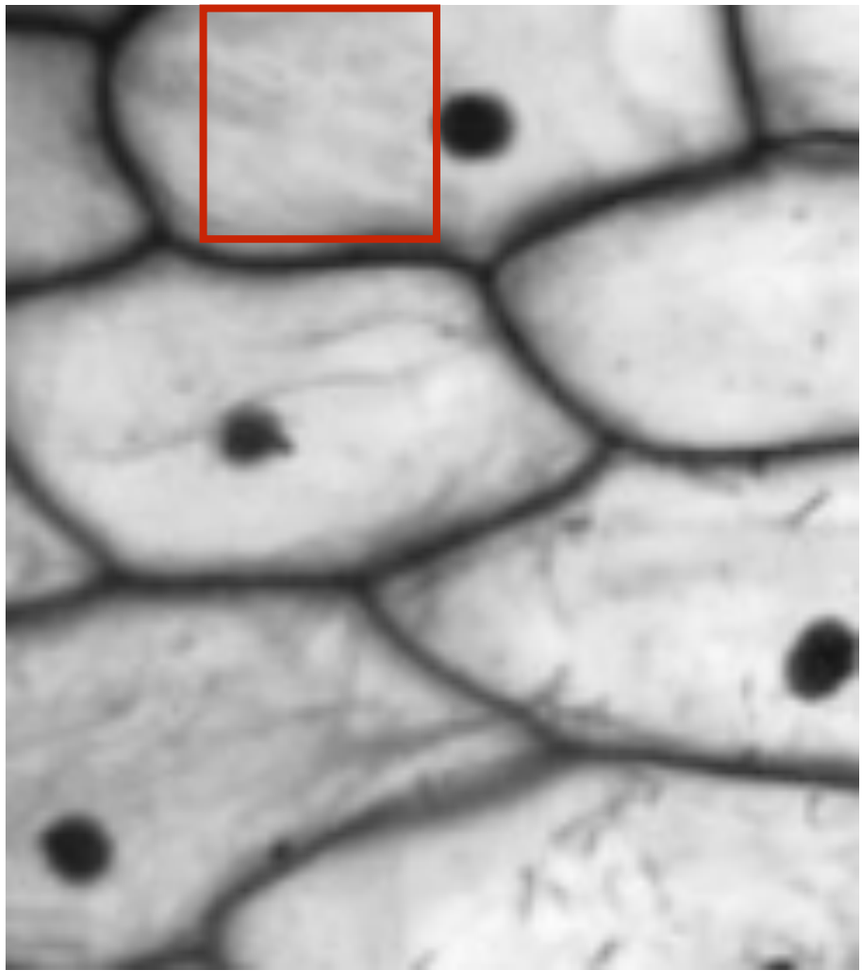
First Lecture



$\cdot w < \tau$

Sliding window classification

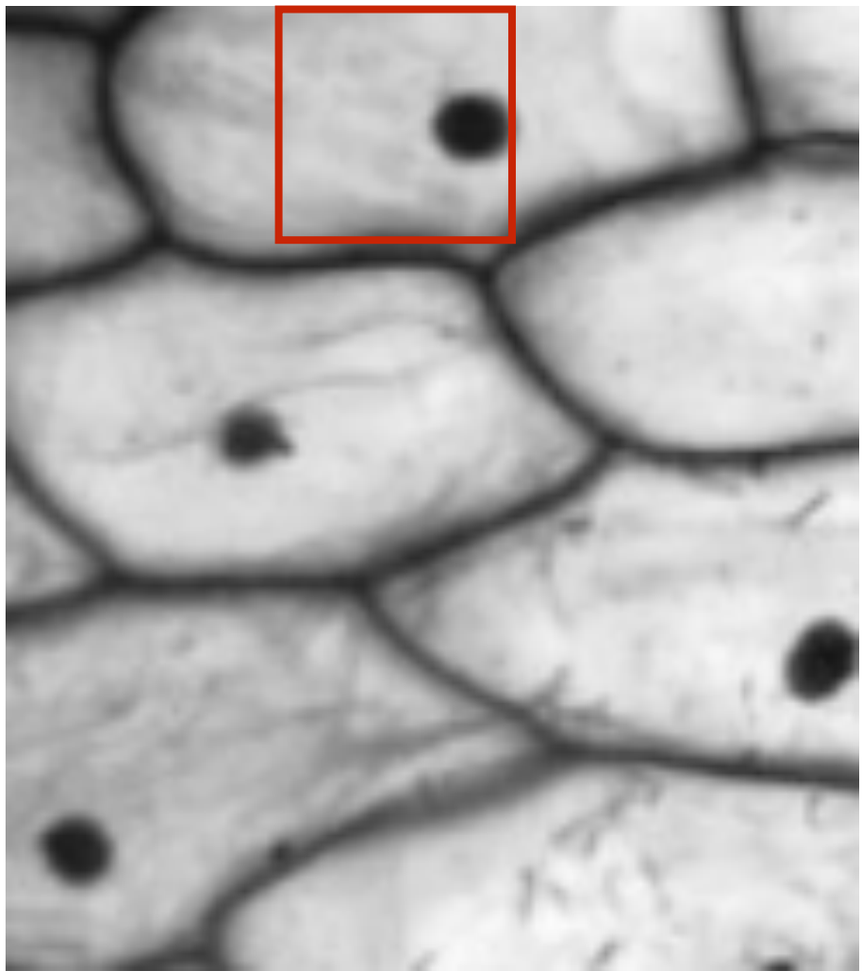
First Lecture



$\cdot w < \tau$

Sliding window classification

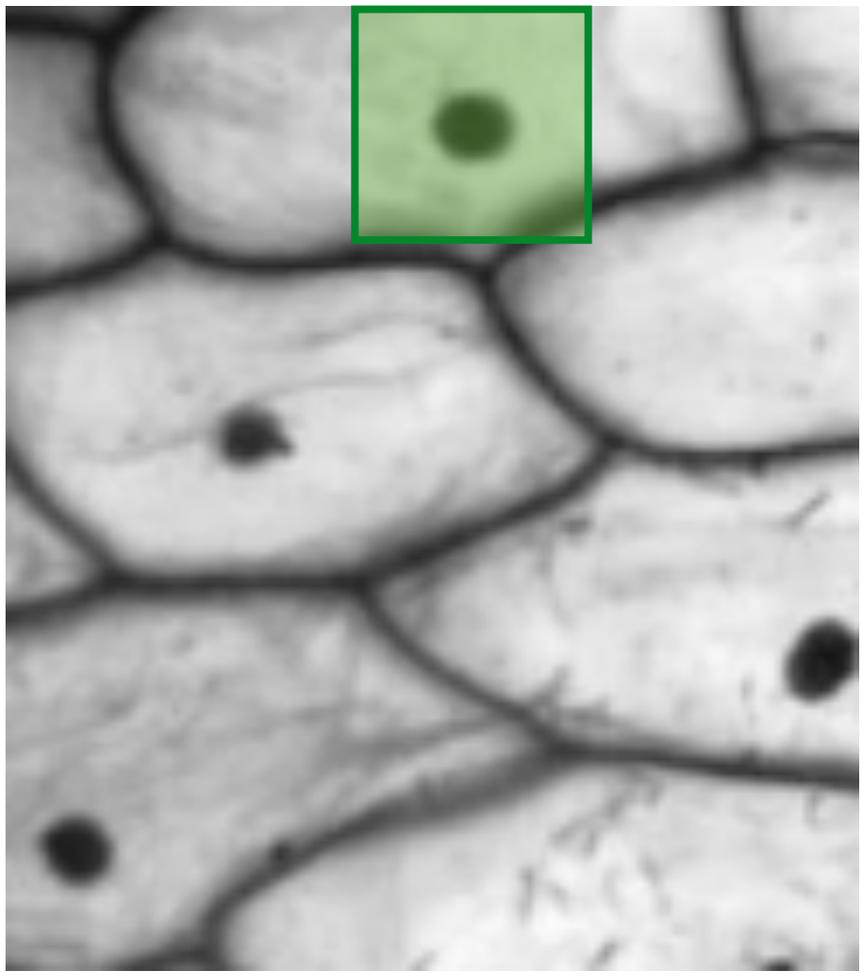
First Lecture



$\cdot w < \tau$

Sliding window classification

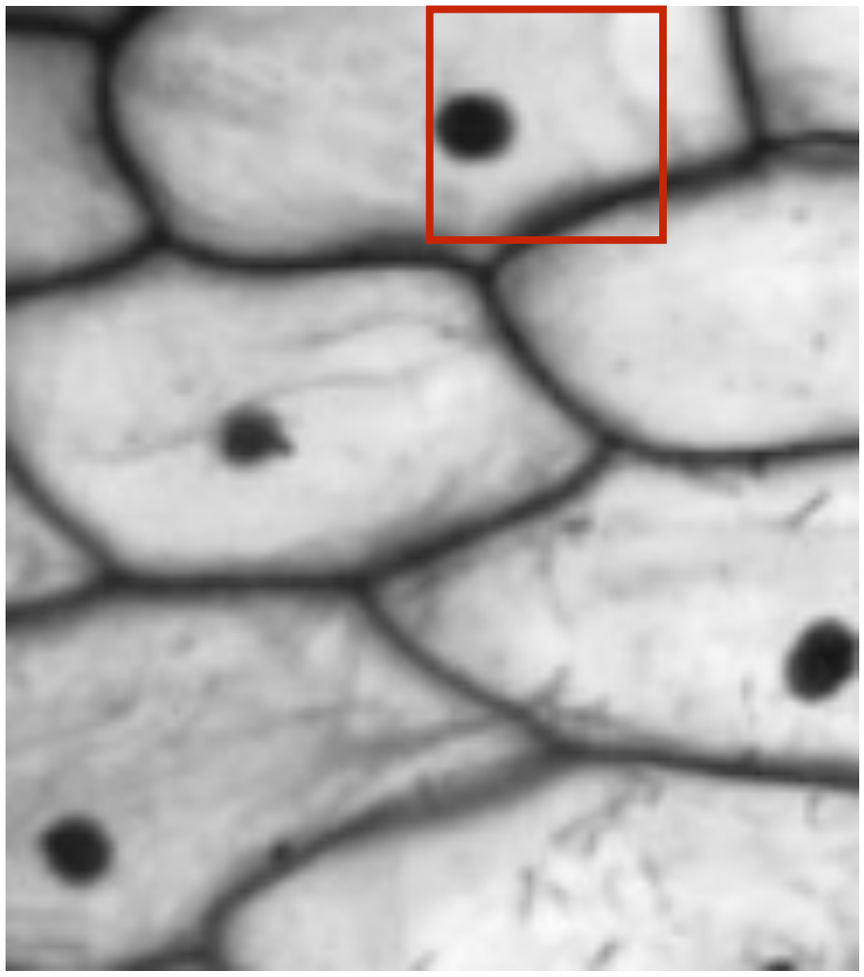
First Lecture



$\cdot w > \tau$

Sliding window classification

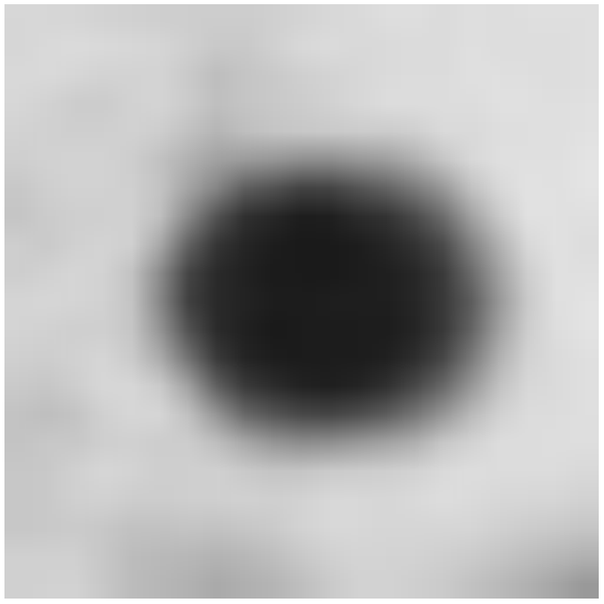
First Lecture



$\cdot w < \tau$

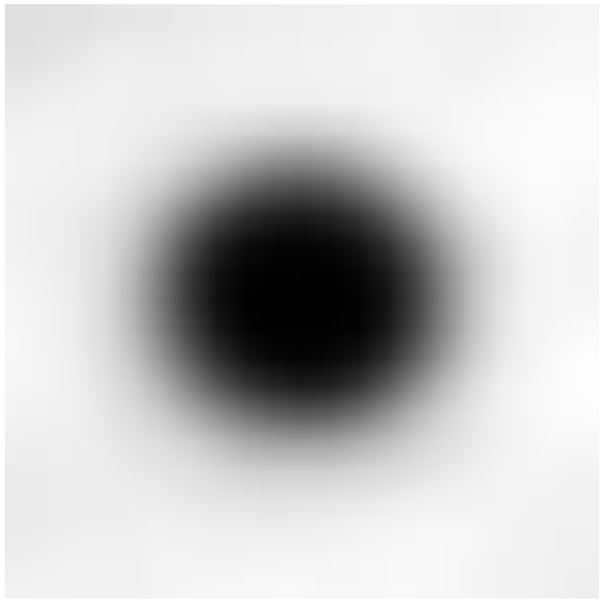
Sliding window classification

First Lecture

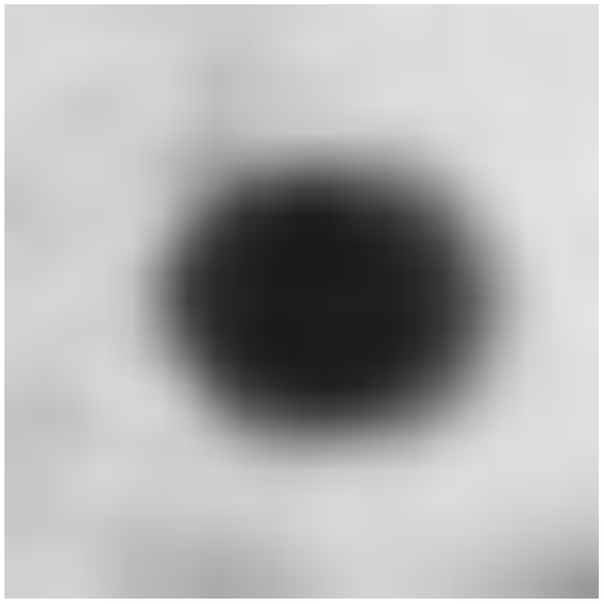
$$w \cdot$$  $$> \tau$$

Linear filters

First Lecture



average template



$$\cdot \qquad > \tau$$

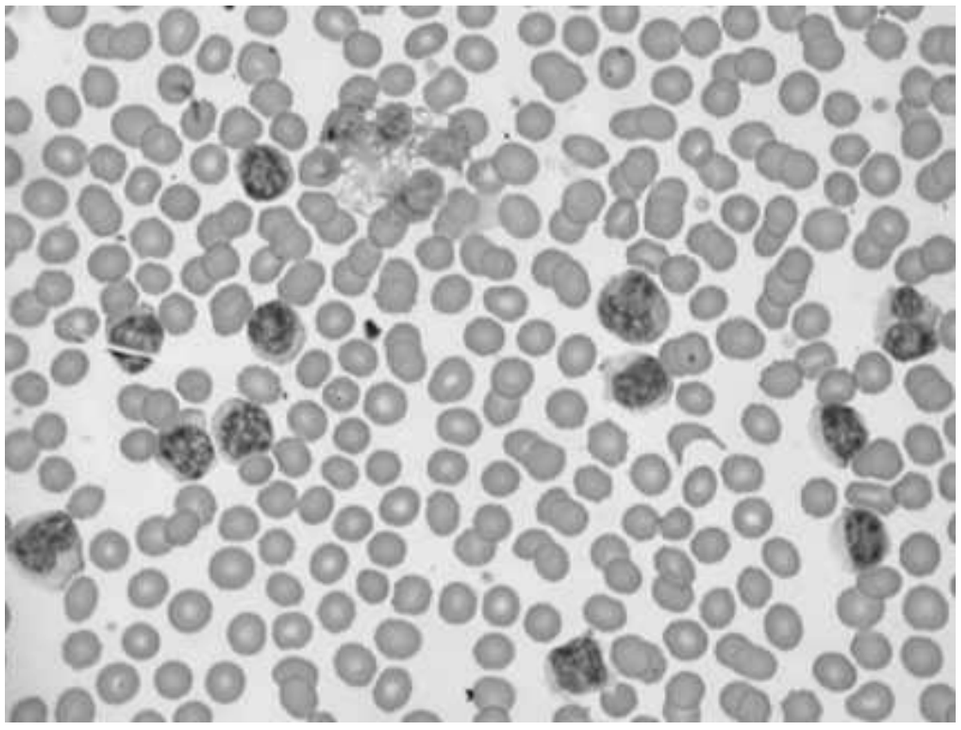
Linear filters

Today

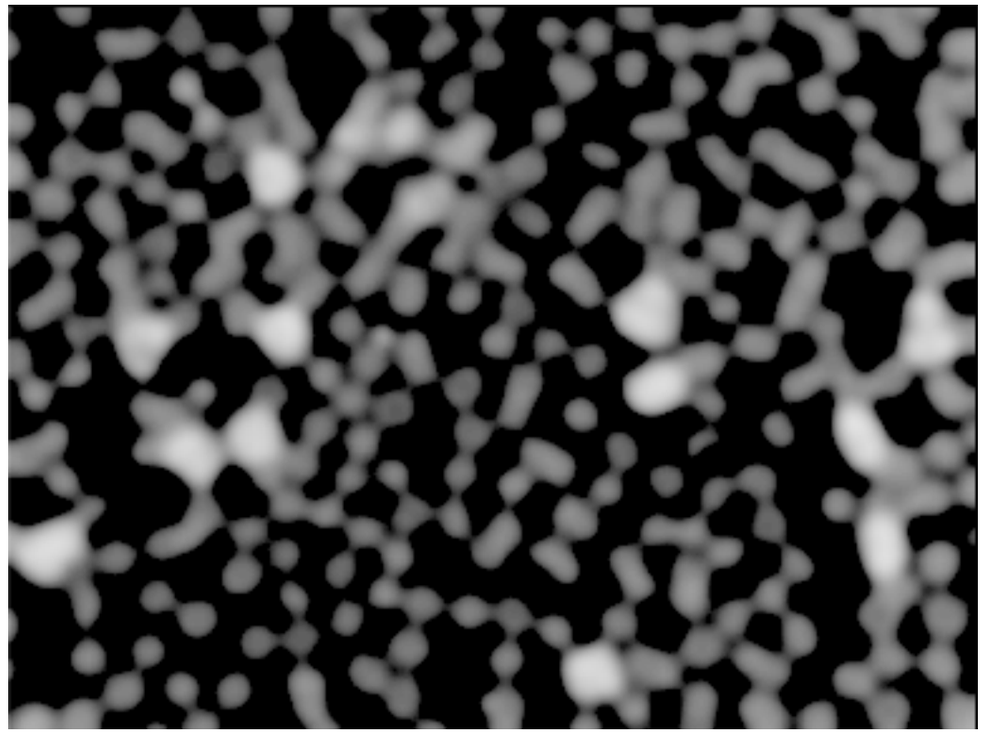
- Learning a linear classifier
  - Loss functions
  - Optimization
  - Overfitting
  - Train - Validation - Test Splits

Statistical Learning

Which is the best linear classifier?

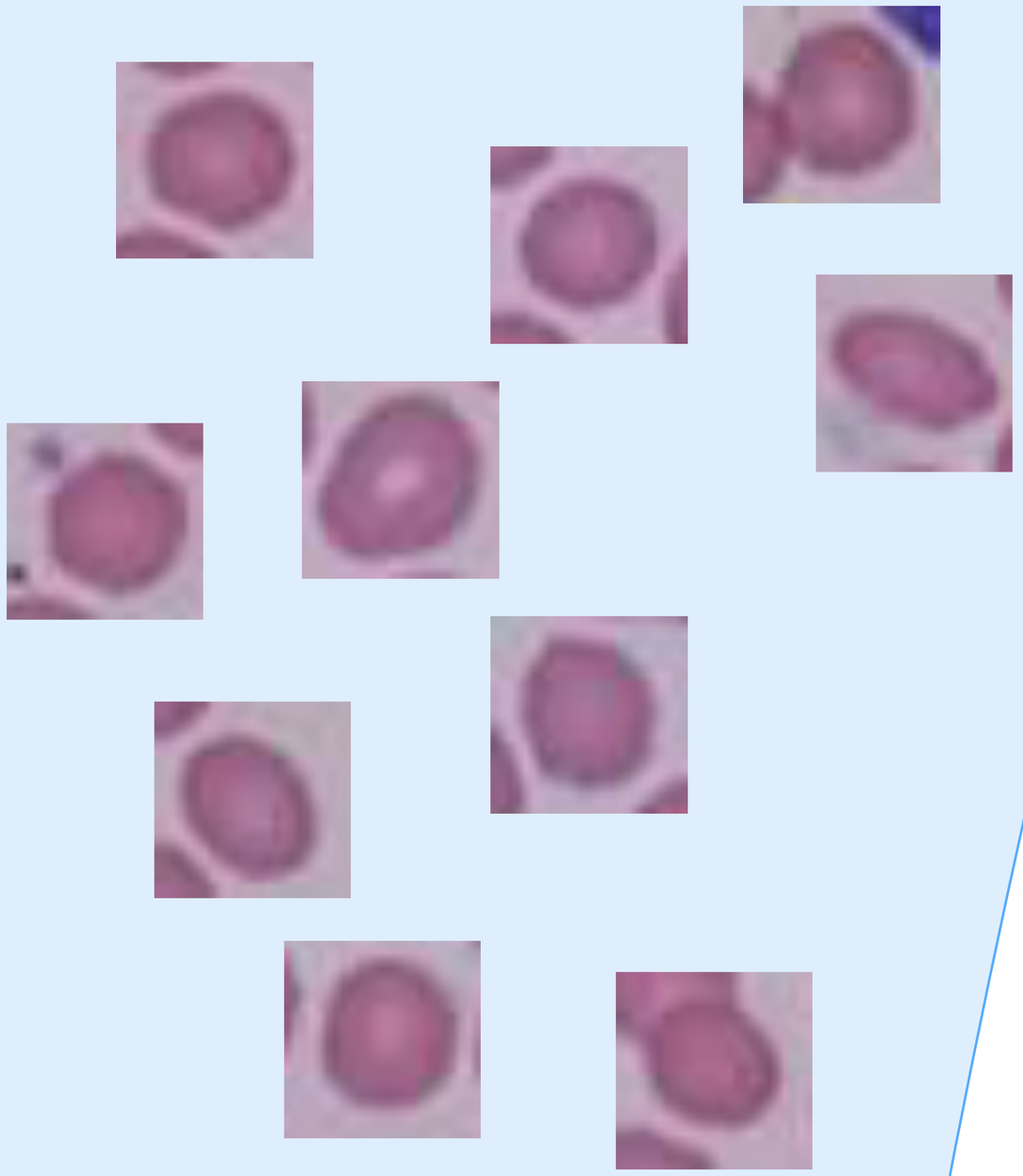


$\star w =$



…to use for sliding-window detection.

