Robust Regression

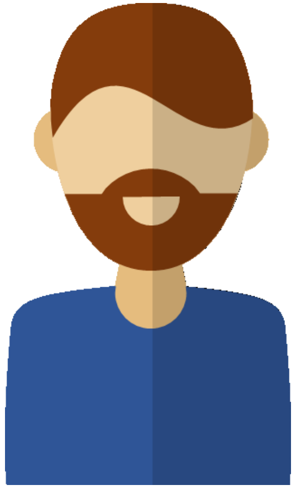
… and how I could relax after I stopped relaxing

Workshop on Algorithms and Optimization, ICTS, Bengaluru

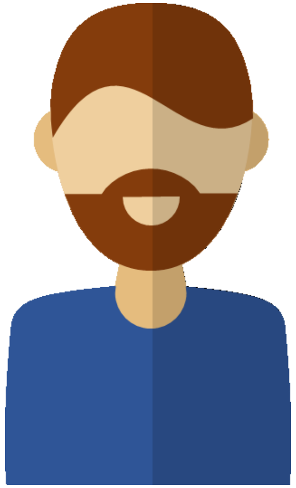
Purushottam Kar

A Recommendation System Problem

A Recommendation System Problem



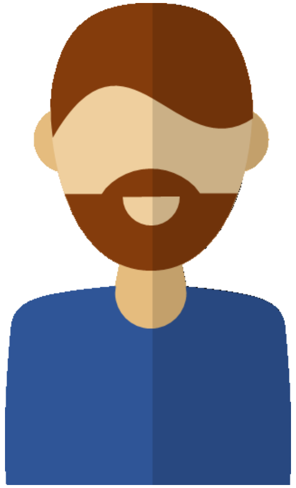
A Recommendation System Problem



ITEMS

$\mathbf{x}^1$ $\mathbf{x}^2$ $\mathbf{x}^3$ $\mathbf{x}^4$ $\mathbf{x}^5$ $\mathbf{x}^6$ $\mathbf{x}^7$

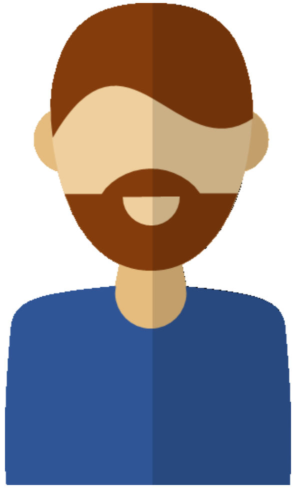
A Recommendation System Problem



ITEMS

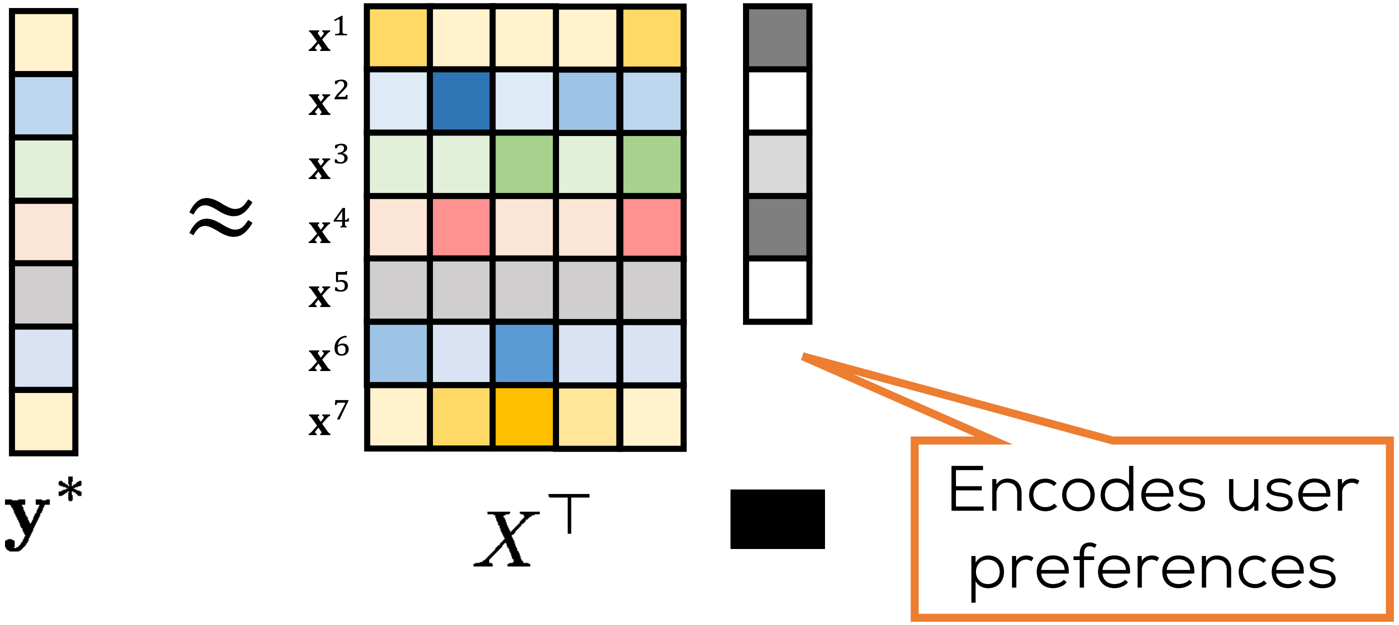
$\mathbf{x}^1$	$\mathbf{x}^2$	$\mathbf{x}^3$	$\mathbf{x}^4$	$\mathbf{x}^5$	$\mathbf{x}^6$	$\mathbf{x}^7$
0.5	0.2	0.2	0.5	0.9	0.8	0.4

A Recommendation System Problem

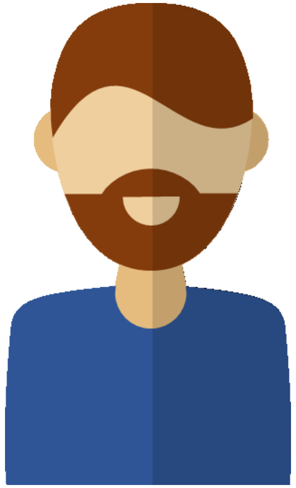


ITEMS

$\mathbf{x}^1$	$\mathbf{x}^2$	$\mathbf{x}^3$	$\mathbf{x}^4$	$\mathbf{x}^5$	$\mathbf{x}^6$	$\mathbf{x}^7$
0.5	0.2	0.2	0.5	0.9	0.8	0.4