Manually Labelled Data



positive examples

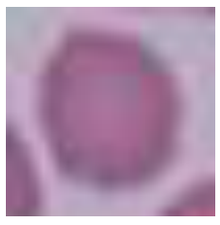
negative examples

Linear Classifier

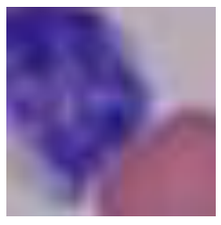
Find a w such that...

$\cdot w + w_0 > 0$

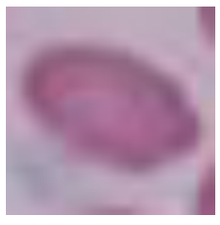
$\cdot w + w_0 < 0$



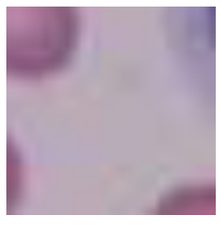
$\cdot w + w_0 > 0$



$\cdot w + w_0 < 0$

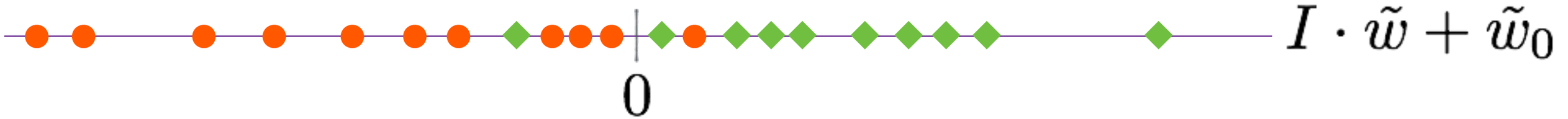


$\cdot w + w_0 > 0$

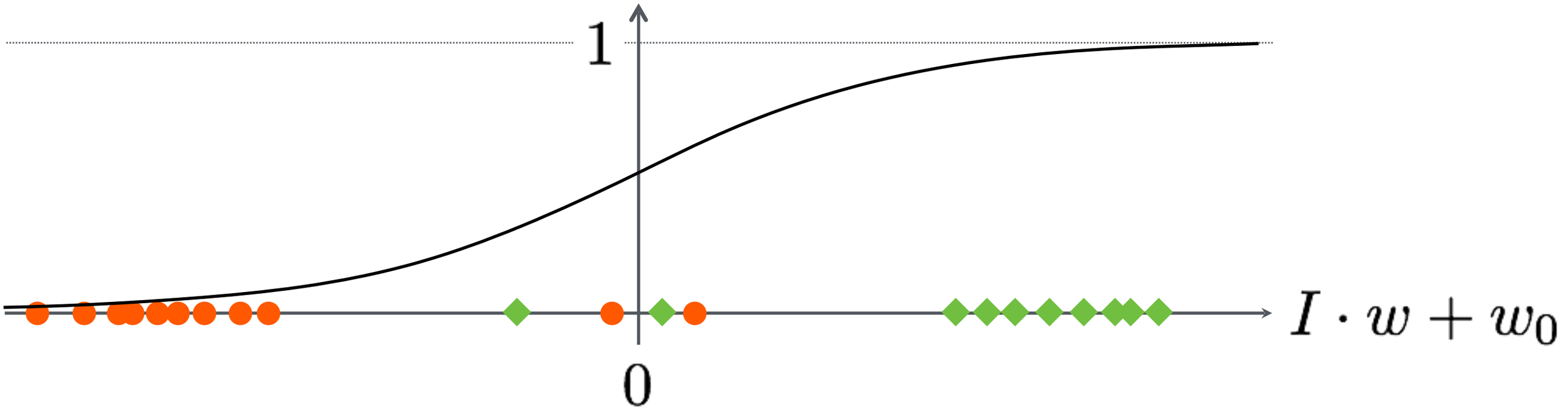


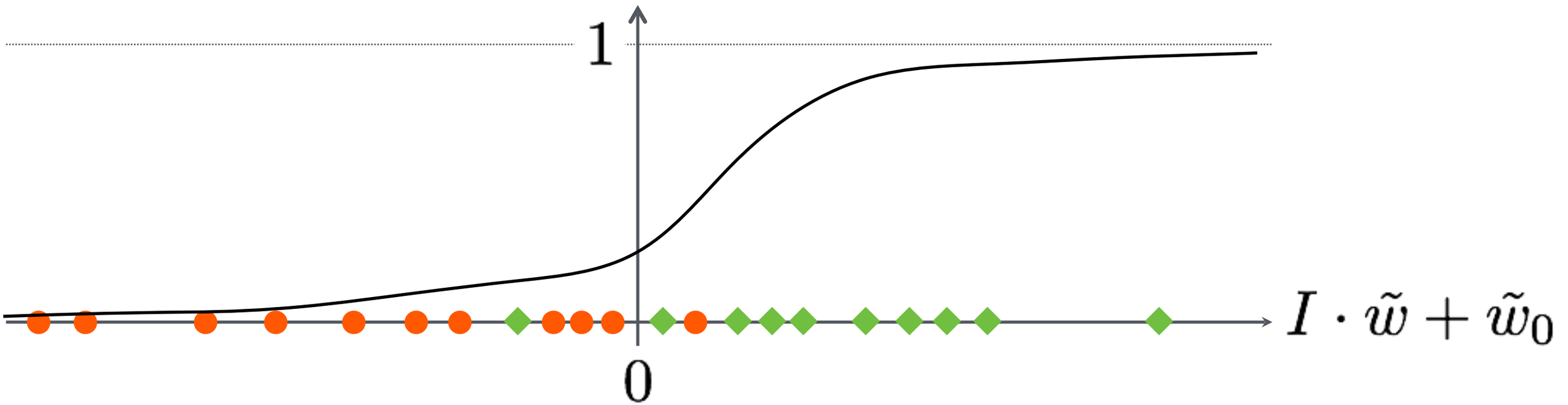
$\cdot w + w_0 < 0$

Produce a Probability

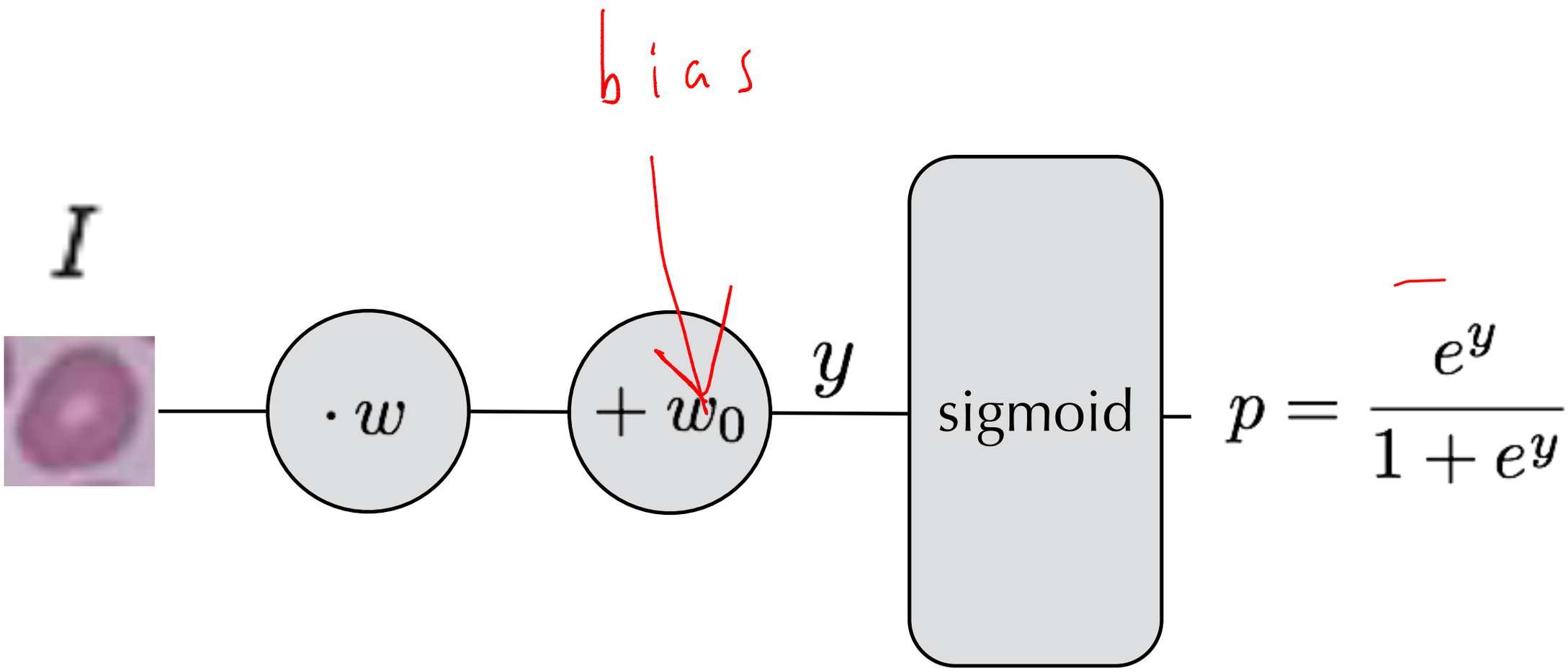


Produce a Probability



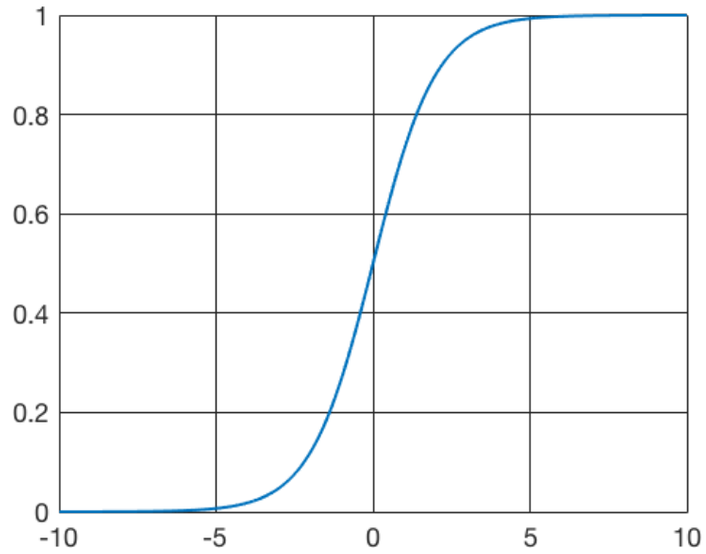


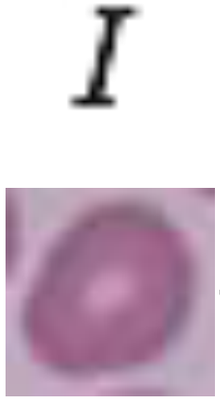
Sigmoid Function



Linear classifier (again)

Sigmoid Function





I

$\cdot w$

$+ w_0$

y

sigmoid

e^y

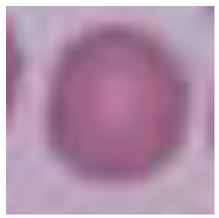
$$p = \frac{e^y}{1 + e^y}$$

Linear classifier (again)

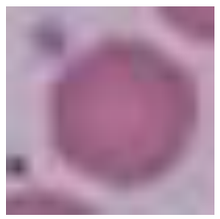
Performance Function

$\rightarrow p_1$

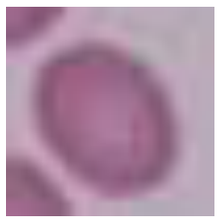
$\rightarrow p_4$



$\rightarrow p_5$

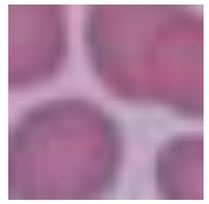


$\rightarrow p_6$



$\rightarrow p_8$

$\rightarrow p_2$



$\rightarrow p_3$

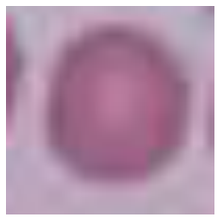
$\rightarrow p_7$

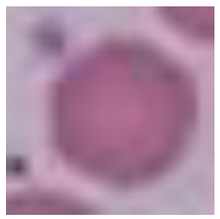
$\rightarrow p_9$

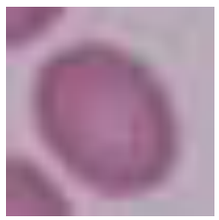
Performance Function: Likelihood


$\rightarrow p_1$

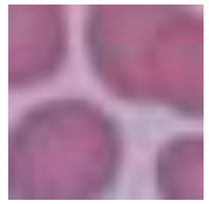

$\rightarrow p_4$


$\rightarrow p_5$


$\rightarrow p_6$


$\rightarrow p_8$


$\rightarrow p_2$


$\rightarrow p_3$


$\rightarrow p_7$


$\rightarrow p_9$

$$\prod p_i \qquad \bullet \qquad \prod (1 - p_i)$$

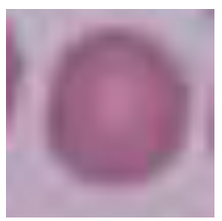
positive examples

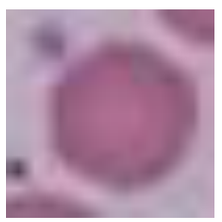
negative examples

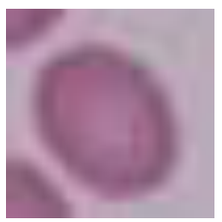
Performance Function: Log-Likelihood


$\rightarrow p_1$

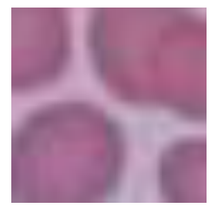

$\rightarrow p_4$

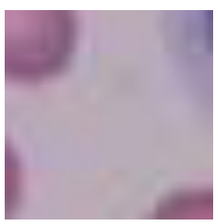

$\rightarrow p_5$


$\rightarrow p_6$


$\rightarrow p_8$


$\rightarrow p_2$


$\rightarrow p_3$


$\rightarrow p_7$


$\rightarrow p_9$

$$\sum \ln p_i + \sum \ln (1 - p_i)$$

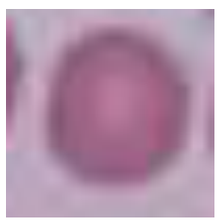
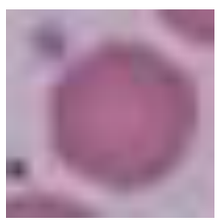
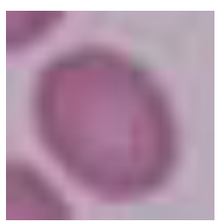
positive examples

negative examples

~~Performance~~ Function: Negative Log-Likelihood

Loss



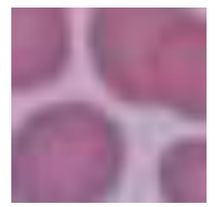
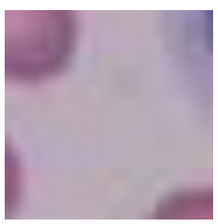
p_1

p_4

p_5

p_6

p_8


p_2

p_3

p_7

p_9

$$-\Big(\sum \ln p_i + \sum \ln(1-p_i)\Big)$$

positive examples

negative examples

Multiple Outputs

$$p_k = \frac{e^{y_k}}{\sum_{m=0}^{n} e^{y_m}}$$

softmax function

Optimization

Objective

Maximize likelihood :

$$\max_{\theta} \prod_{i \in S_1} p_i(\theta) \cdot \prod_{i \in S_0} (1 - p_i(\theta))$$

parameters

$\theta = \{w, w_0\}$

positive

negative

↑ partial loss

Objective

Maximize likelihood :

$$\max_{\theta} \prod_{i \in S_1} p_i(\theta) \cdot \prod_{i \in S_0} (1 - p_i(\theta))$$

parameters

$\Theta = \{w, w_0\}$

positive

negative

= minimize negative log-likelihood

↑ partial loss

Objective

Maximize likelihood :

$$\max_{\theta} \prod_{i \in S_1} p_i(\theta) \cdot \prod_{i \in S_0} (1 - p_i(\theta))$$

parameters

$\Theta = \{w, w_0\}$

positive

negative

= minimize negative log-likelihood

$$\min_{\theta} L(\theta) = -\sum_{i \in S_1} \ln(p_i(\theta)) - \sum_{i \in S_0} \ln((1 - p_i(\theta)))$$

↑ partial loss

Objective

Maximize likelihood :

$$\max_{\theta} \prod_{i \in S_1} p_i(\theta) \cdot \prod_{i \in S_0} (1 - p_i(\theta))$$

parameters

$\Theta = \{w, w_0\}$

positive

negative

= minimize negative log-likelihood

$$\min_{\theta} L(\theta) = -\sum_{i \in S_1} \ln(p_i(\theta)) - \sum_{i \in S_0} \ln((1 - p_i(\theta)))$$

$$= \sum_{i} L_i(\theta)$$

partial loss

Gradient Descent

Local instead of global optimization:

$$\nabla L(\theta) = \sum_i \nabla L_i(\theta)$$

learning rate

Gradient Descent

Local instead of global optimization:

$$\nabla L(\theta) = \sum_i \nabla L_i(\theta)$$

Update rule:

learning rate

$$\theta^{(k+1)} = \theta^{(k)} - \mu \nabla L(\theta) = \theta^{(k)} - \mu \sum_i \nabla L_i(\theta)$$

Stochastic Gradient Descent

Use only subset of data for update

$$\theta^{(k+1)} = \theta^{(k)} - \mu \sum_i \nabla L_i(\theta) \approx \theta^{(k)} - \mu \nabla L_i(\theta)$$

$$\simeq \theta^{(k)} - \mu \frac{1}{m} \sum_j \nabla L_j(\theta)$$

Stochastic Gradient Descent

Use only subset of data for update

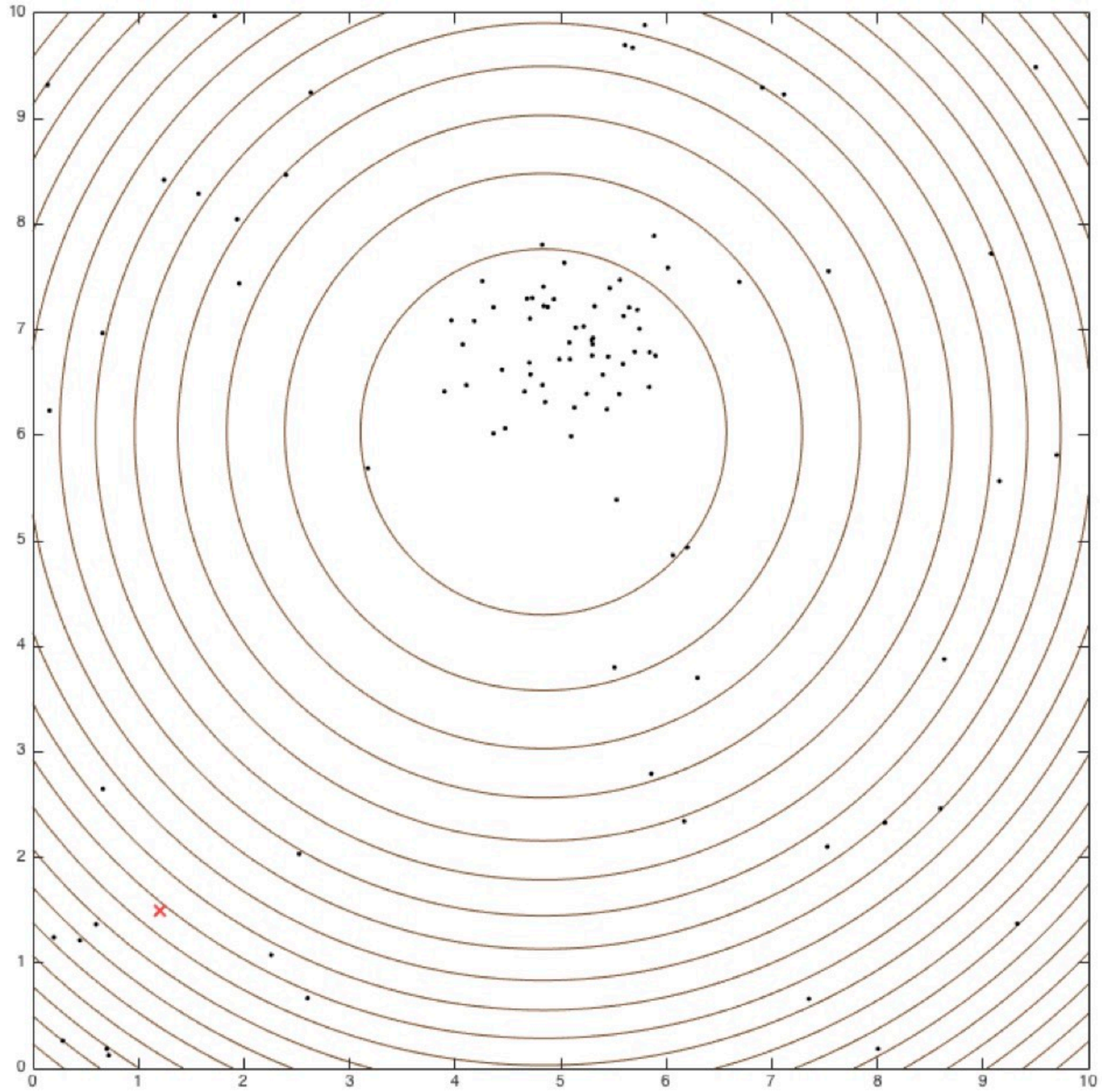
$$\theta^{(k+1)} = \theta^{(k)} - \mu \sum_i \nabla L_i(\theta) \approx \theta^{(k)} - \mu \nabla L_i(\theta)$$

$$\simeq \theta^{(k)} - \mu \frac{1}{m} \sum_j \nabla L_j(\theta)$$

Better: Use a few data points (**mini batch**)