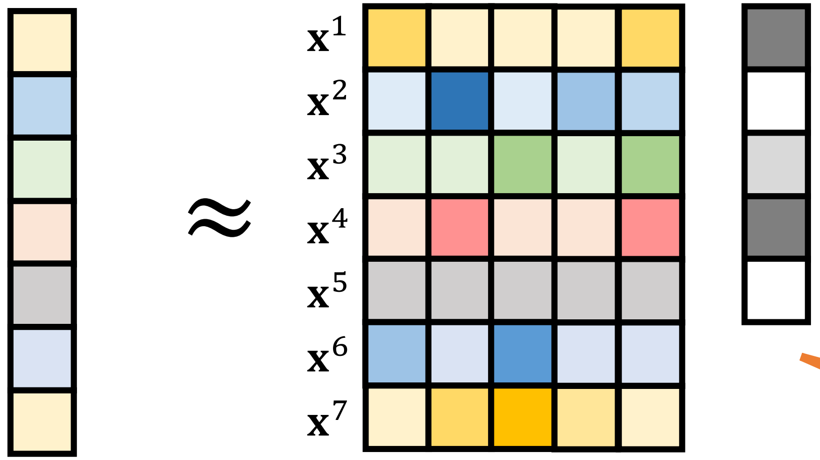


A Recommendation System Problem



ITEMS

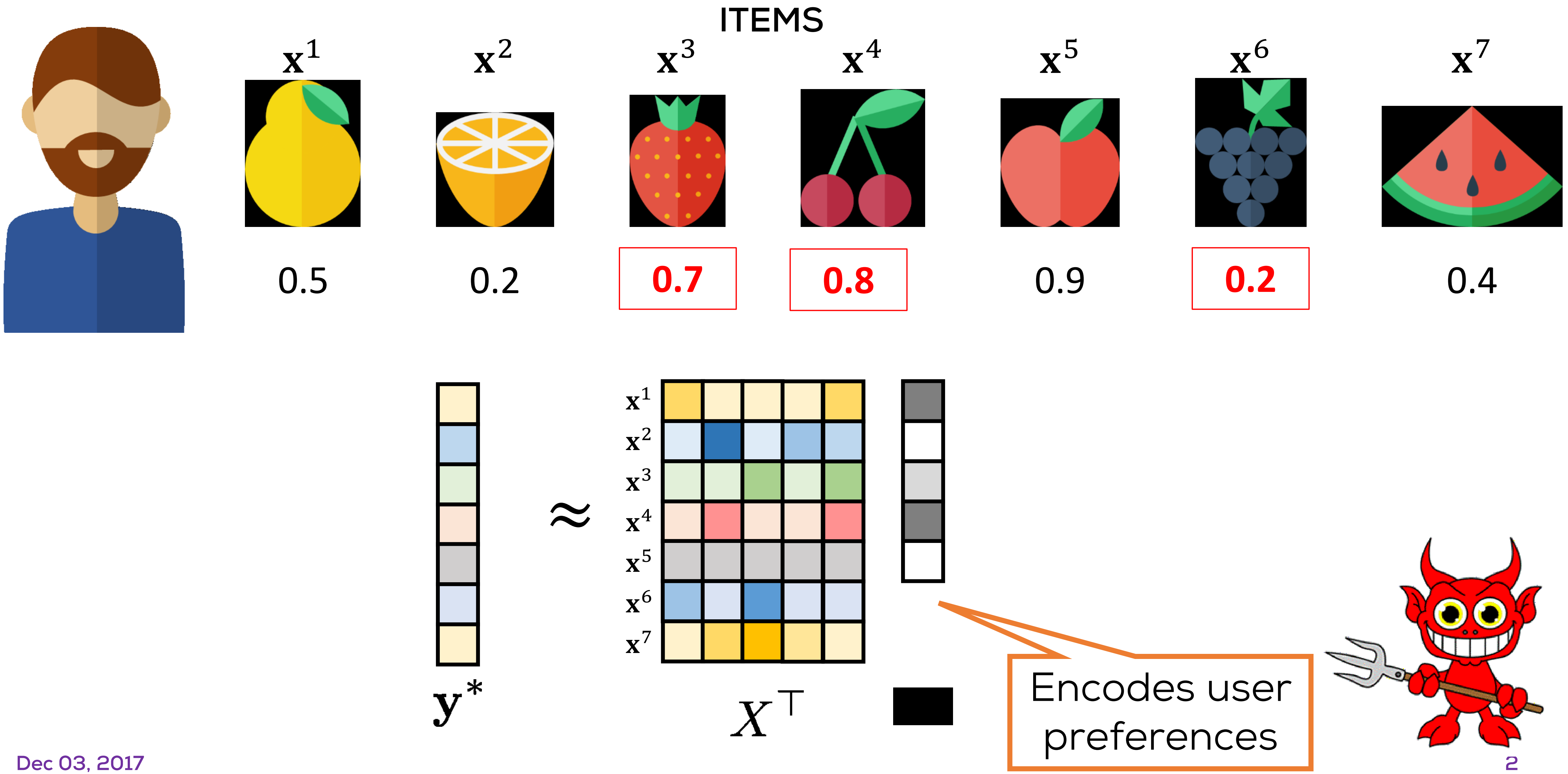
$\mathbf{x}^1$	$\mathbf{x}^2$	$\mathbf{x}^3$	$\mathbf{x}^4$	$\mathbf{x}^5$	$\mathbf{x}^6$	$\mathbf{x}^7$
0.5	0.2	0.2	0.5	0.9	0.8	0.4



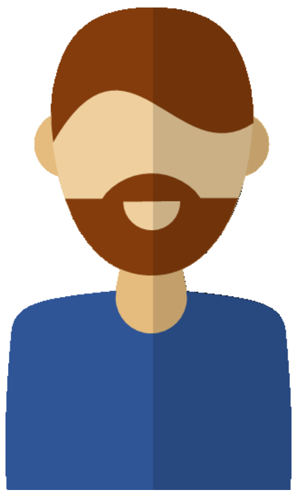
$$\mathbf{y}^* \approx X^\top$$ [redacted]

Encodes user preferences

A Recommendation System Problem

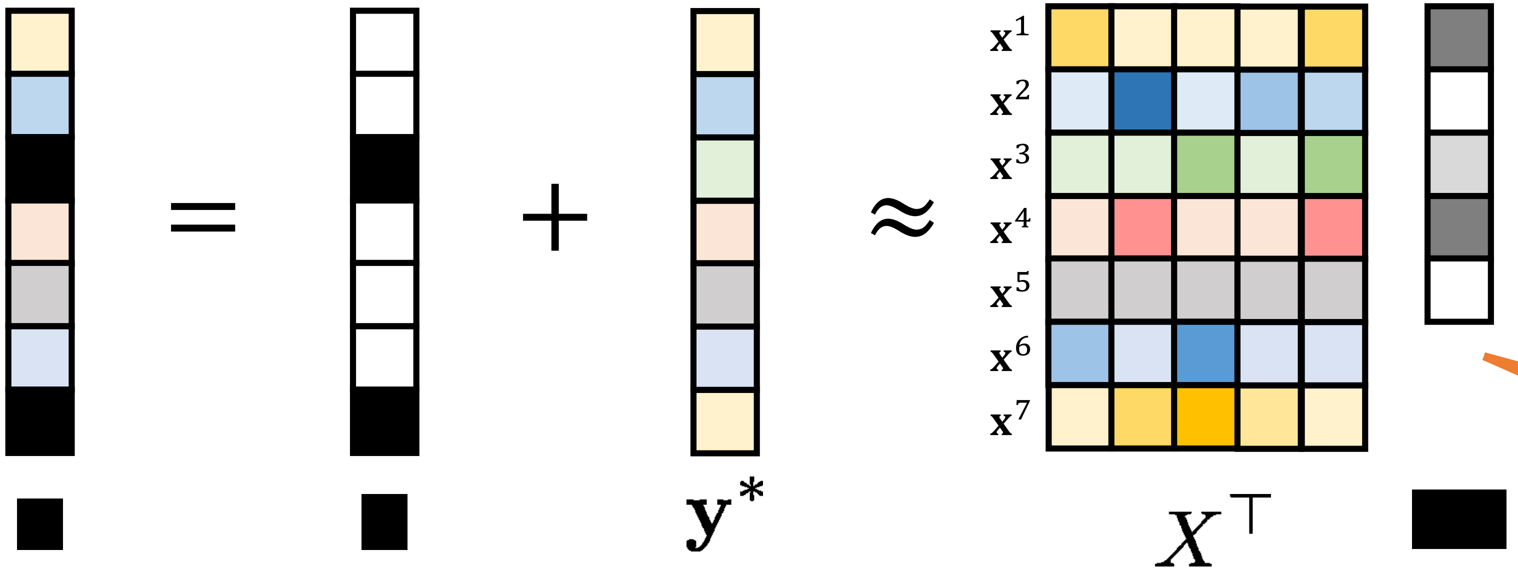


A Recommendation System Problem



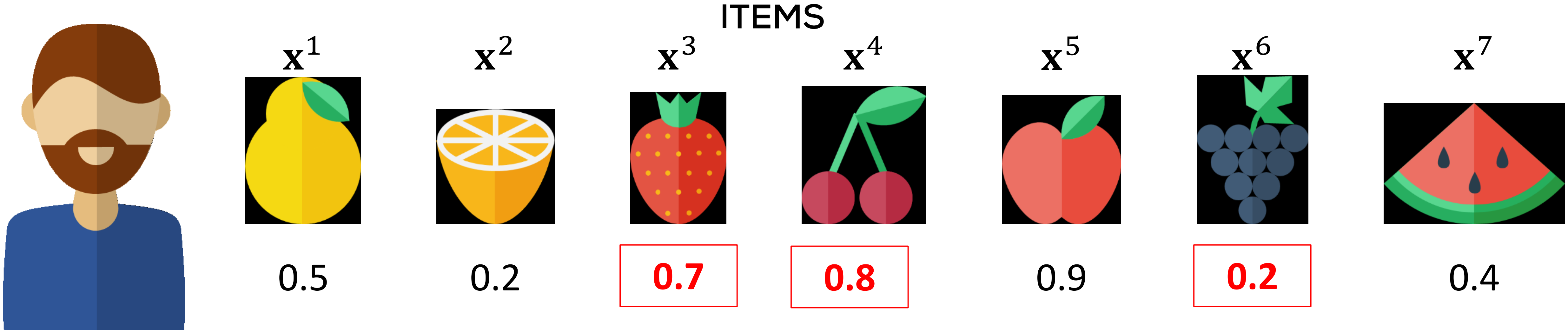
ITEMS

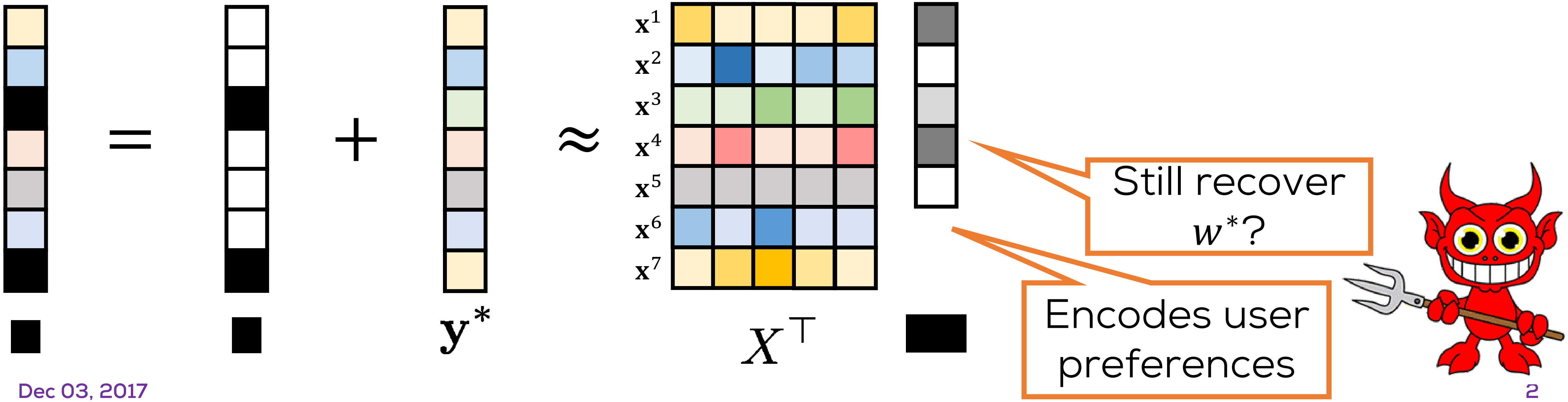
$\mathbf{x}^1$	$\mathbf{x}^2$	$\mathbf{x}^3$	$\mathbf{x}^4$	$\mathbf{x}^5$	$\mathbf{x}^6$	$\mathbf{x}^7$
0.5	0.2	**0.7**	**0.8**	0.9	**0.2**	0.4