Simple 2D Example

Compute 2D location θ from data points (Gaussian noise)



Simple 2D Example

Compute 2D location θ from data points (Gaussian noise)

Residual per measurement: $r_i(\theta) = x_i - \theta$

$$\ln\left(e^{\frac{(x_i-\theta)^2}{2\sigma^2}}\right) = \frac{(x_i-\theta)^2}{2\sigma^2}$$

Simple 2D Example

Compute 2D location θ from data points (Gaussian noise)

Residual per measurement: $r_i(\theta) = x_i - \theta$

Likelihood: $\prod_i e^{\frac{r_i(\theta)^2}{2\sigma^2}} = \prod_i e^{\frac{(x_i-\theta)^2}{2\sigma^2}}$

$$\ln\left(e^{\frac{(x_i-\theta)^2}{2\sigma^2}}\right) = \frac{(x_i-\theta)^2}{2\sigma^2}$$

Simple 2D Example

Compute 2D location θ from data points (Gaussian noise)

Residual per measurement: $r_i(\theta) = x_i - \theta$

Likelihood: $\prod_i e^{\frac{r_i(\theta)^2}{2\sigma^2}} = \prod_i e^{\frac{(x_i-\theta)^2}{2\sigma^2}}$

$$\ln\left(e^{\frac{(x_i-\theta)^2}{2\sigma^2}}\right) = \frac{(x_i-\theta)^2}{2\sigma^2}$$

Negative Log-Likelihood: $L(\theta) = \sum_i (x_i - \theta)^2 = \sum_i L_i(\theta)$

Simple 2D Example

Objective: $$\min_{\theta} L(\theta) = \sum_i (x_i - \theta)^2 = \sum_i L_i(\theta)$$

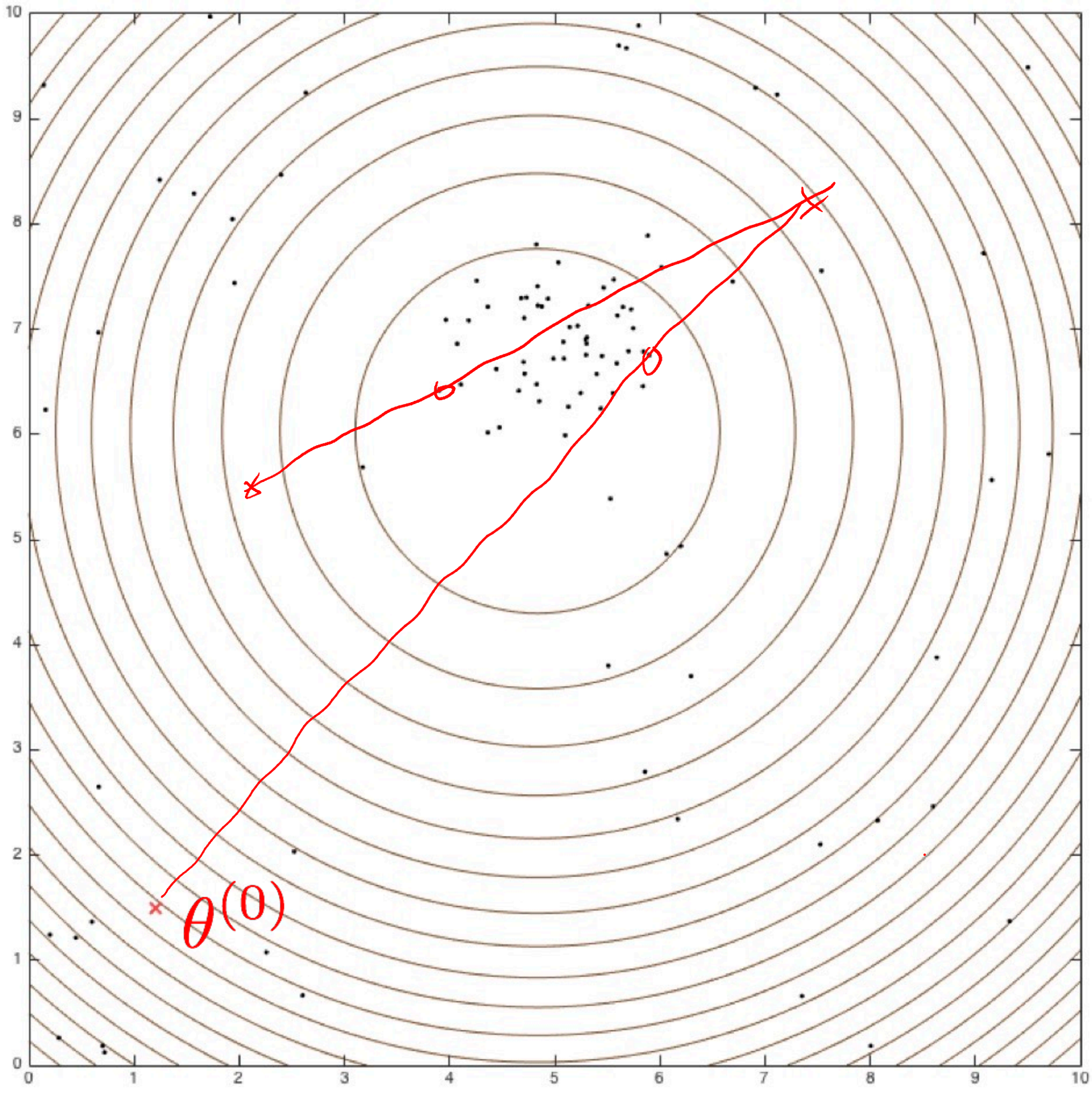
Simple 2D Example

Objective: $$\min_{\theta} L(\theta) = \sum_i (x_i - \theta)^2 = \sum_i L_i(\theta)$$

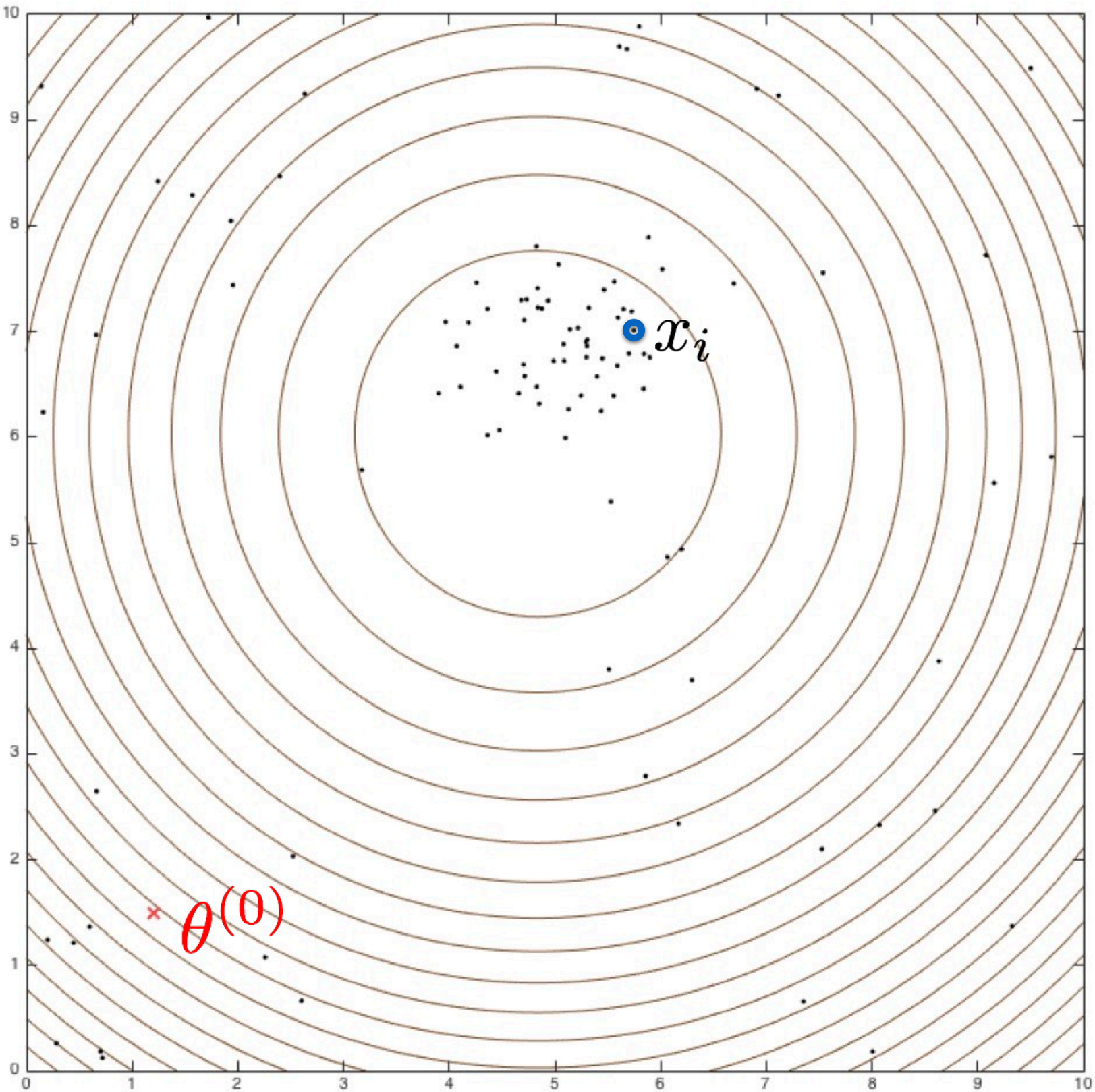
Gradient:

$$\nabla L(\theta) = -\sum_i 2 \cdot (x_i - \theta)$$

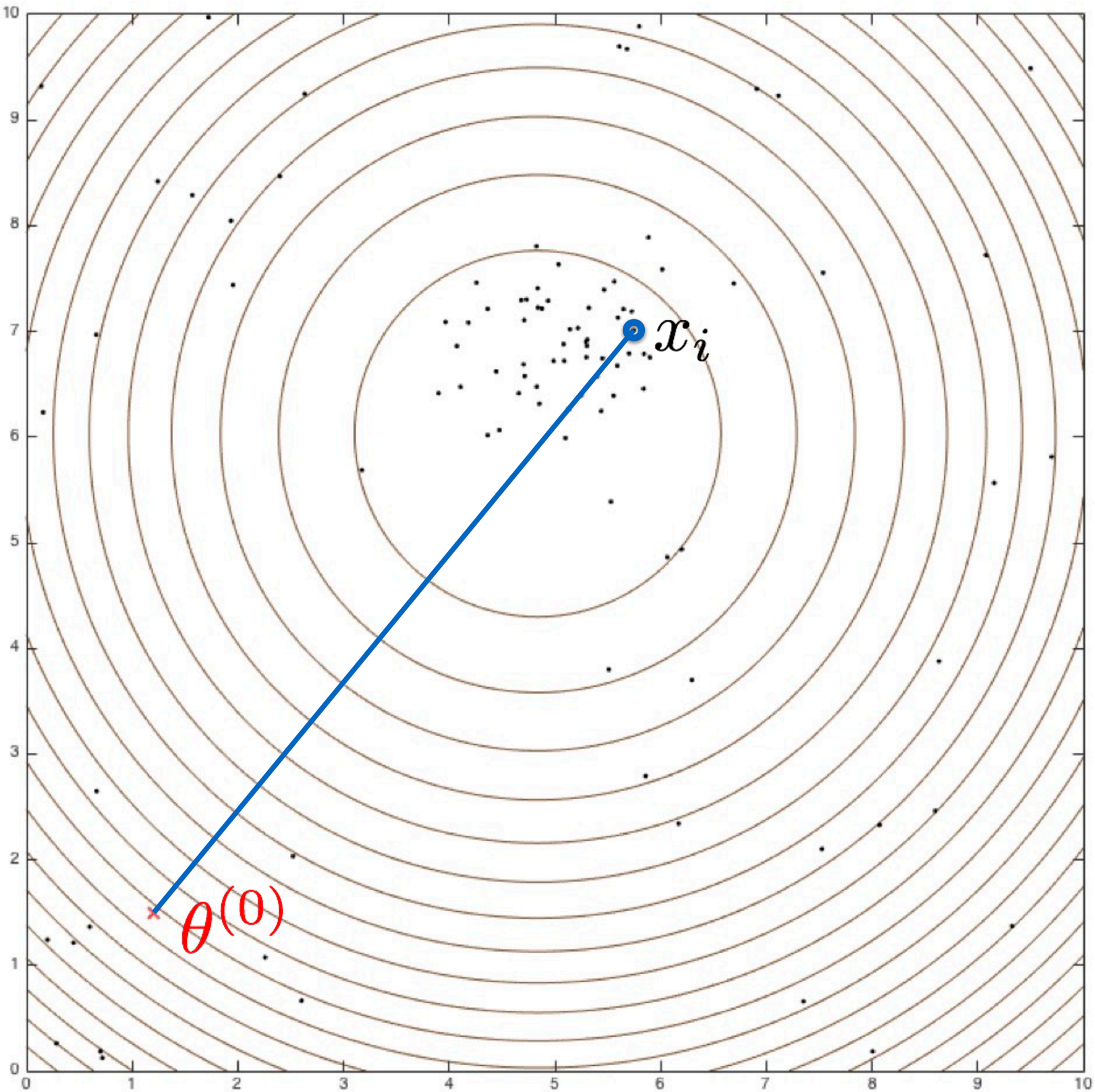
Stochastic Gradient Descent



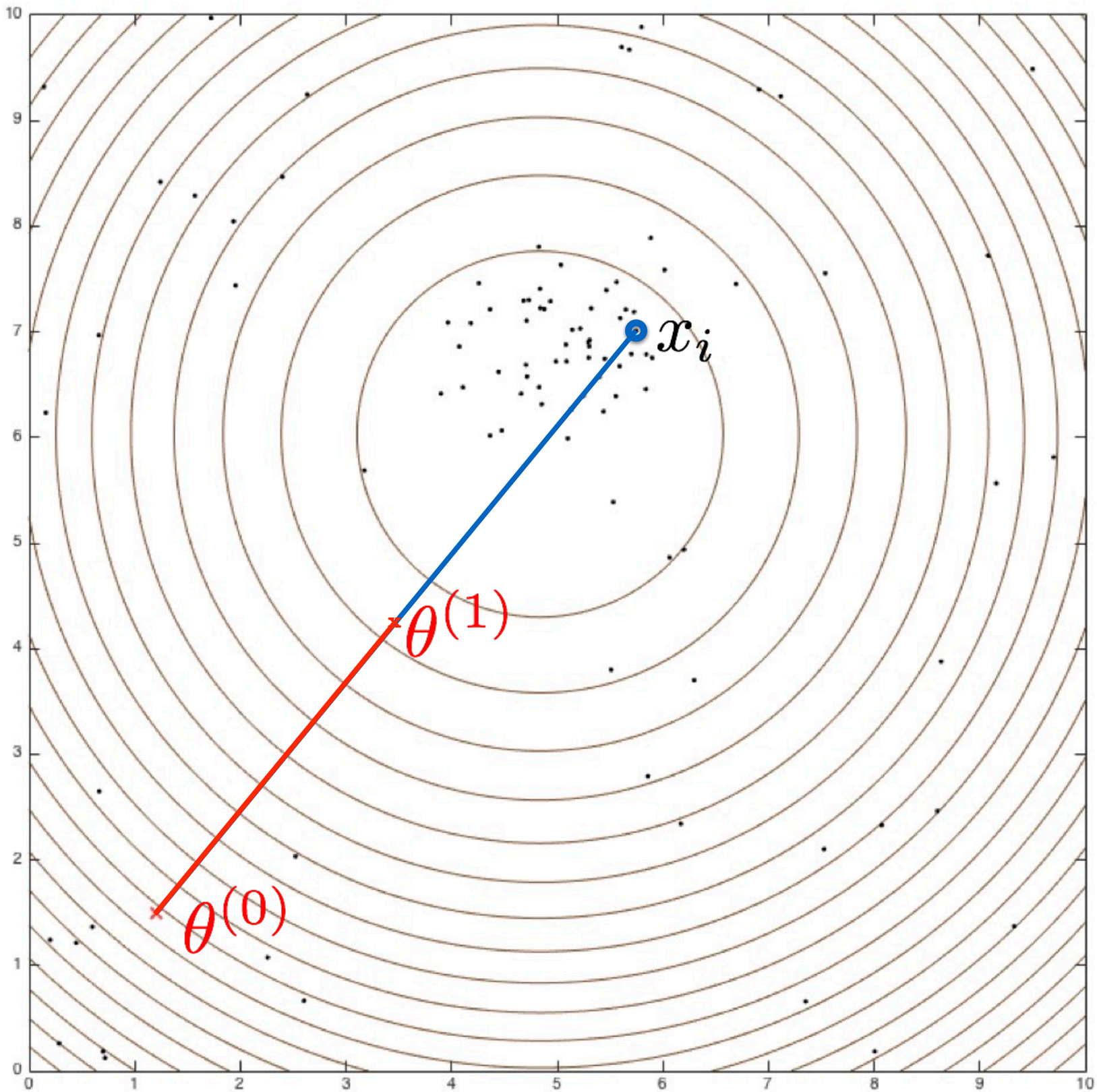
Stochastic Gradient Descent



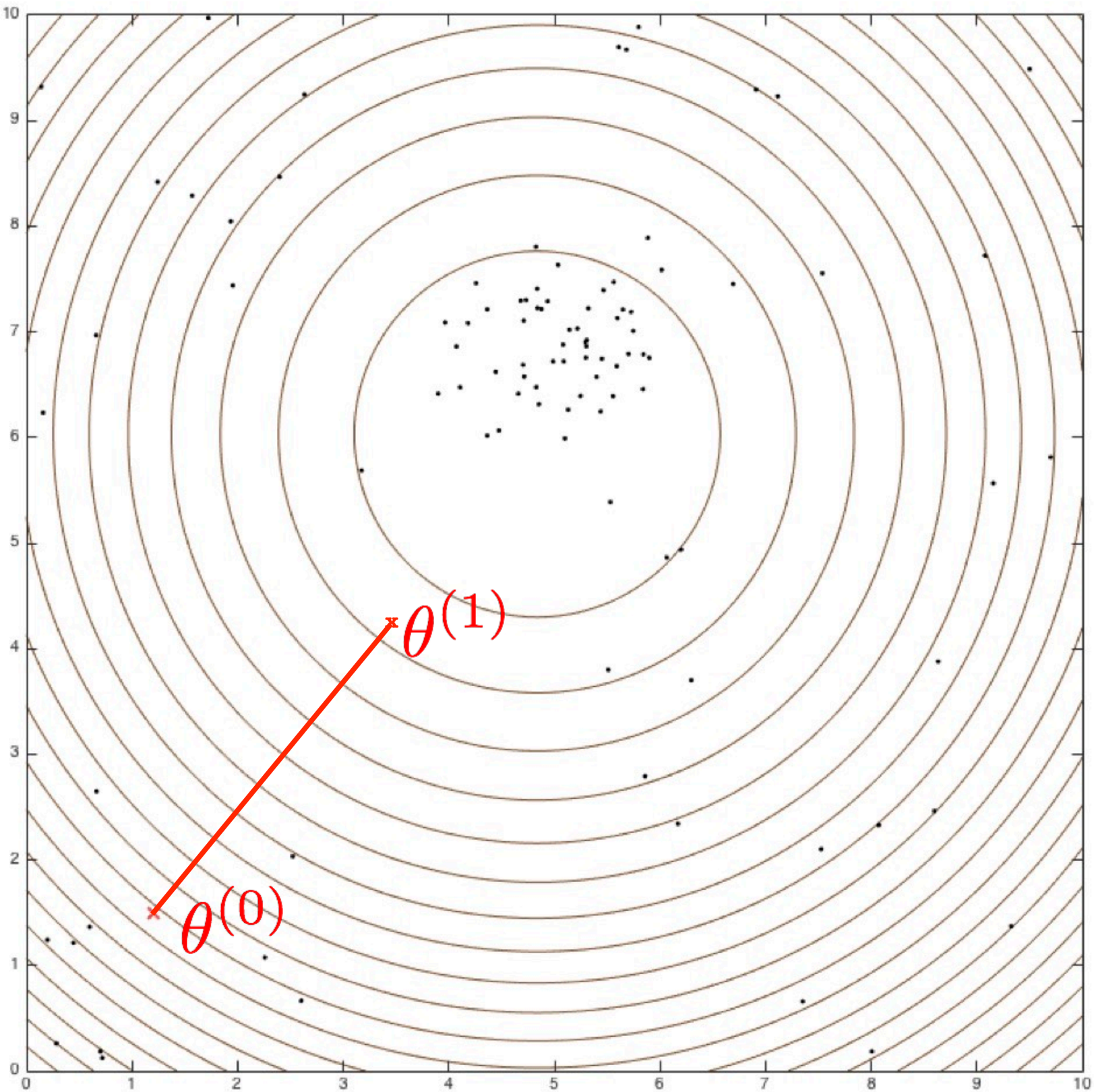
Stochastic Gradient Descent



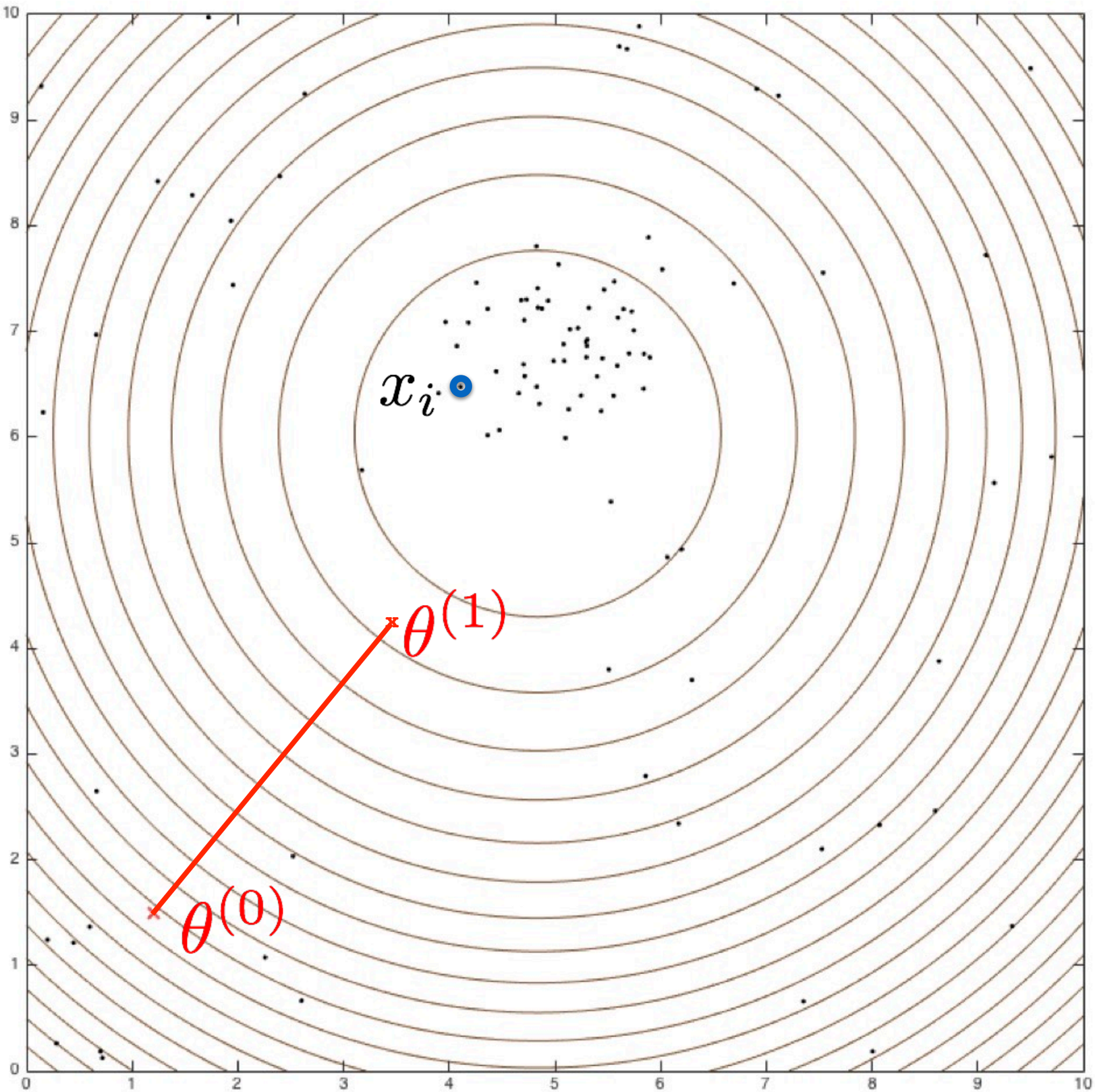
Stochastic Gradient Descent



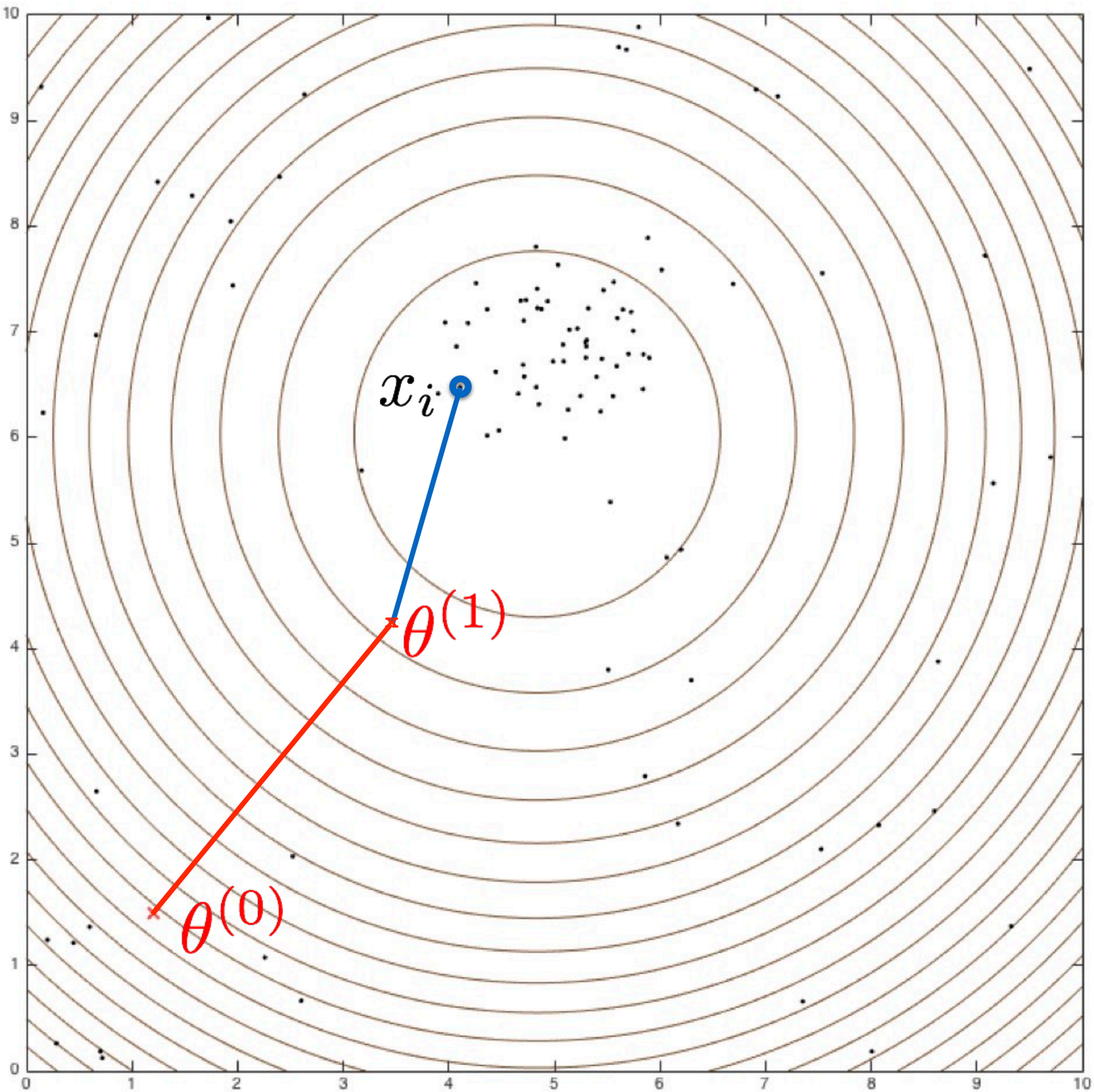
Stochastic Gradient Descent



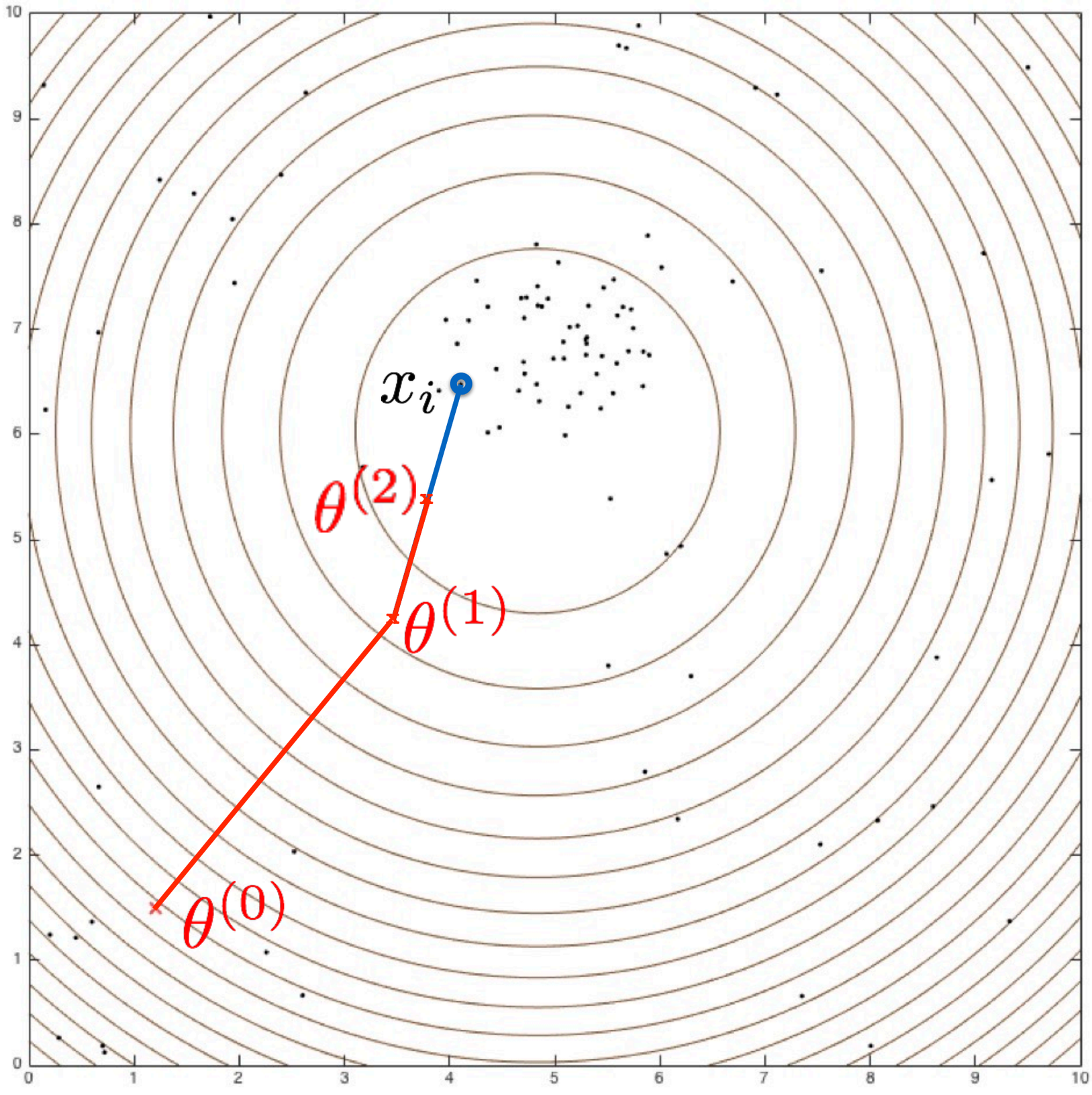
Stochastic Gradient Descent



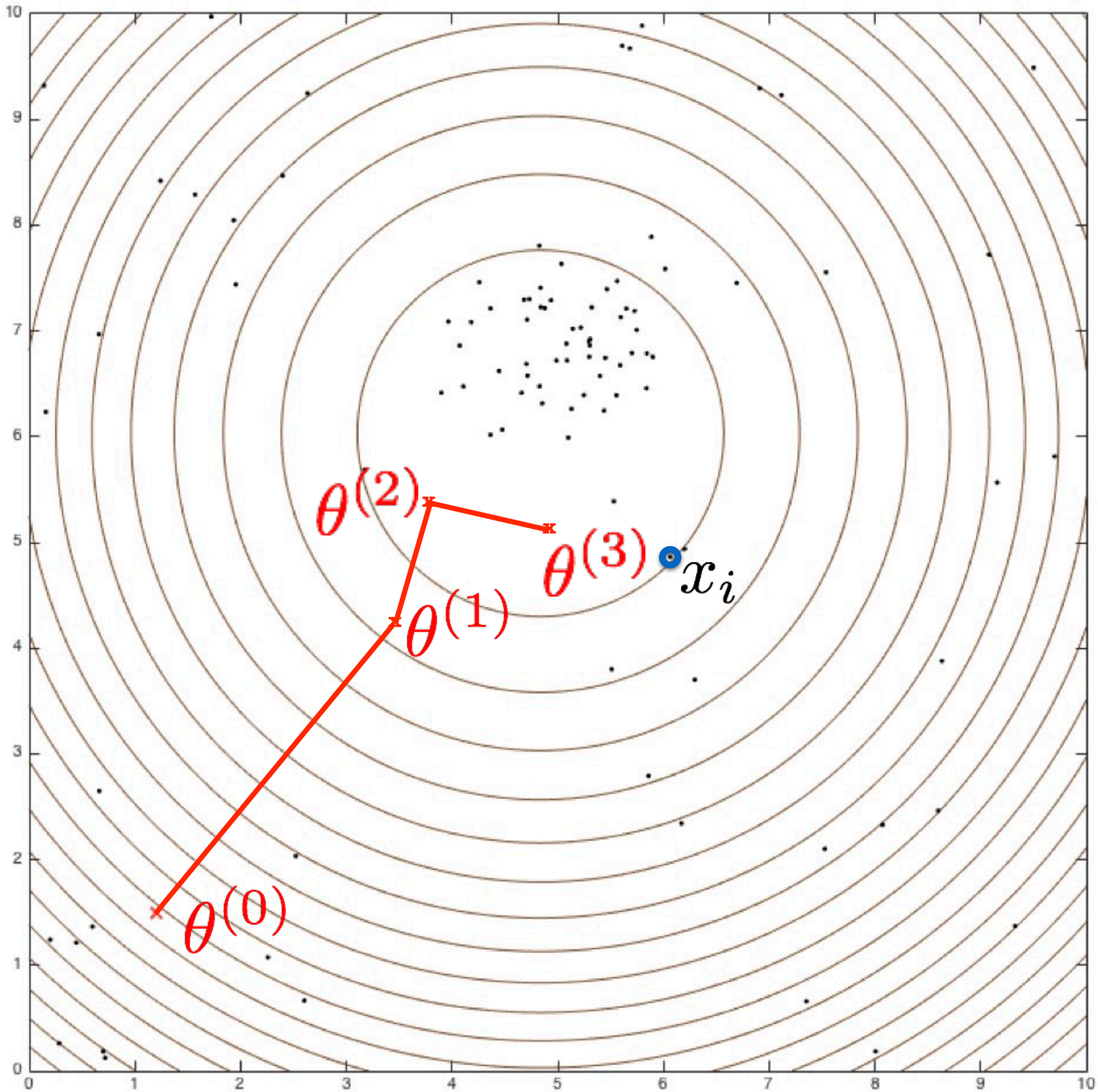
Stochastic Gradient Descent



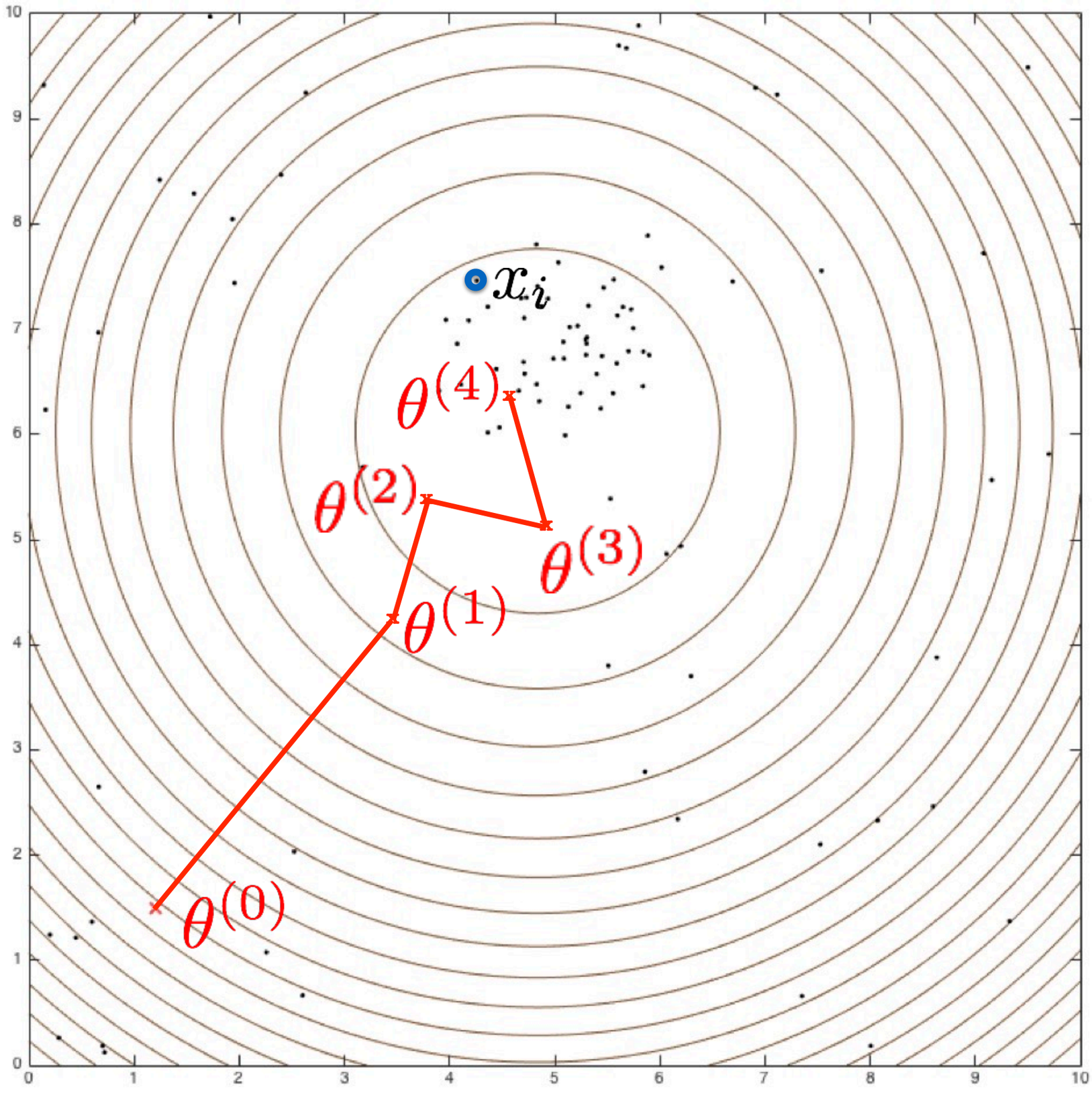
Stochastic Gradient Descent



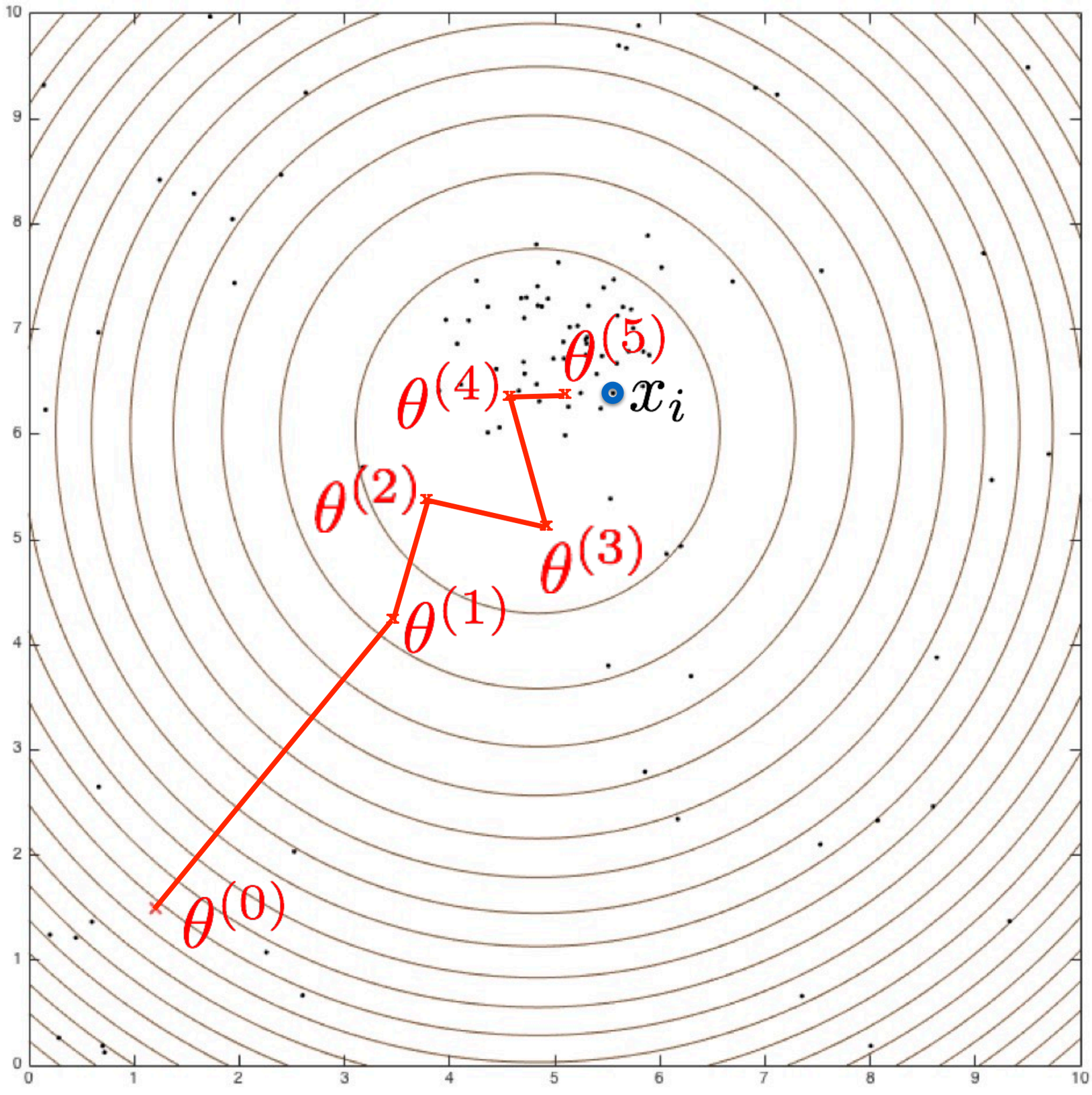
Stochastic Gradient Descent



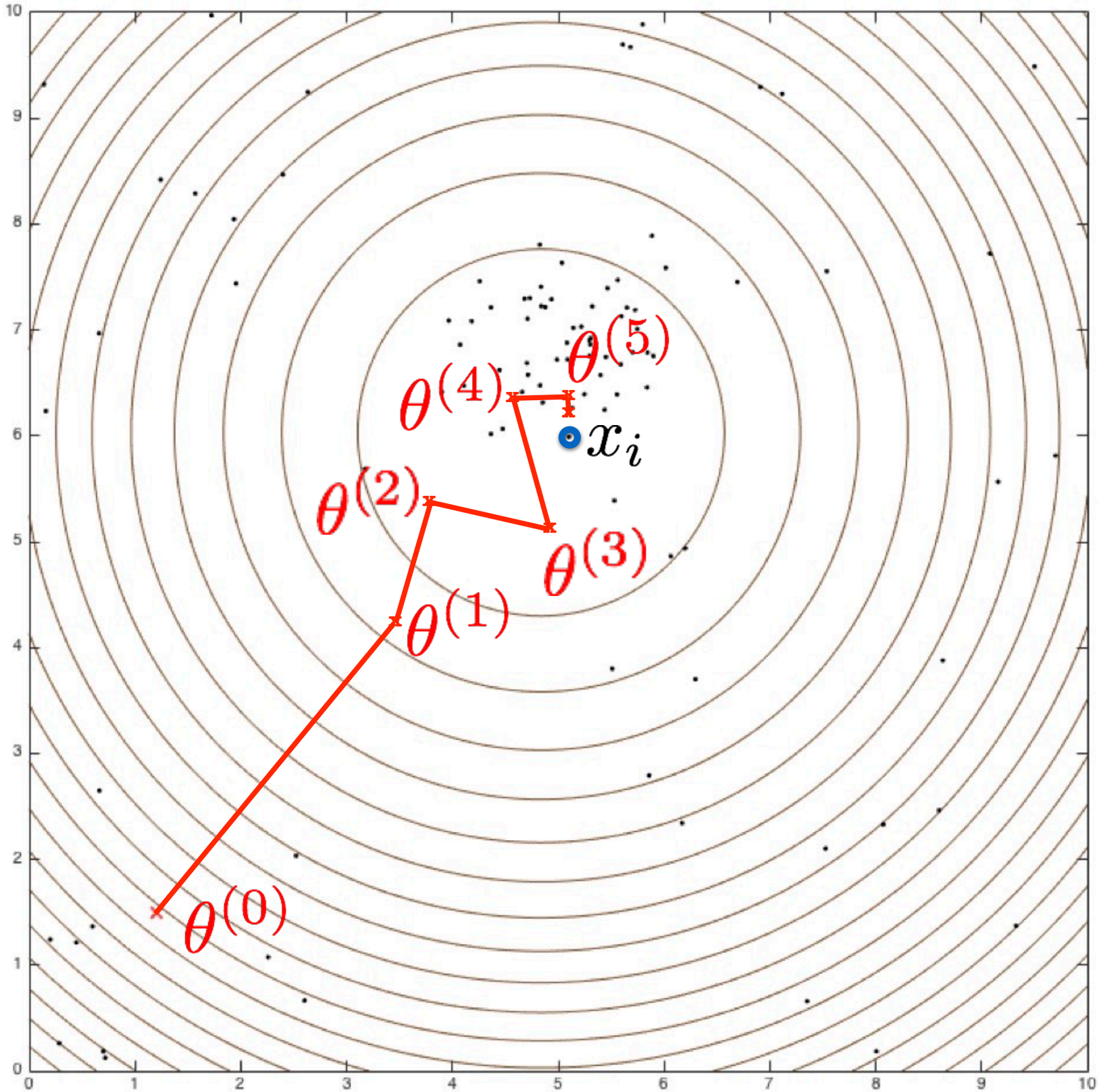
Stochastic Gradient Descent



Stochastic Gradient Descent



Stochastic Gradient Descent



Stochastic Gradient Descent with Momentum

Stochastic gradient descent:

$$\theta^{(k+1)} = \theta^{(k)} - \mu \nabla L_i(\theta)$$