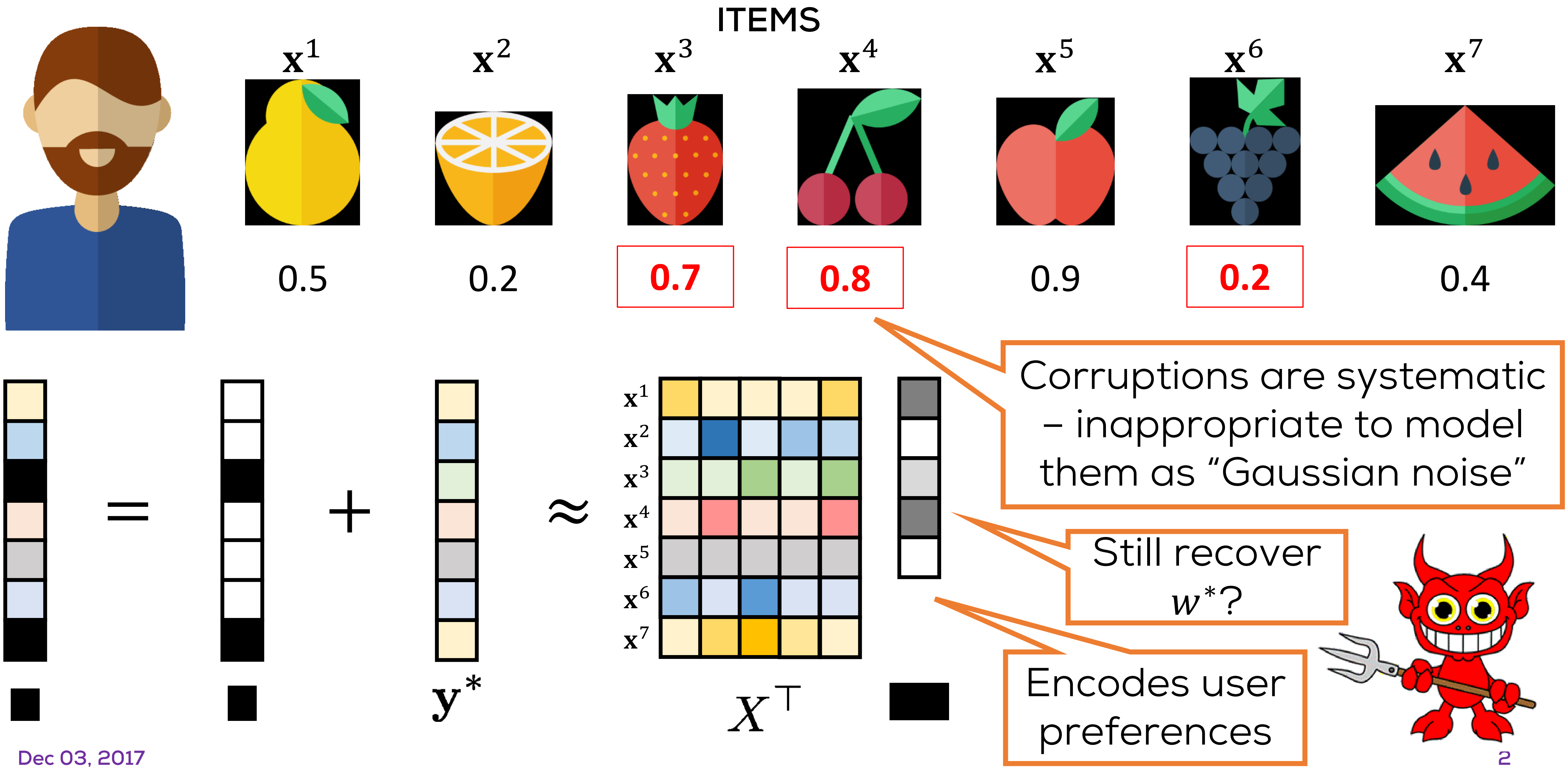
Encodes user preferences

A Recommendation System Problem

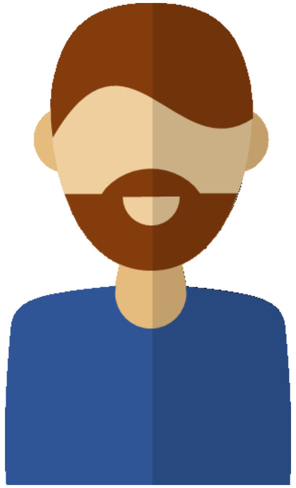




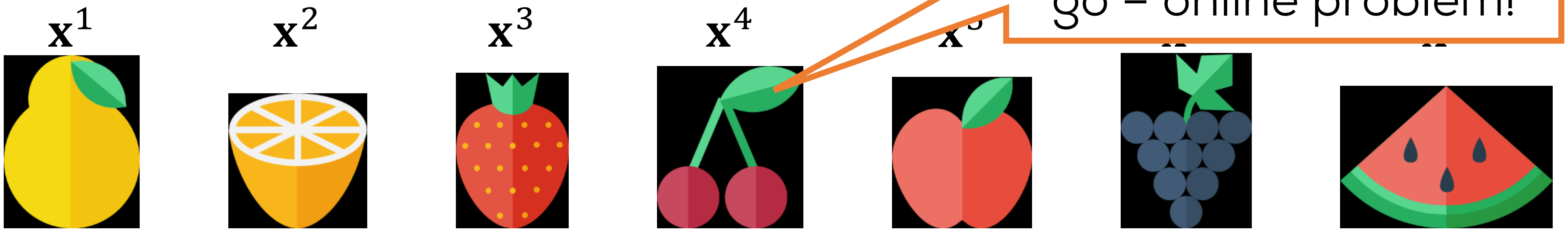
A Recommendation System Problem



A Recommendation System Prob



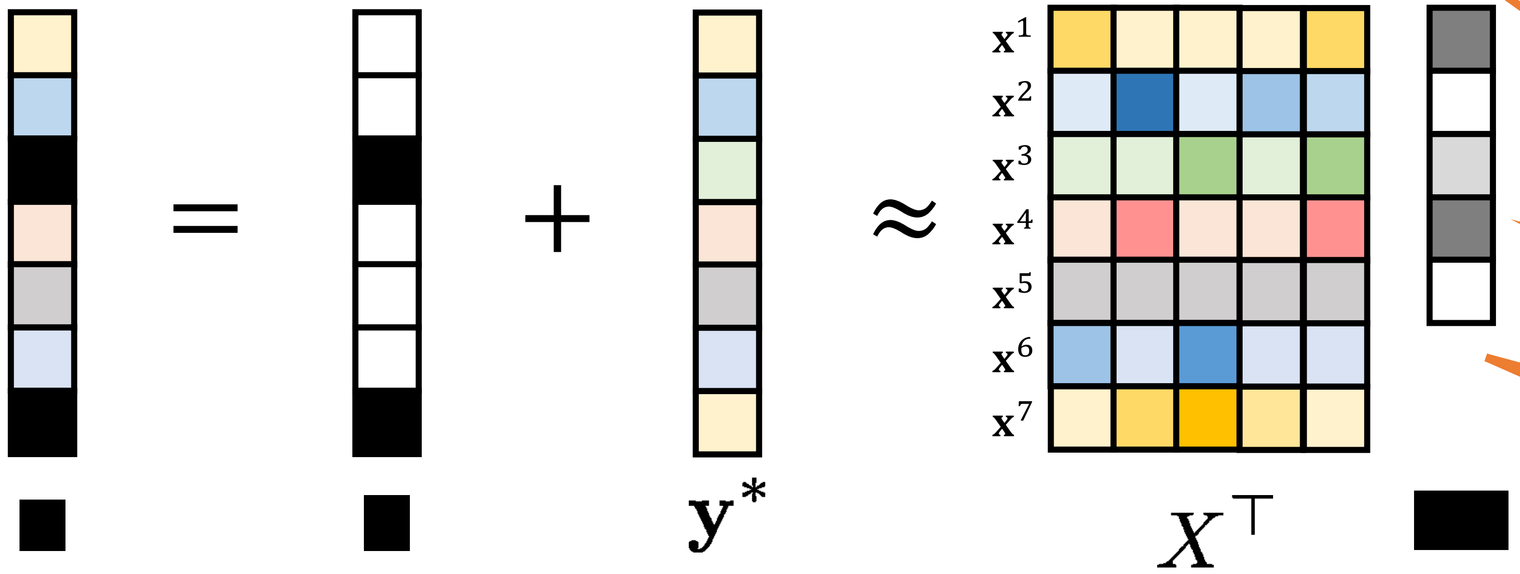
ITEMS



0.5	0.2	**0.7**	**0.8**	0.9	**0.2**	0.4

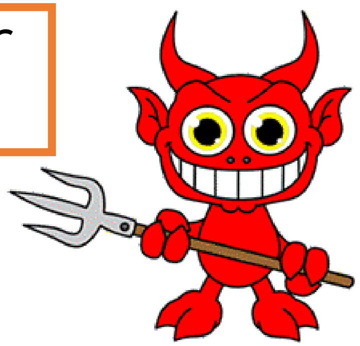
The items and ratings are not received in one go – online problem!

Corruptions are systematic – inappropriate to model them as "Gaussian noise"



Still recover w^*?

Encodes user preferences



A Biometric Identification Problem

A Biometric Identification Problem

DATABASE

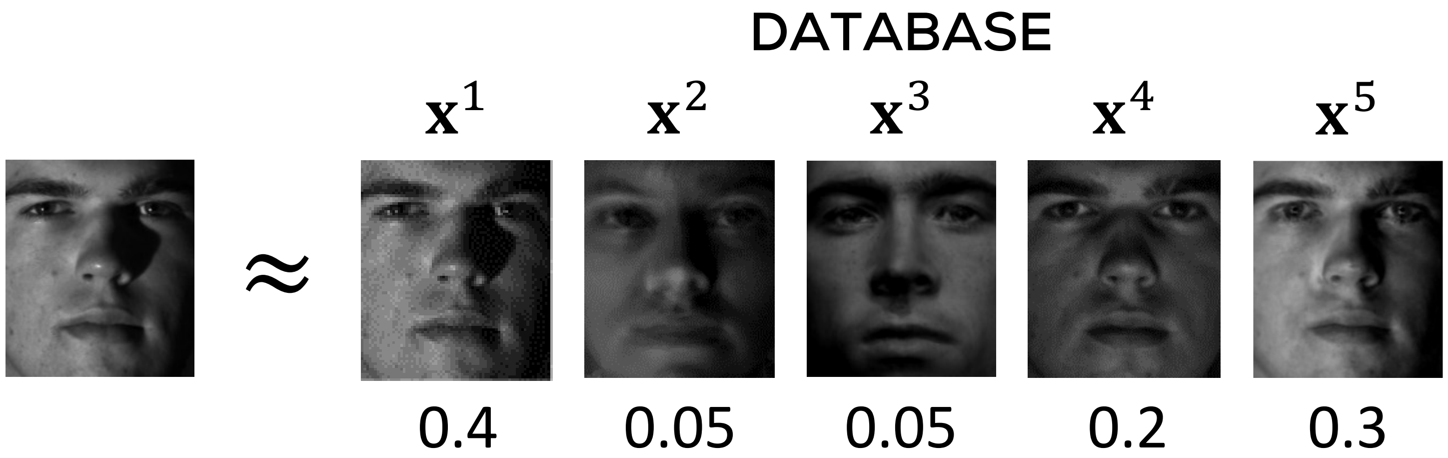
$\mathbf{x}^1$ $\mathbf{x}^2$ $\mathbf{x}^3$ $\mathbf{x}^4$ $\mathbf{x}^5$

A Biometric Identification Problem

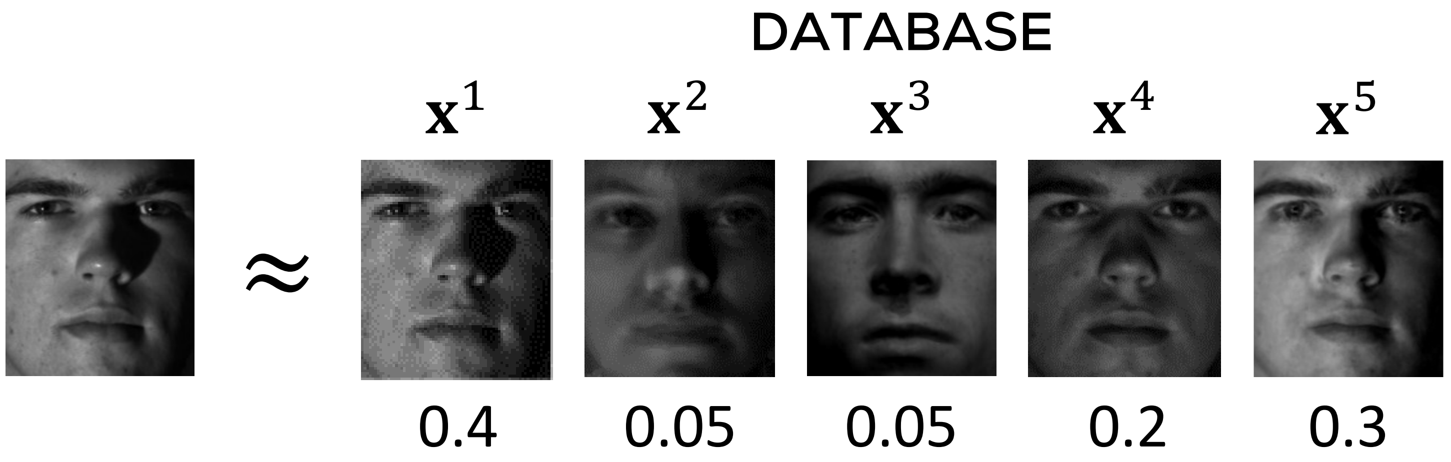
DATABASE

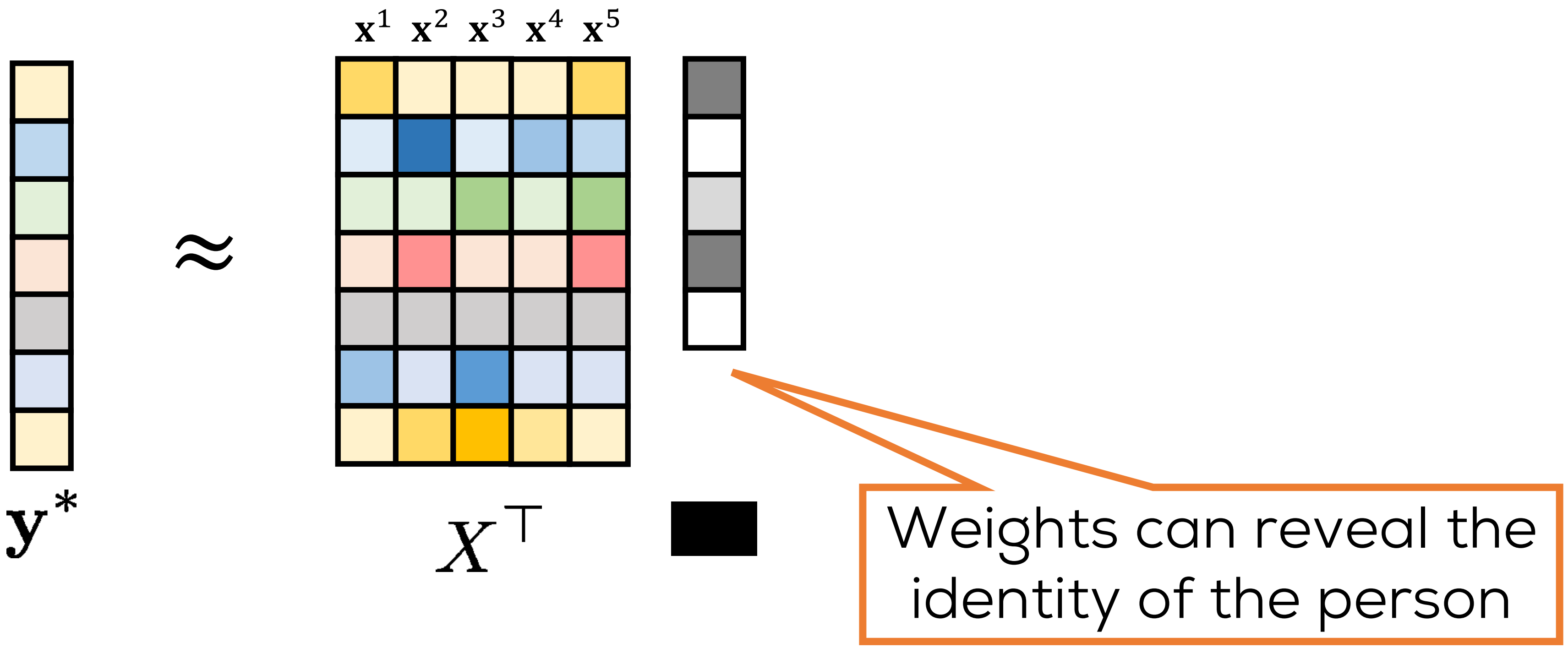
$\mathbf{x}^1$ $\mathbf{x}^2$ $\mathbf{x}^3$ $\mathbf{x}^4$ $\mathbf{x}^5$