momentum

might not be used

Stochastic Gradient Descent with Momentum

Stochastic gradient descent:

$$\theta^{(k+1)} = \theta^{(k)} - \mu \nabla L_i(\theta)$$

Stochastic gradient descent with momentum:

$$\theta^{(k+1)} = \theta^{(k)} - \mu g^{(k)}$$

momentum

might not be used

Stochastic Gradient Descent with Momentum

Stochastic gradient descent:

$$\theta^{(k+1)} = \theta^{(k)} - \mu \nabla L_i(\theta)$$

Stochastic gradient descent with momentum:

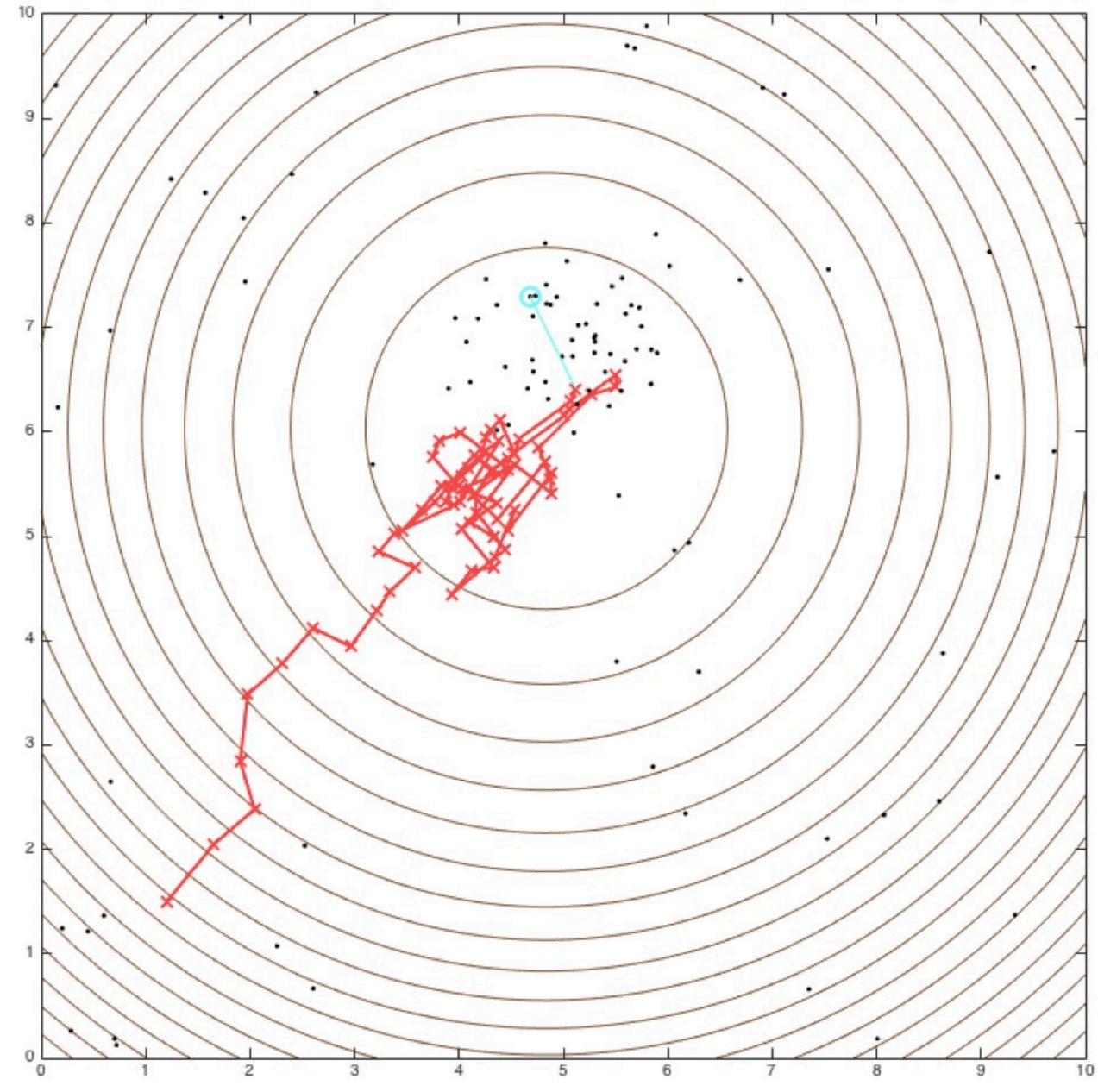
$$\theta^{(k+1)} = \theta^{(k)} - \mu g^{(k)}$$

$$g^{(k)} = \gamma g^{(k-1)} + (1-\gamma) \nabla L_i(\theta^{(k)})$$

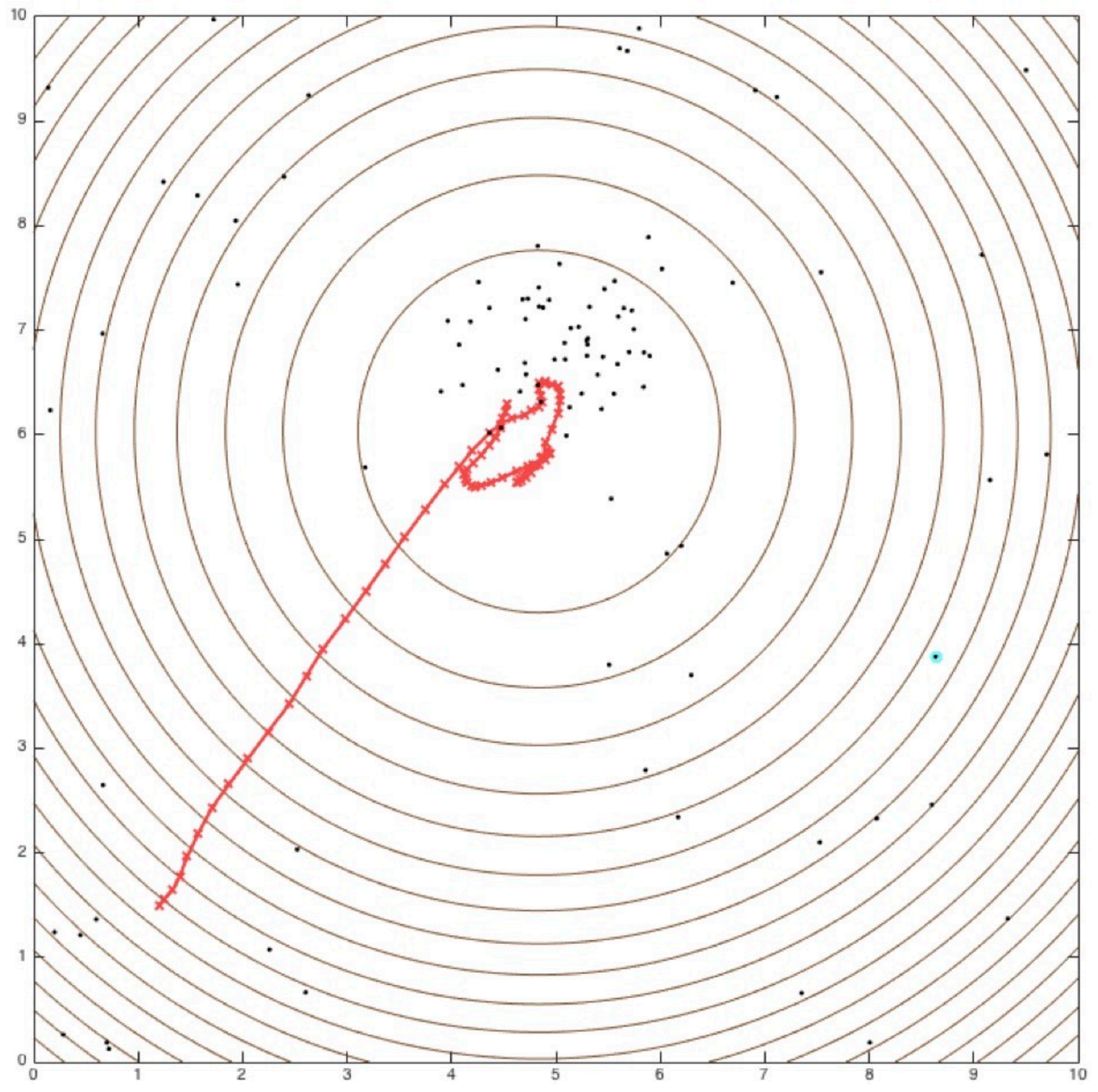
momentum

might not be used

Stochastic Gradient Descent with Momentum



without momentum



with momentum

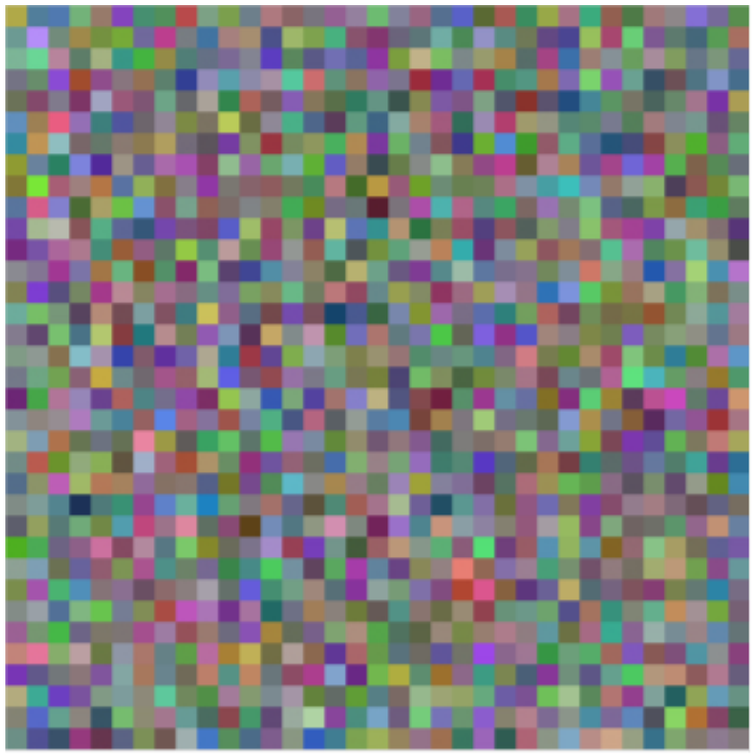
More on Optimization

See Chapter 8 of [Goodfellow, Bengio, Courville, Deep Learning, MIT Press 2016]

(will not be part of the exam)