A Biometric Identification Problem



A Biometric Identification Problem

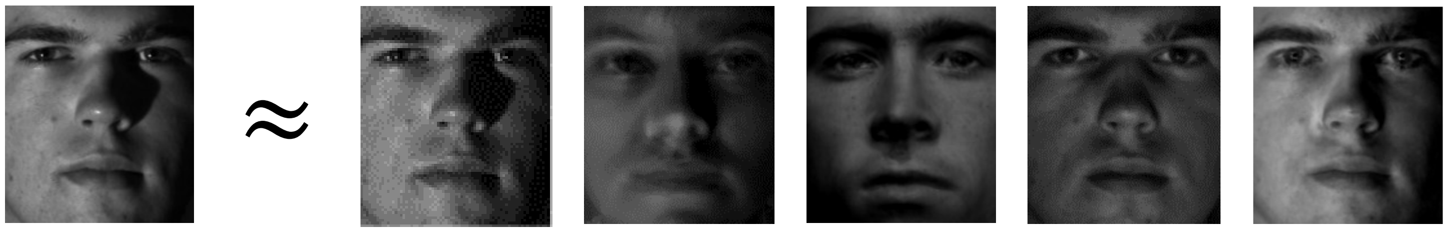




A Biometric Identification Problem

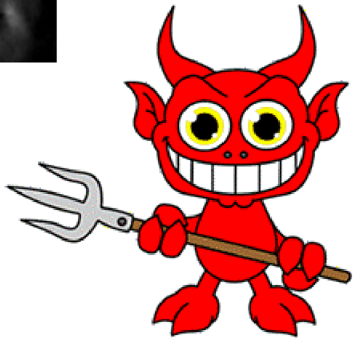
DATABASE

$\mathbf{x}^1$ $\mathbf{x}^2$ $\mathbf{x}^3$ $\mathbf{x}^4$ $\mathbf{x}^5$

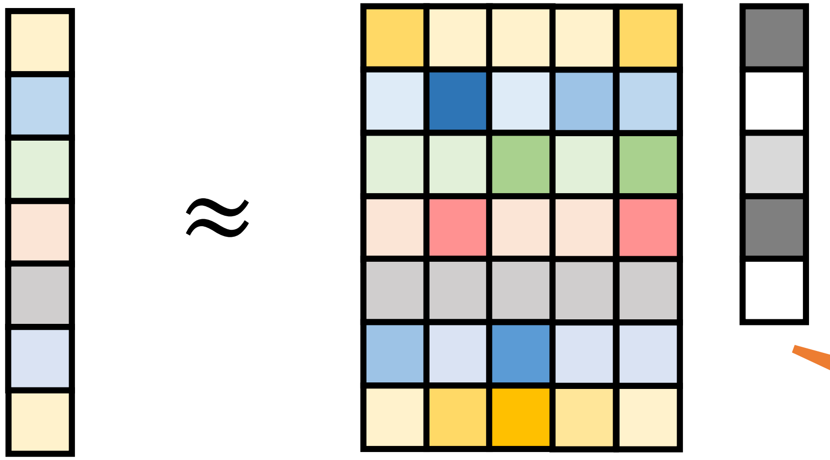


$\approx$

0.4 0.05 0.05 0.2 0.3



$\mathbf{x}^1$ $\mathbf{x}^2$ $\mathbf{x}^3$ $\mathbf{x}^4$ $\mathbf{x}^5$

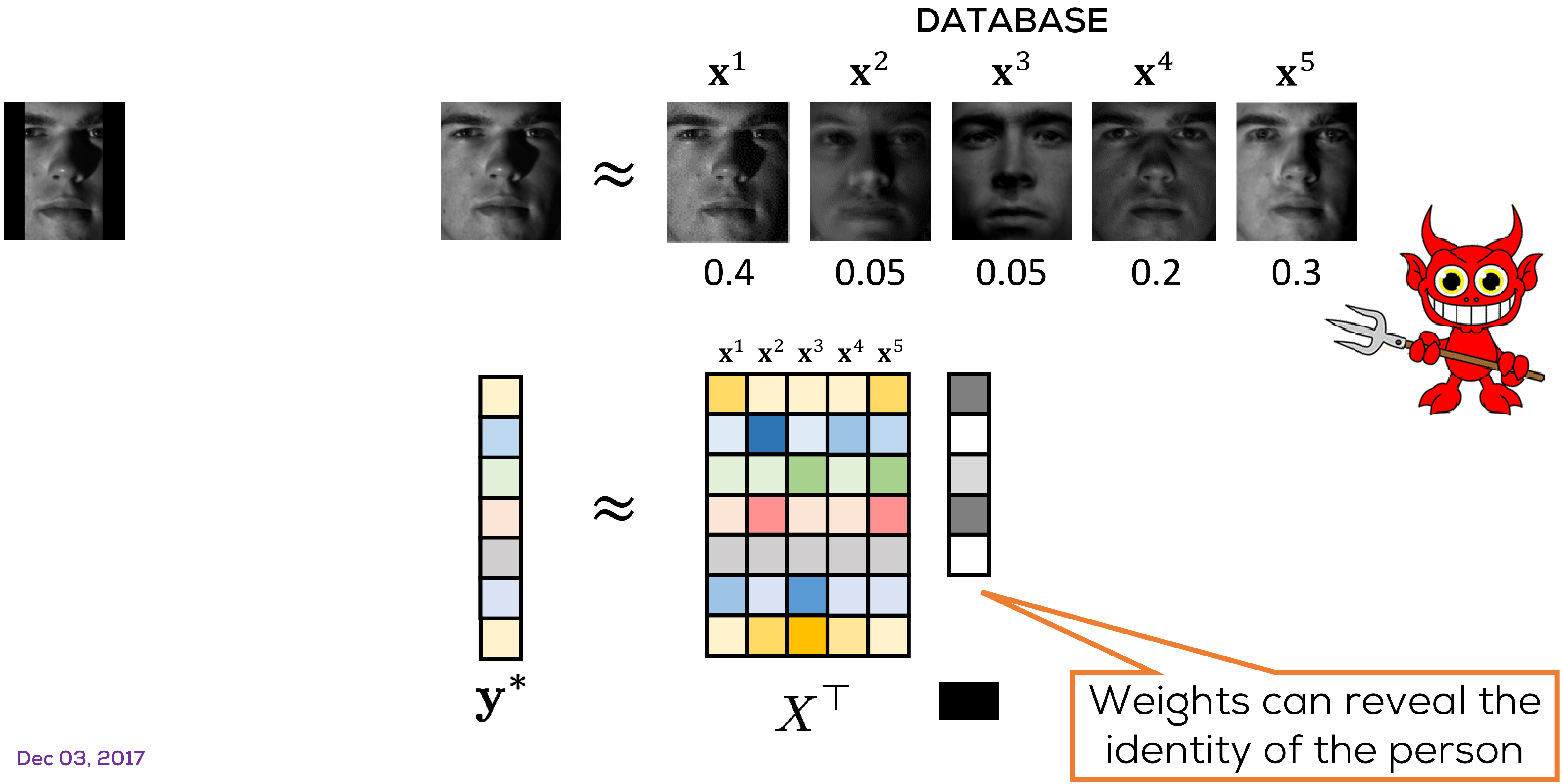


$\approx$

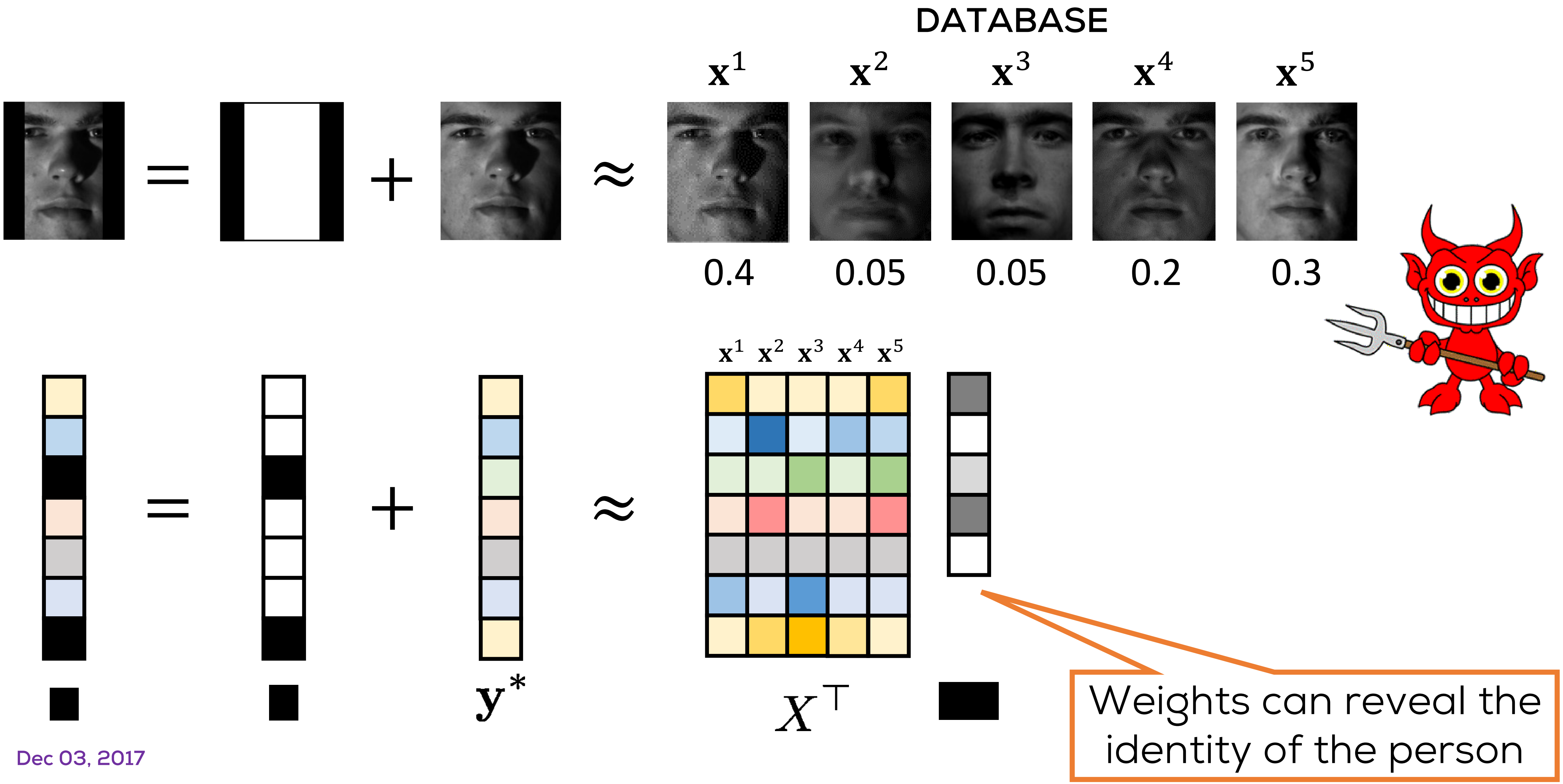
$\mathbf{y}^*$ $X^\top$

Weights can reveal the identity of the person

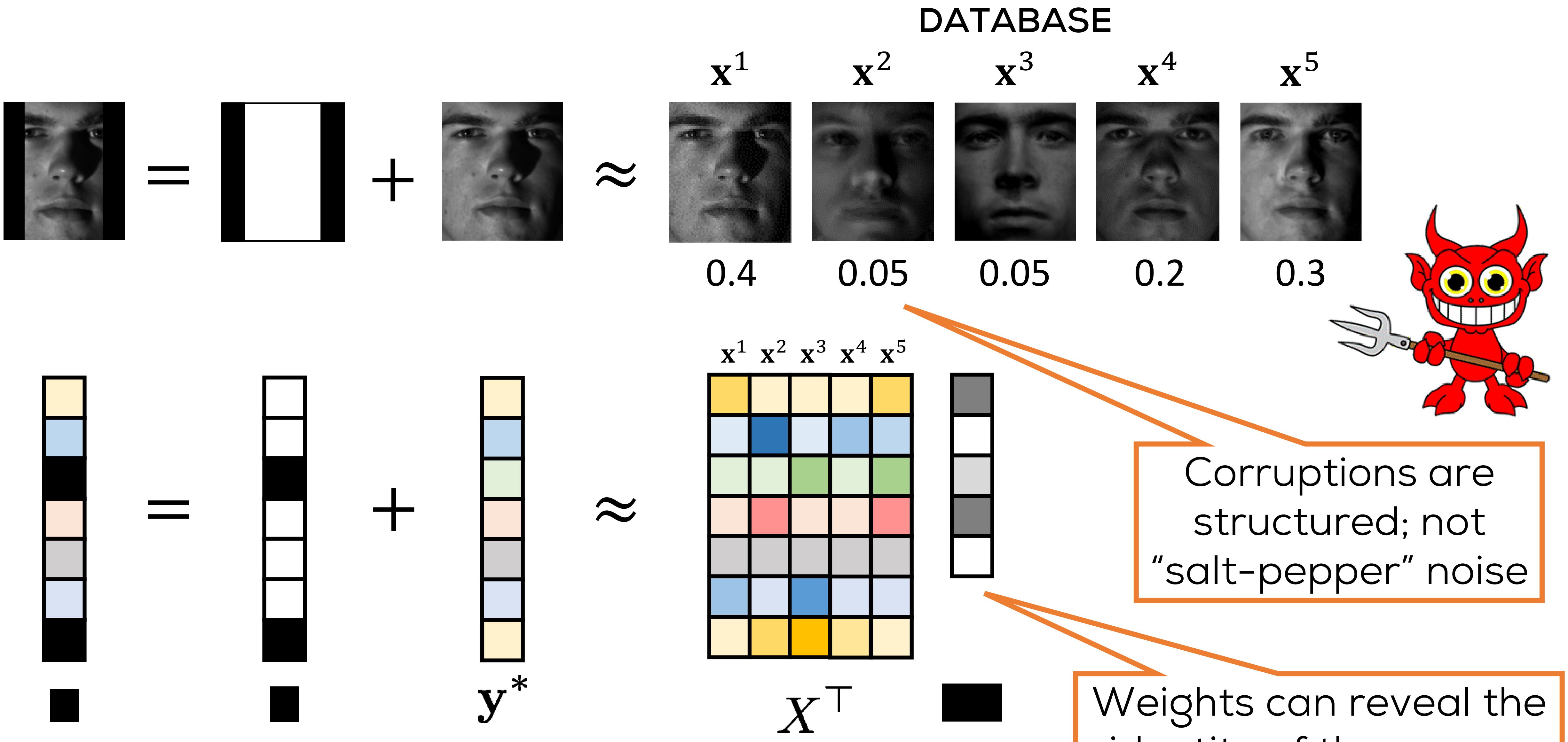
A Biometric Identification Problem



A Biometric Identification Problem



A Biometric Identification Problem



Robust Learning and Estimation

- Classical subfield of statistics (Huber, 1964, Tukey, 1960)
- Newfound interest – scalability and efficiency
  - Robust classification (Feng et al, 2014)
  - Robust regression (Bhatia et al 2015, Chen et al, 2013)
  - Robust PCA (Candès et al, 2009, Netrapalli et al, 2014)
  - Robust matrix completion (Cherapanamjeri et al, 2017)
  - Robust optimization (Charikar et al, 2016)
  - Robust estimation (Diakonikolas et al, 2017, Lai et al, 2016, Pravesh's talk)
- Extremely exciting area – from theory and app perspectives

Robust Learning and Estimation - Application

Netrapalli et al, 2014, Bhatia et al, 2015

Robust Learning and Estimation - Application

- When data is actually corrupted e.g. click fraud in RecSys

Netrapalli et al, 2014, Bhatia et al, 2015

Robust Learning and Estimation - Application

- When data is actually corrupted e.g. click fraud in RecSys
- Even when data not corrupted (to allow problem reformulation)