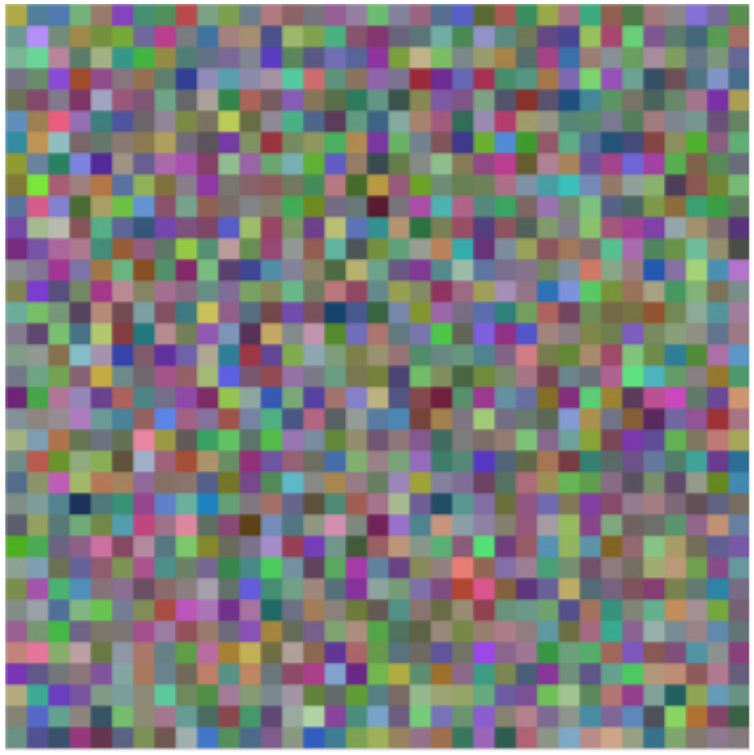
Back to learning a linear classifier …

Linear Cell Classifier

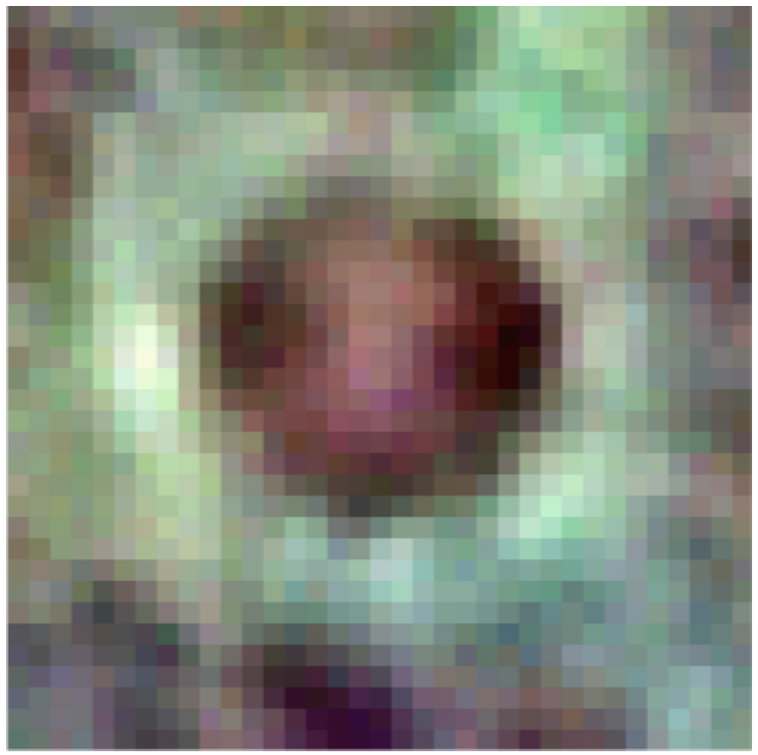


Initialization

Linear Cell Classifier

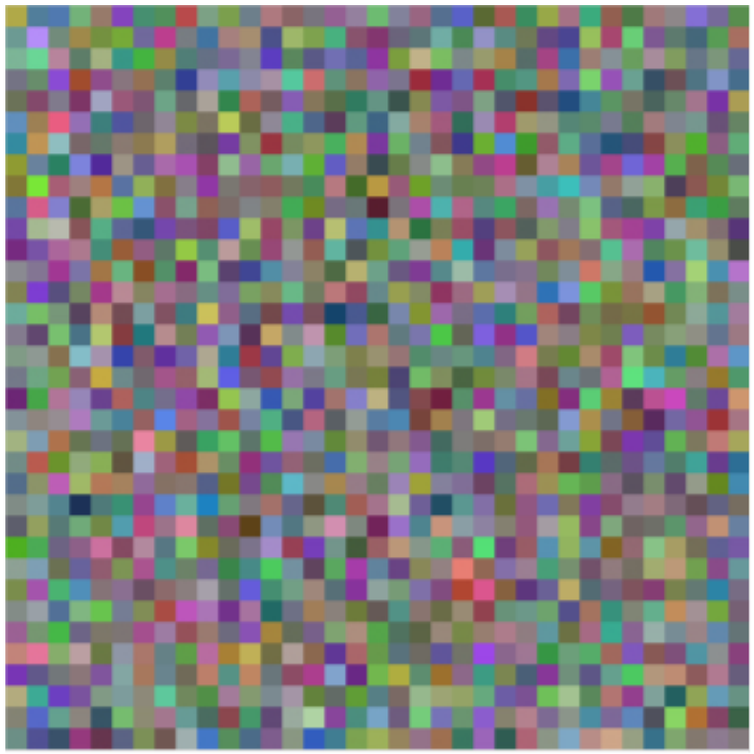


Initialization

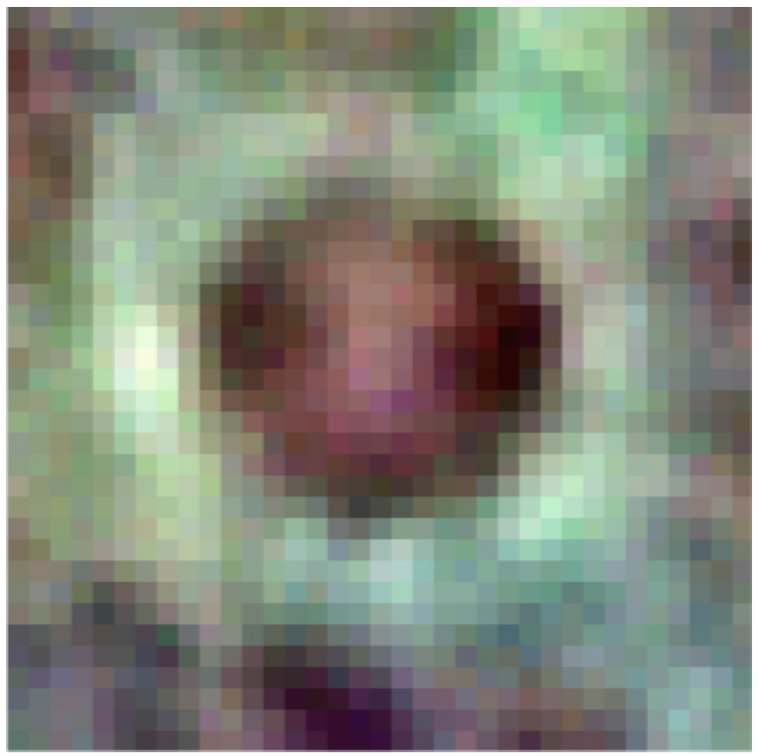


50 iterations

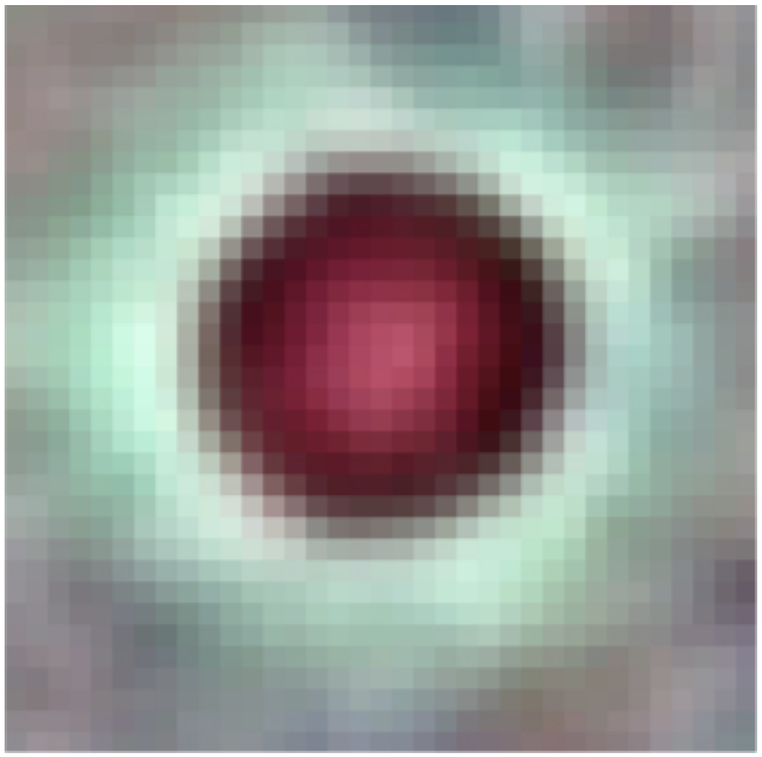
Linear Cell Classifier



Initialization

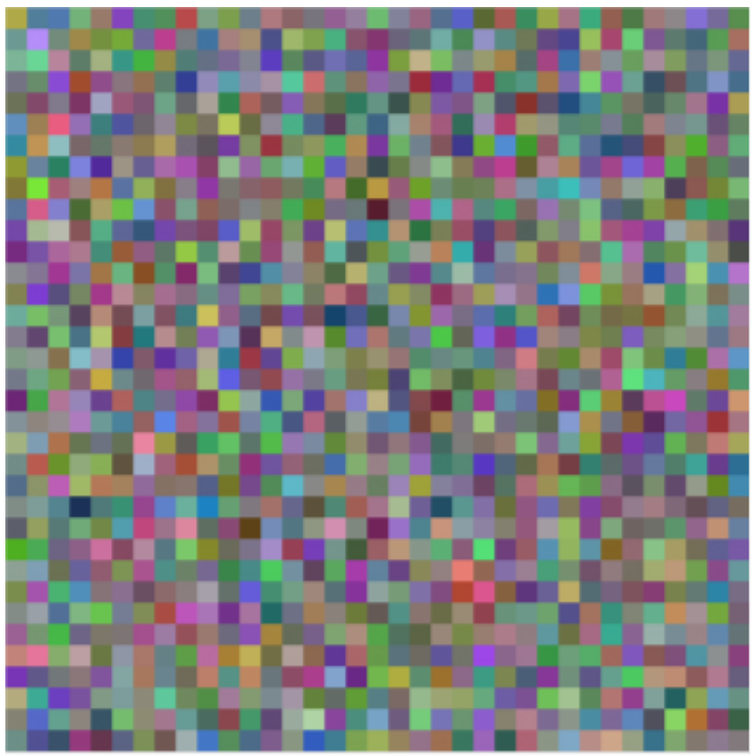


50 iterations

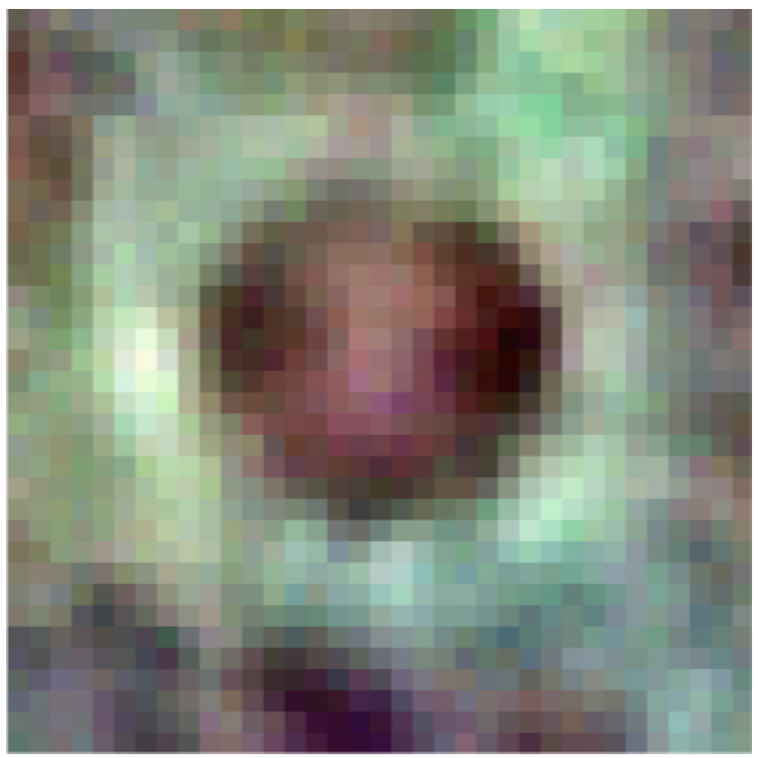


1000 iterations

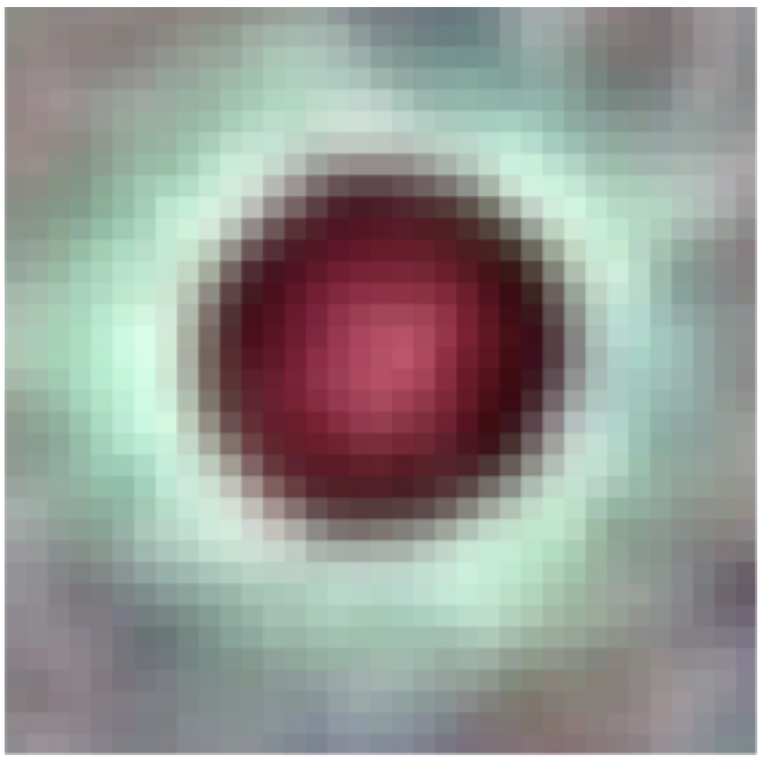
Linear Cell Classifier



Initialization

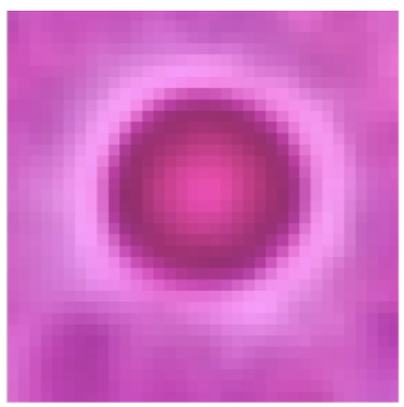


50 iterations

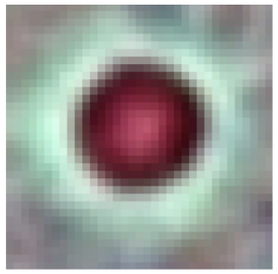


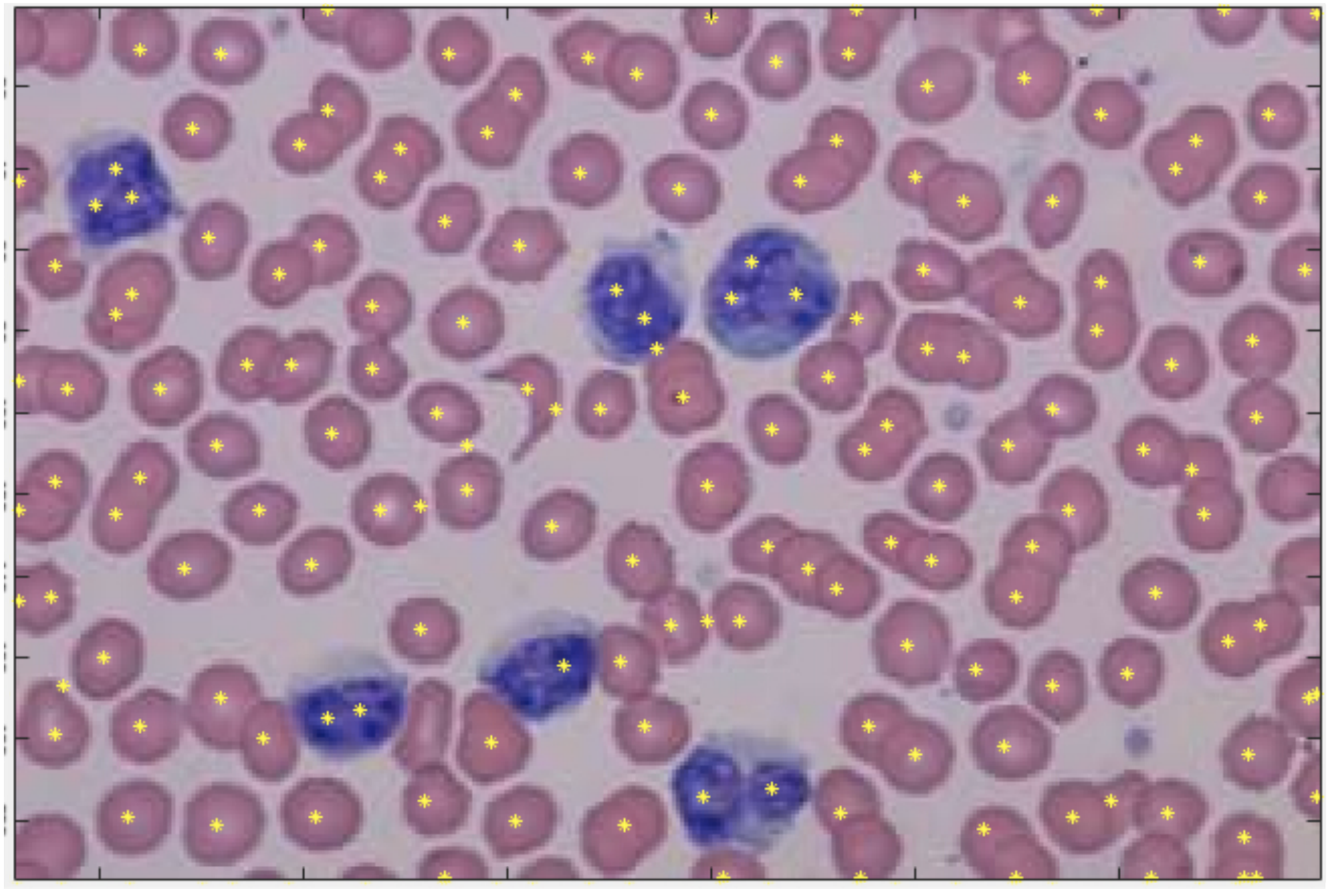
1000 iterations

Mean cell
(normalized)

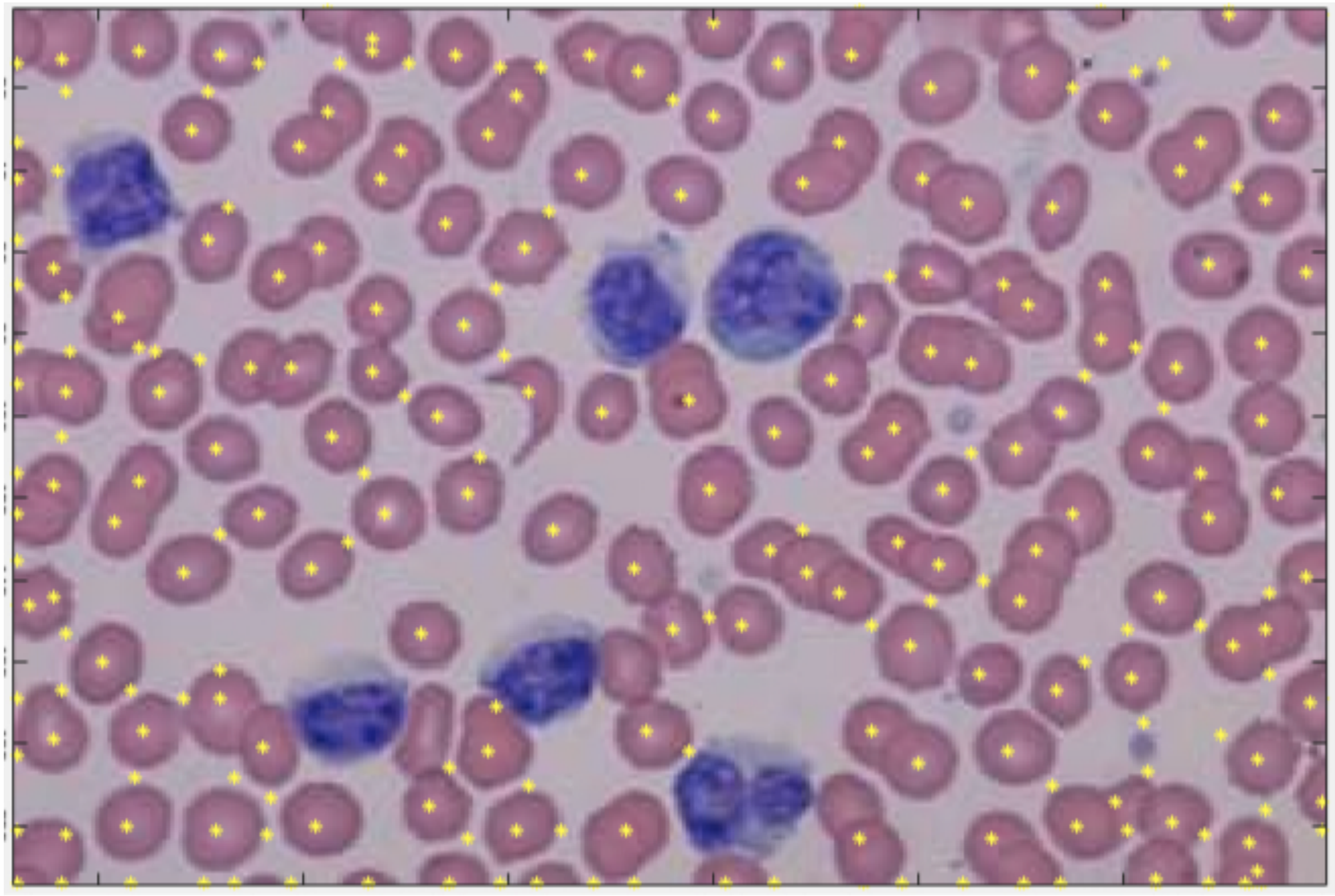


Importance of Sampling





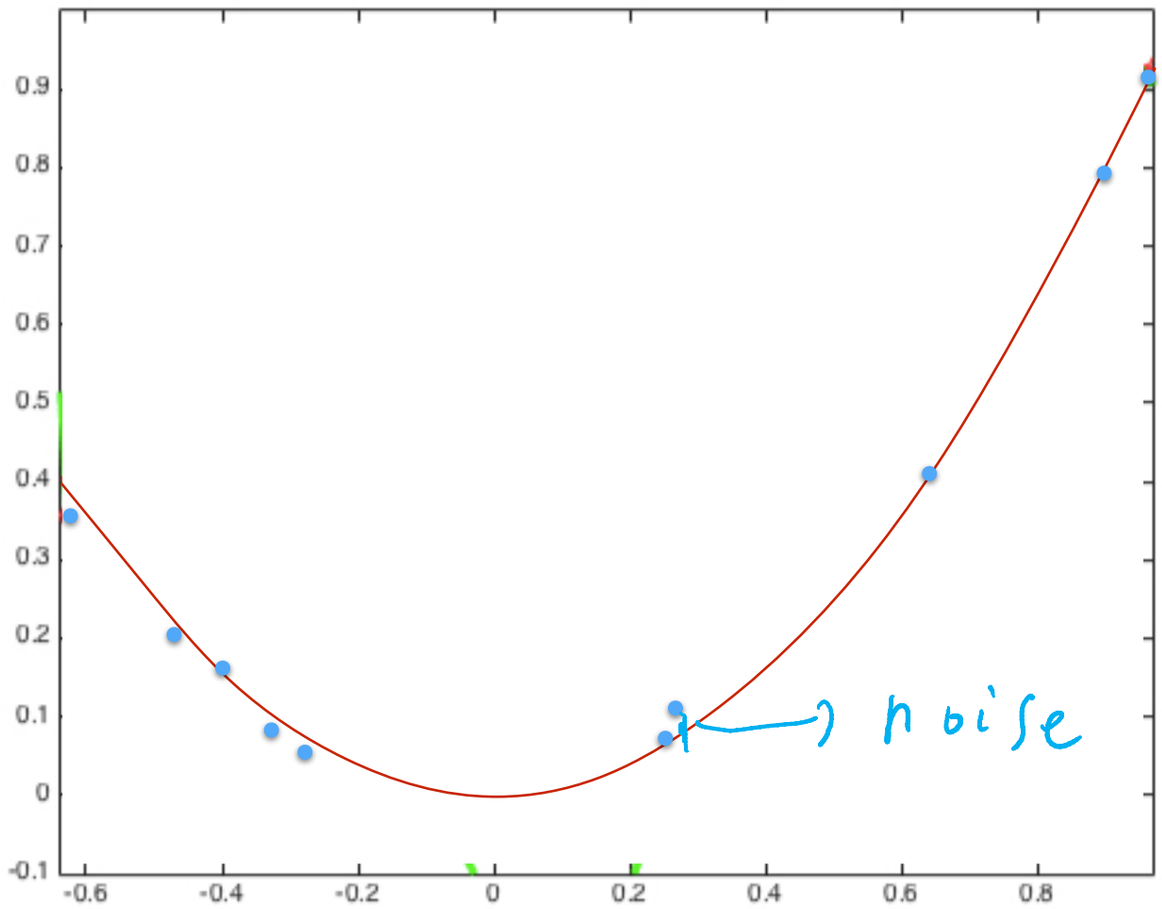
Importance of Sampling



Overfitting

Overfitting

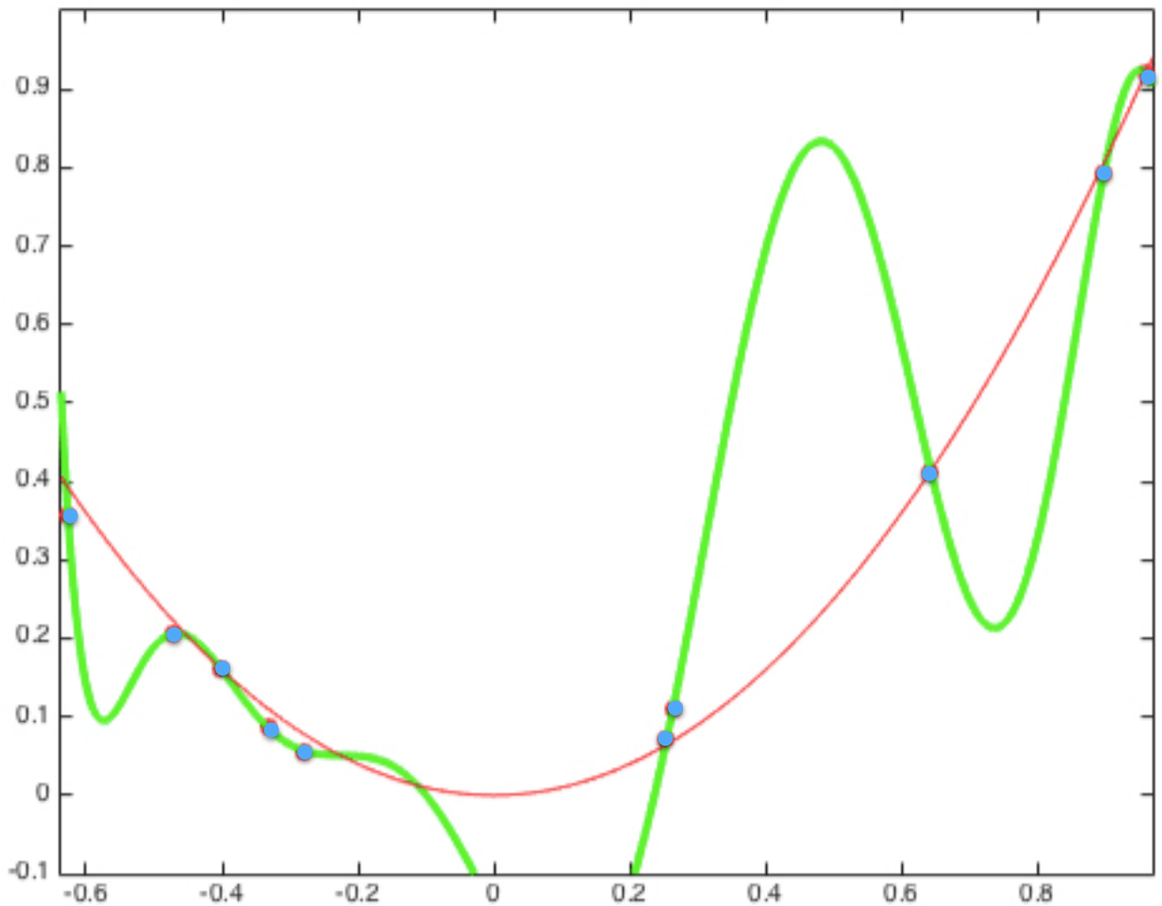
10 data points



$$y = x^2$$

Overfitting

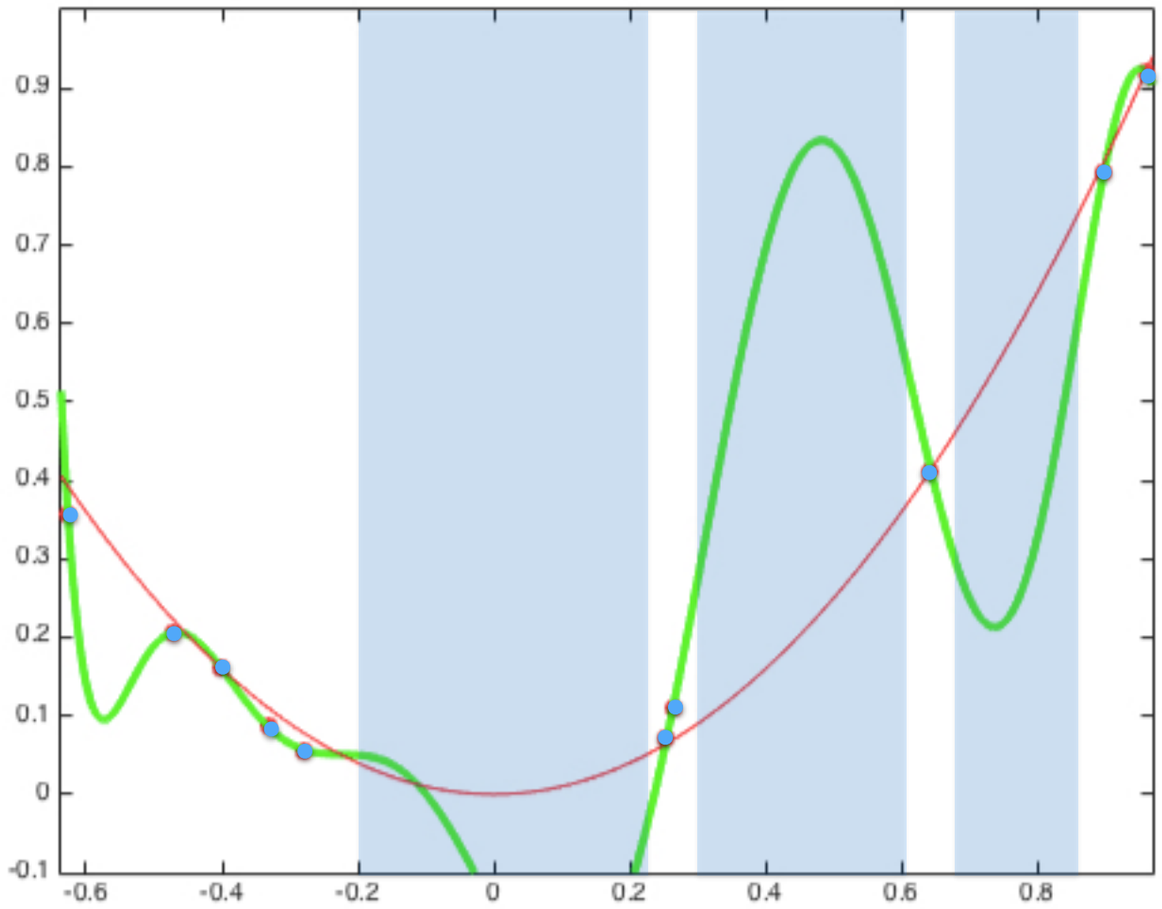
10 data points - Fitting 10^{th} degree polynomial



$$y = x^2$$

Overfitting

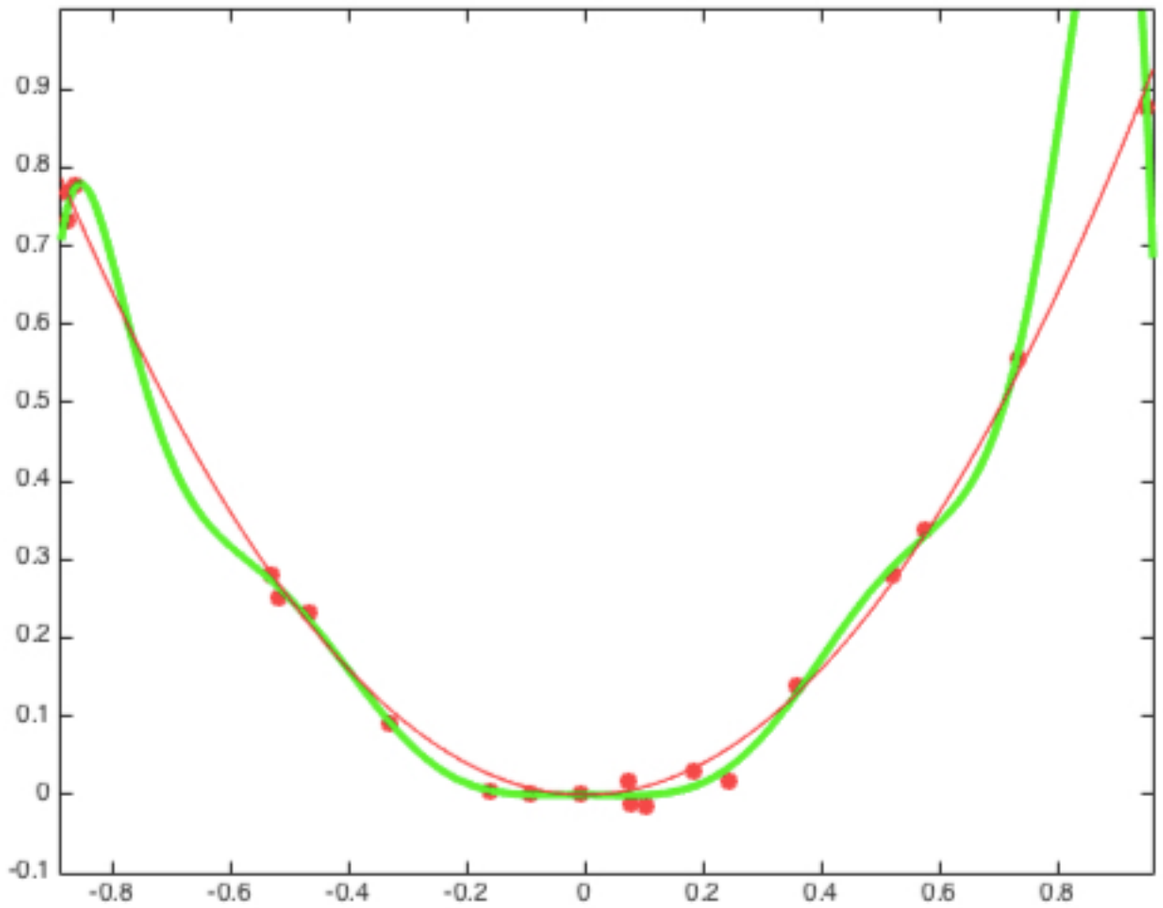
10 data points - Fitting 10th degree polynomial



$$y = x^2$$

Overfitting

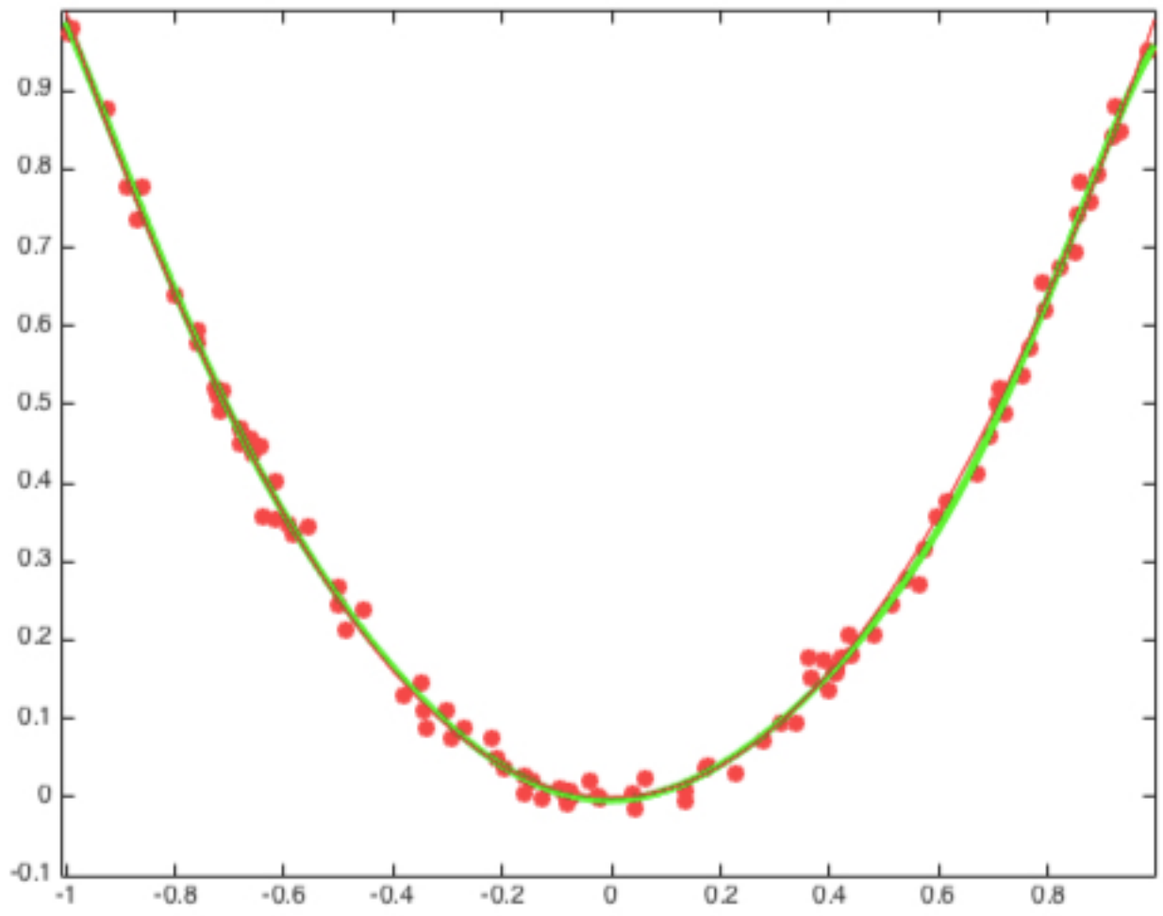
20 data points - Fitting 10th degree polynomial



$$y = x^2$$

Overfitting

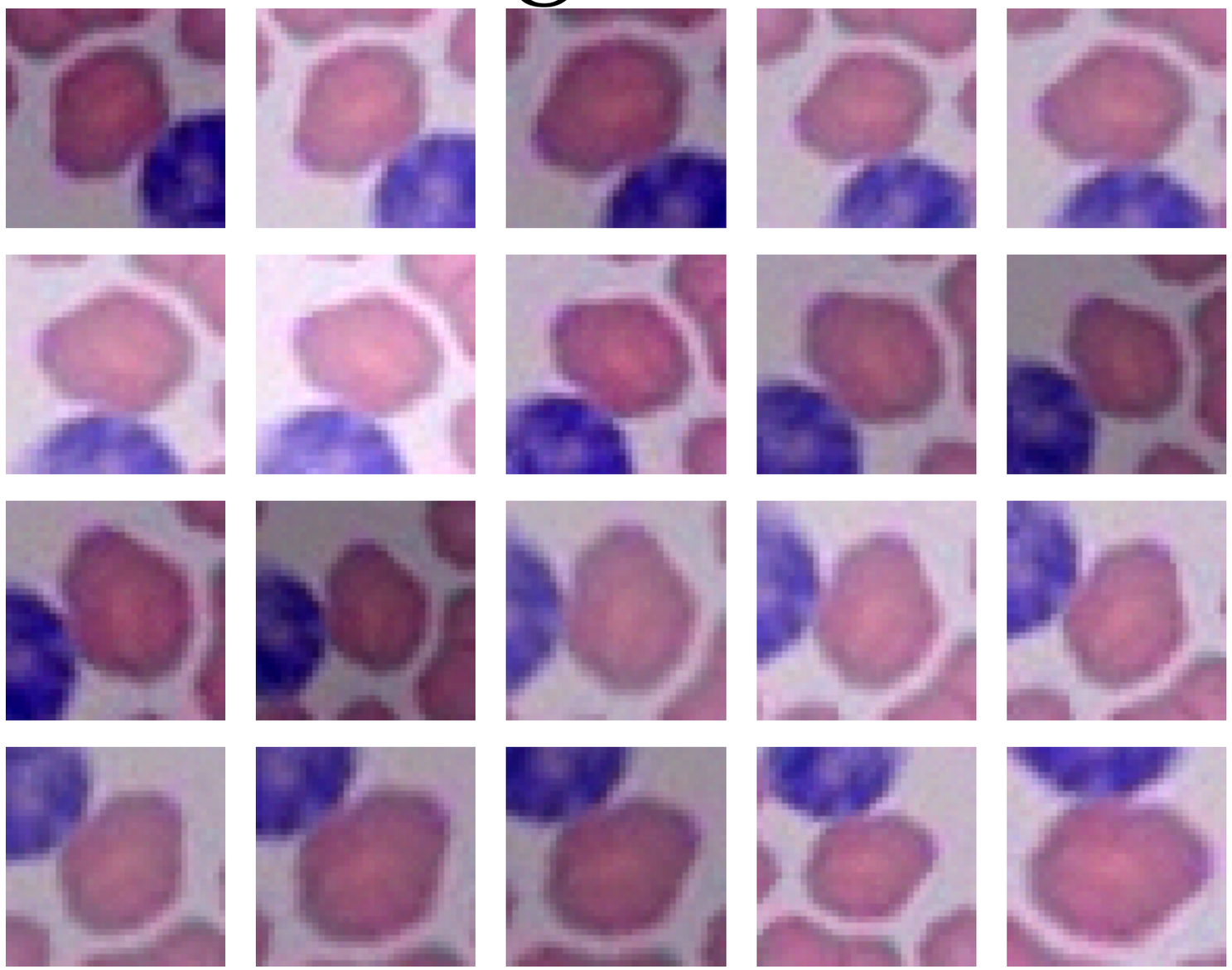
100 data points - Fitting 10th degree polynomial



$$y = x^2$$

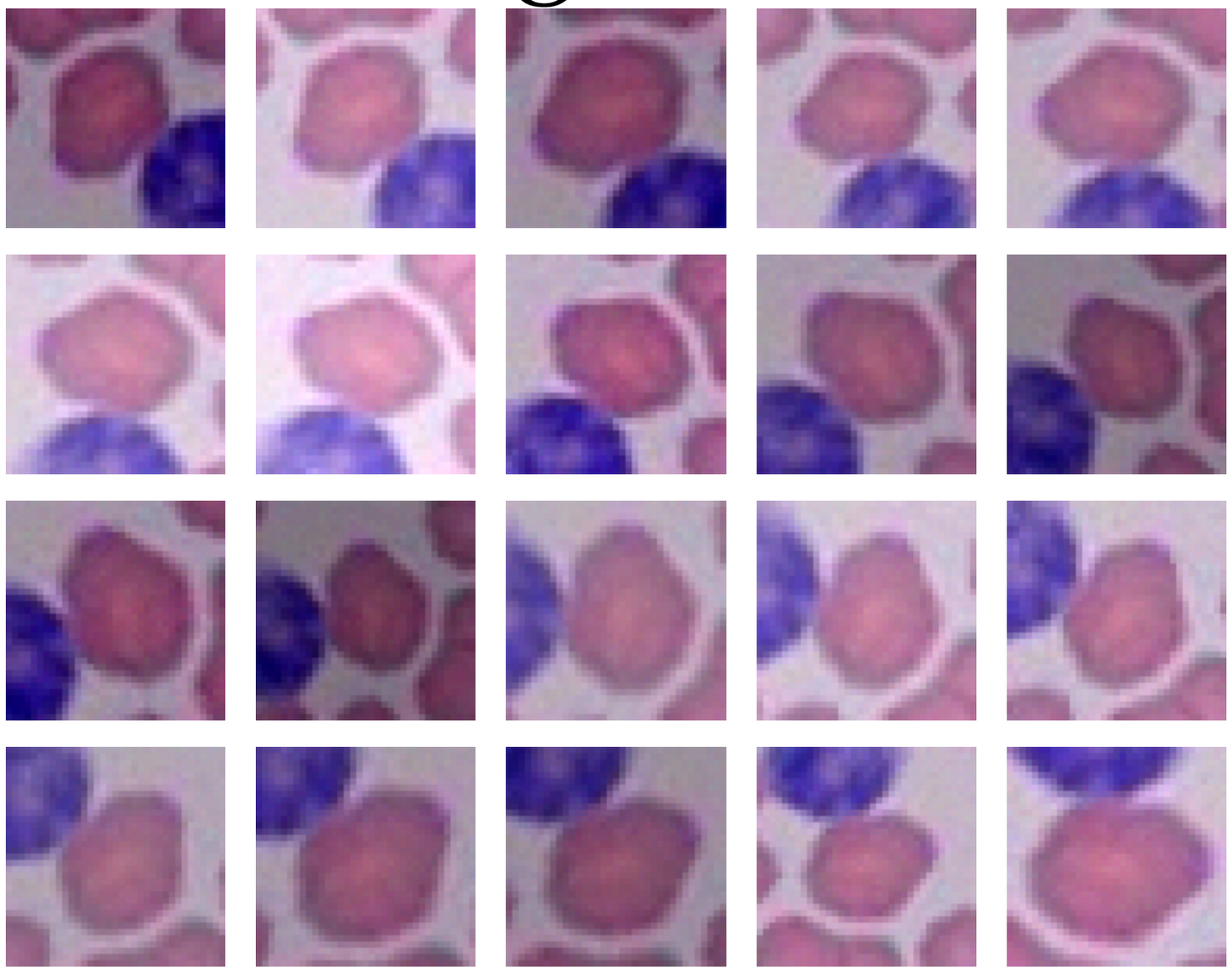
Data Augmentation

- Rotate
- Scale
- Change brightness
- Add noise



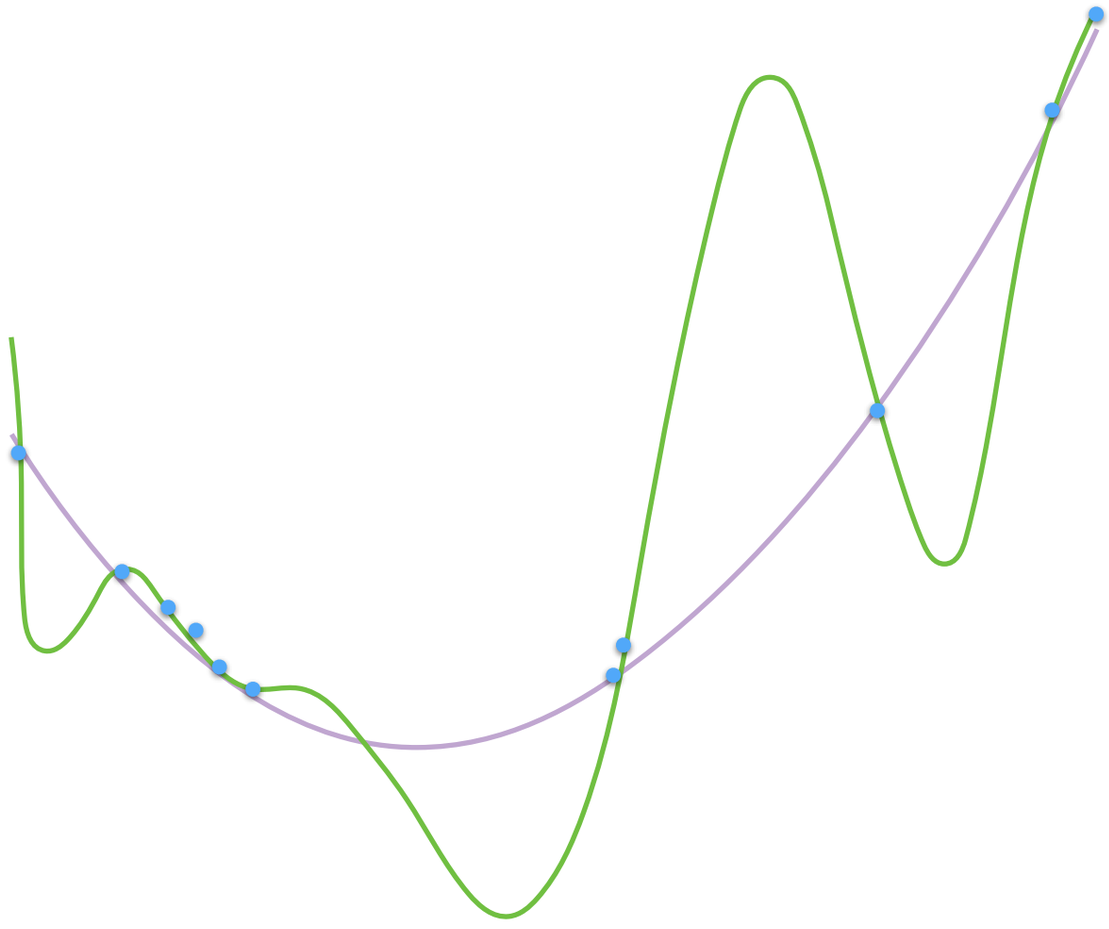
Data Augmentation

- Rotate
- Scale
- Change brightness
- Add noise



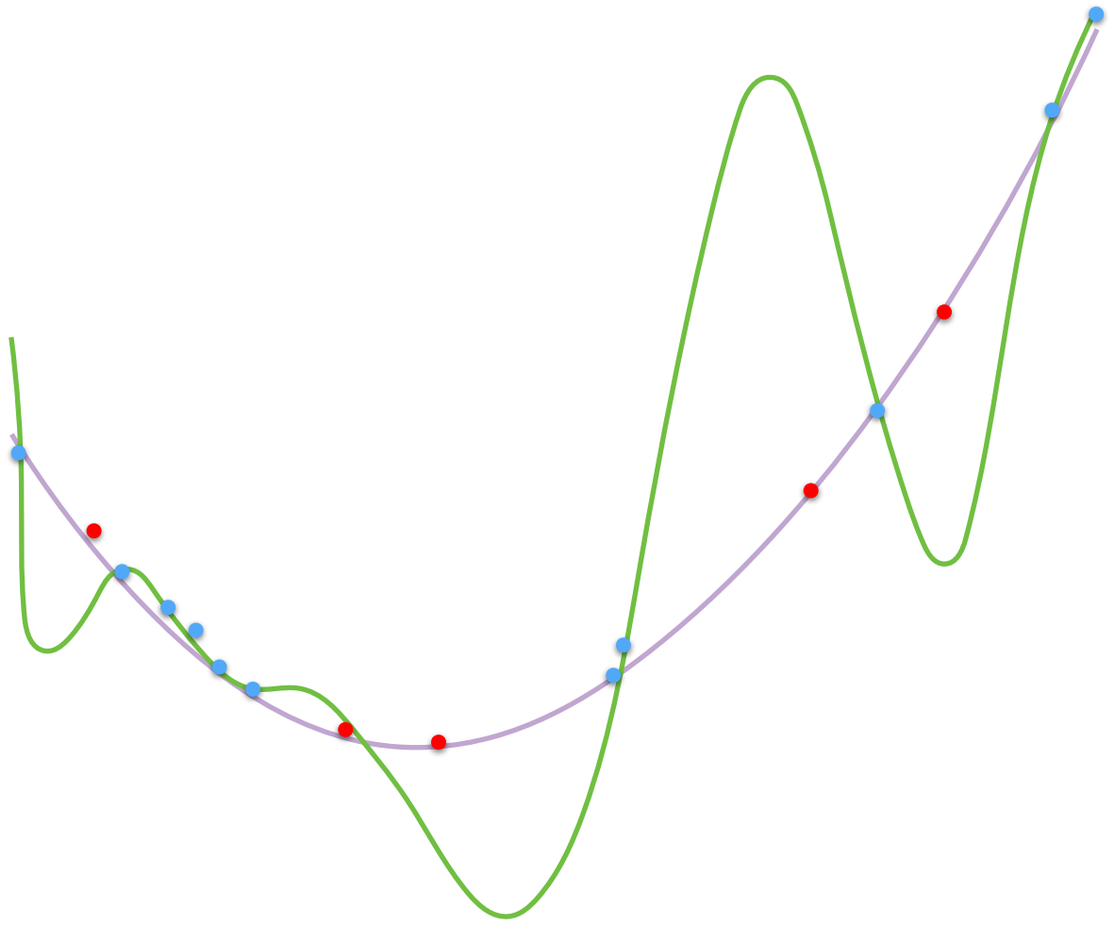
Average: (normalized)

Validation Data



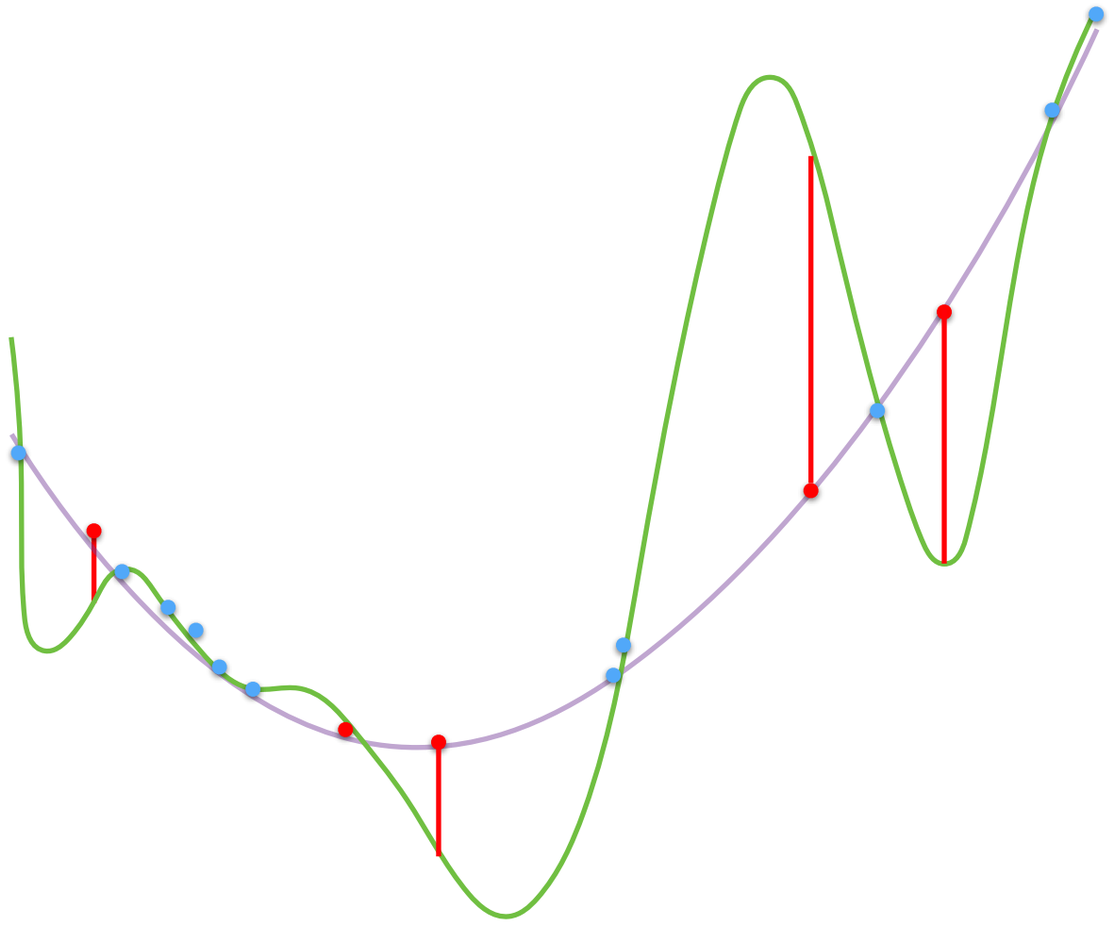
$$y = x^2$$

Validation Data



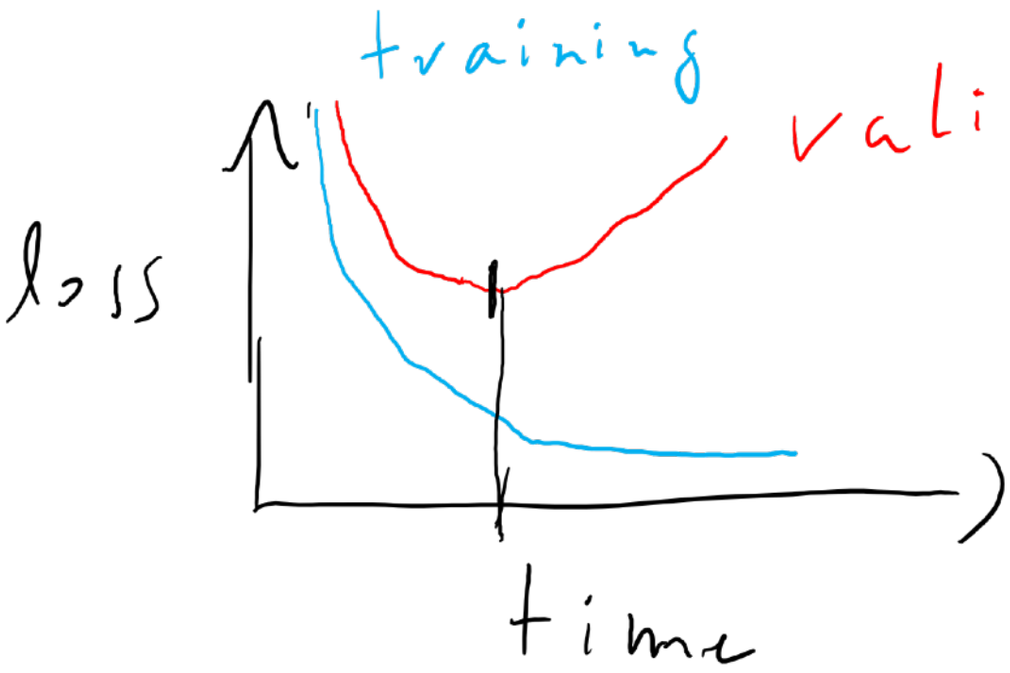
$$y = x^2$$

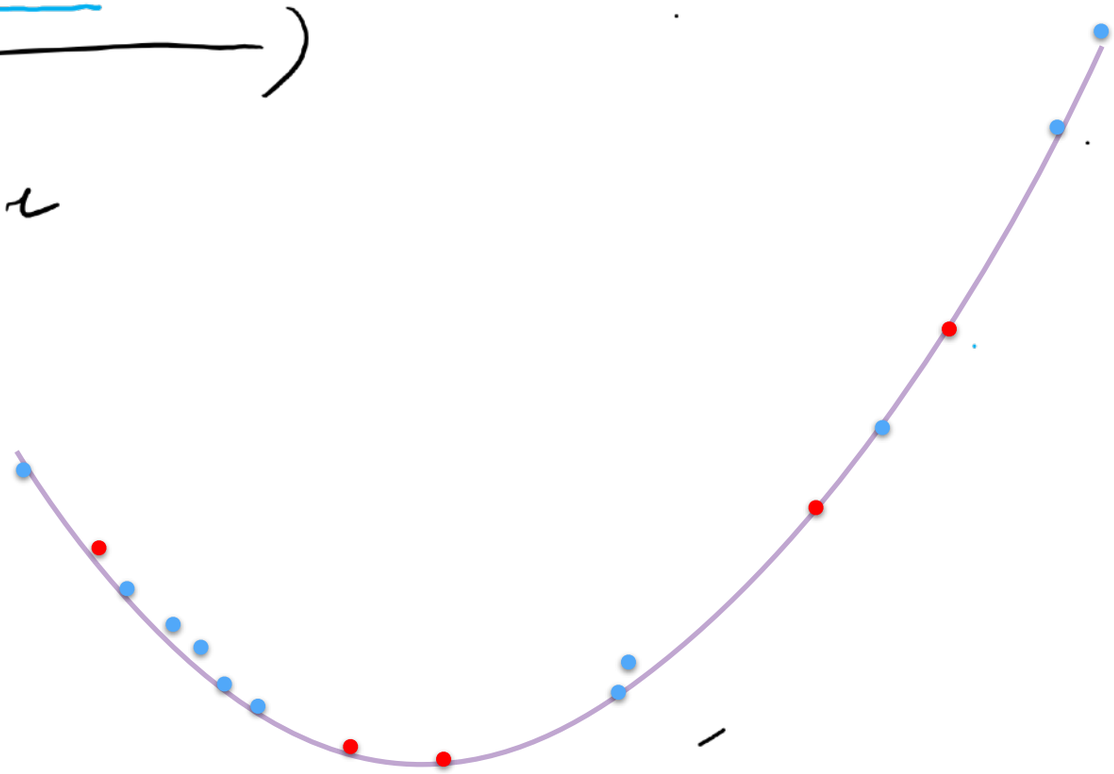
Validation Data



$$y = x^2$$

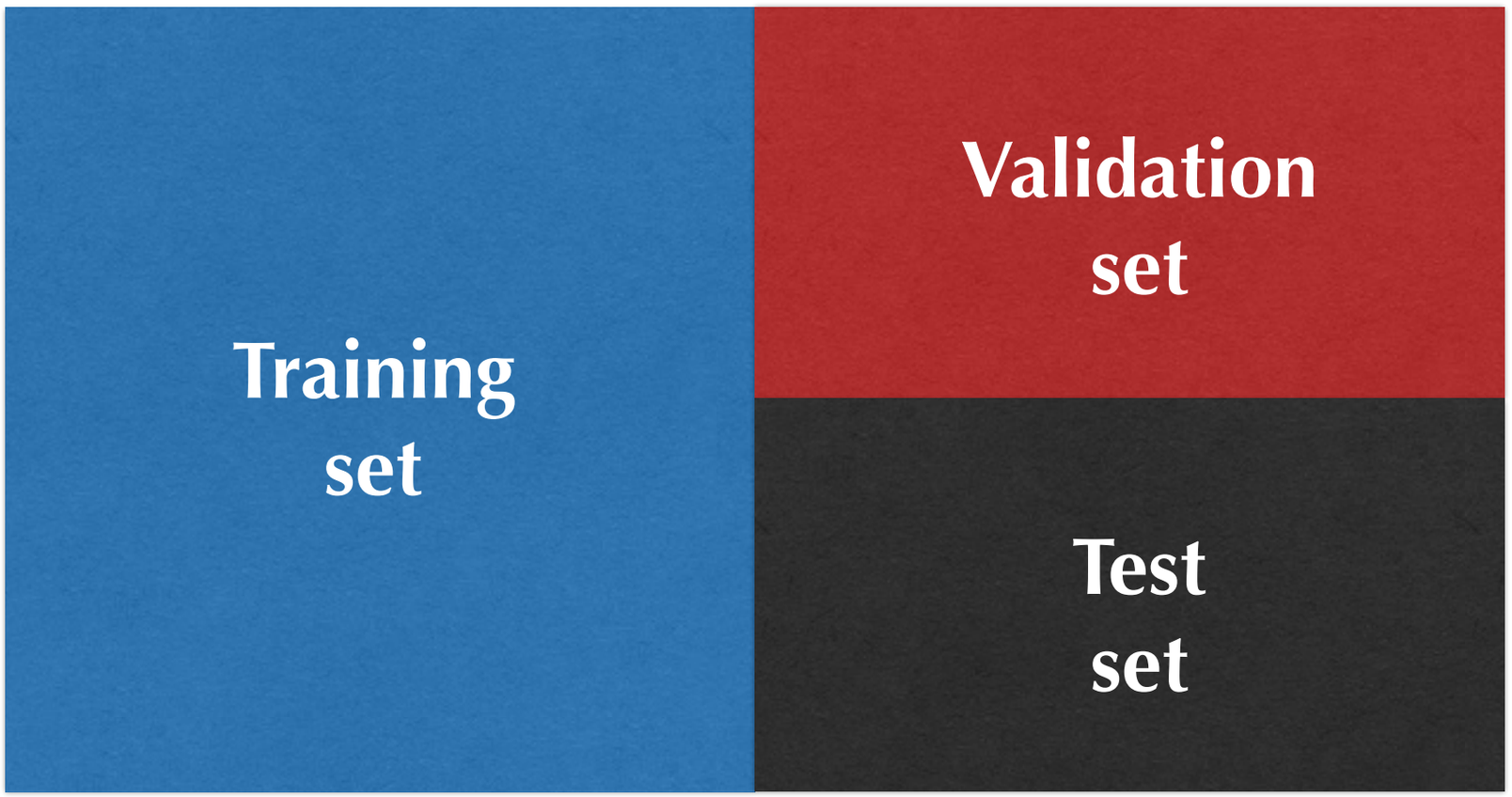
Validation Data



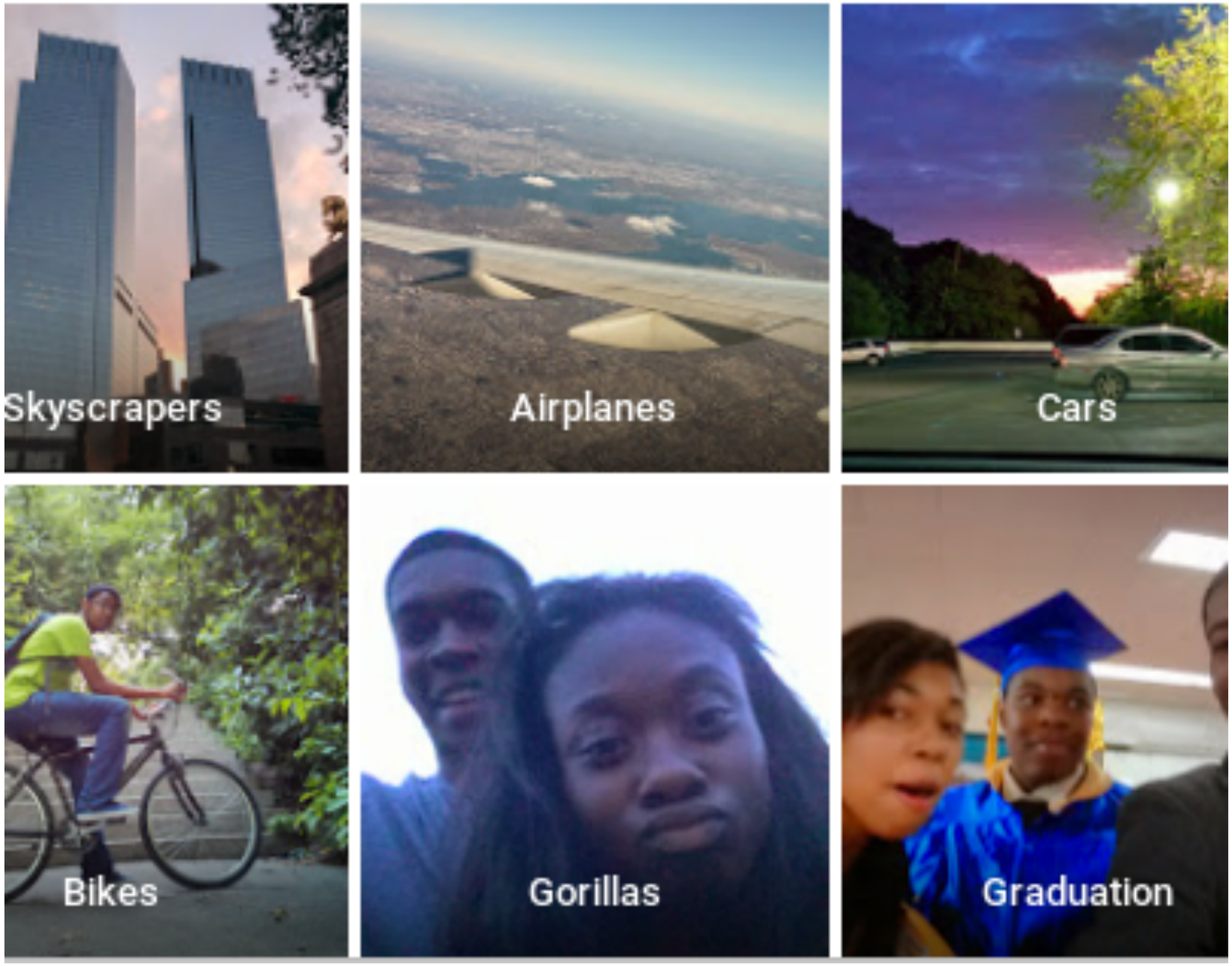


$$y = x^2$$

Training, Validation and Test

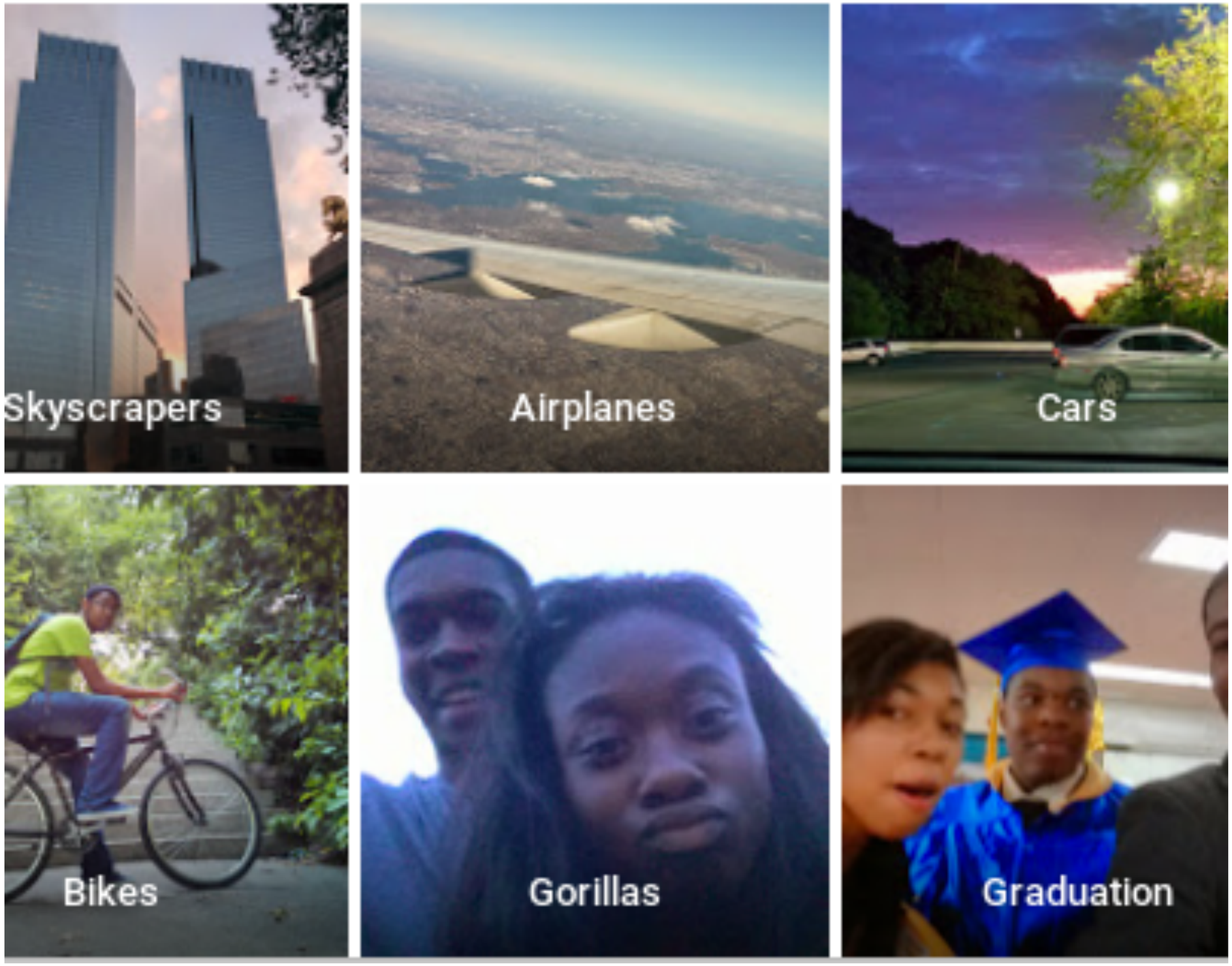


Dataset Bias



@jackyalcine 2015

Dataset Bias



@jackyalcine 2015

Lessons Learned

- Main lessons from this lecture
  - Learning a classifier via statistical learning
  - Minimizing negative log-likelihood via stochastic gradient descent
  - Importance of validation set

Lessons Learned

- Main lessons from this lecture
  - Learning a classifier via statistical learning
  - Minimizing negative log-likelihood via stochastic gradient descent
  - Importance of validation set
- Next lecture: Convolutional Neural Networks