Netrapalli et al, 2014, Bhatia et al, 2015

Robust Learning and Estimation - Application

- When data is actually corrupted e.g. click fraud in RecSys
- Even when data not corrupted (to allow problem reformulation)
  - Foreground-background extraction can be cast as robust PCA

Netrapalli et al, 2014, Bhatia et al, 2015

Robust Learning and Estimation – Application

- When data is actually corrupted e.g. click fraud in RecSys
- Even when data not corrupted (to allow problem reformulation)
  - Foreground-background extraction can be cast as robust PCA

Netrapalli et al, 2014, Bhatia et al, 2015

Robust Learning and Estimation - Application

- When data is actually corrupted e.g. click fraud in RecSys
- Even when data not corrupted (to allow problem reformulation)
  - Foreground-background extraction can be cast as robust PCA

=

+

Netrapalli et al, 2014, Bhatia et al, 2015

Robust Learning and Estimation - Application

- When data is actually corrupted e.g. click fraud in RecSys
- Even when data not corrupted (to allow problem reformulation)
  - Foreground-background extraction can be cast as robust PCA

  - Image in-painting can be cast as robust regression

Netrapalli et al, 2014, Bhatia et al, 2015

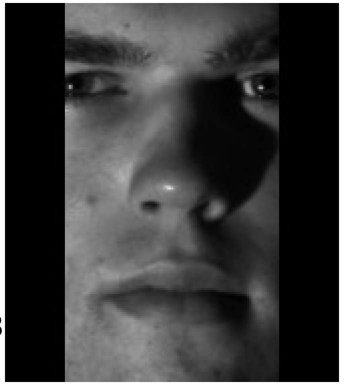
Robust Learning and Estimation - Application

- When data is actually corrupted e.g. click fraud in RecSys
- Even when data not corrupted (to allow problem reformulation)
  - Foreground-background extraction can be cast as robust PCA


=

+


  - Image in-painting can be cast as robust regression

Netrapalli et al, 2014, B

Robust Learning and Estimation – Application

- When data is actually corrupted e.g. click fraud in RecSys
- Even when data not corrupted (to allow problem reformulation)
  - Foreground-background extraction can be cast as robust PCA

  - Image in-painting can be cast as robust regression

trapalli et al, 2014, B

A Toy Problem befitting this near-lunch Hour

- A linear least-squares regression problem
- We have n data points $(\mathbf{x}^i, y^i) \in \mathbb{R}^d \times \mathbb{R}$
- Most of the points are *clean* i.e. for some (unknown) $\mathbf{w}^* \in \mathbb{R}^d$
$$y^i = \langle \mathbf{w}^*, \mathbf{x}^i \rangle$$
- However, k of the points were corrupted by
$$y^i = \langle \mathbf{w}^*, \mathbf{x}^i \rangle + b_i^*$$

- Let S^* denote the set of $n - k$ uncorrupted points
- (When/How) can we recover $\mathbf{w}^*$ from the data?
- Will see an extremely simple and intuitive algorithm
- … and its proof of optimality (sorry ☹ but proof will be light ☺)

Notation

- Let $\mathbf{y} = [y^1, \dots, y^n]^\top \in \mathbb{R}^n, \mathbf{b}^* = [b_1^*, \dots, b_n^*]^\top \in \mathbb{R}^n, X = [\mathbf{x}^1, \dots, \mathbf{x}^n] \in \mathbb{R}^{d \times n}$
$$\mathbf{y} = X^\top \mathbf{w}^* + \mathbf{b}^*$$
- Assume for sake of simplicity that $\|\mathbf{x}^i\|_2 = 1$ for all $i \in [n]$
- Recall since only k points corrupted, $\|\mathbf{b}^*\|_0 \leq k$ and $S^* = \overline{\text{supp}(\mathbf{b}^*)}$
$$\|\mathbf{v}\|_0 = |\{i : \mathbf{v}_i \neq 0\}|$$
- For $S \subseteq [n], \mathbf{y}_S, \mathbf{b}_S^* \in \mathbb{R}^{|S|}, X_S \in \mathbb{R}^{d \times |S|}$ denote subvectors/matrices
- Let $C = XX^\top$ and for any $S \subseteq [n]$, denote $C_S = X_S X_S^\top$

Some Solution Strategies

- Discover the clean set S^* and recover $\mathbf{w}^*$ from it

$$\min_{|S|=n-k} \min_{\mathbf{w}} \left\| \mathbf{y}_S - X_S^\top \mathbf{w} \right\|_2^2 = \min_{|S|=n-k} \min_{\mathbf{w}} \sum_{i \in S} \left(y^i - \langle \mathbf{w}, \mathbf{x}^i \rangle \right)^2$$

- Discover the corruptions $\mathbf{b}^*$ and clean up the responses $\mathbf{y}$

$$\min_{\|\mathbf{b}\|_0 = k} \min_{\mathbf{w}} \|(\mathbf{y} - \mathbf{b}) - X^\top \mathbf{w}\|_2^2$$

- However, the above problems are combinatorial, even NP-hard
- Relax, and just relax!

$$\min_{\|\mathbf{b}\|_1 \leq r} \min_{\mathbf{w}} \|(\mathbf{y} - \mathbf{b}) - X^\top \mathbf{w}\|_2^2$$

$$\min_{\mathbf{w}, \mathbf{b}} \|(\mathbf{y} - \mathbf{b}) - X^\top \mathbf{w}\|_2^2 + \lambda \cdot \|\mathbf{b}\|_1$$

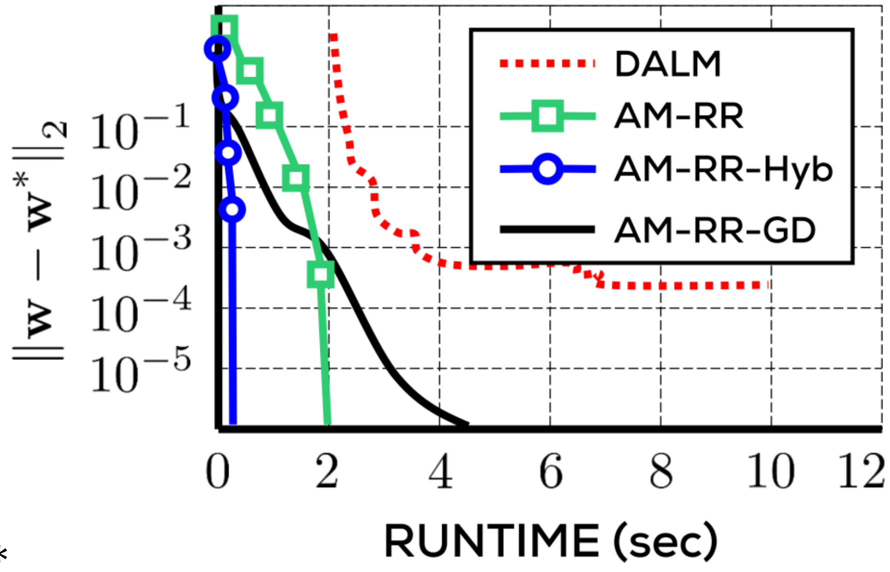
- Extremely popular and well-studied approach in image proc etc.

An "Alternate" Viewpoint

- Relaxation methods can be very slow in practice
- Will see one very simple way to do better
- Reconsider the formulation

$$\min_{|S|=n-k} \min_{\mathbf{w}} \left\| \mathbf{y}_S - X_S^\top \mathbf{w} \right\|_2^2$$

- Two variables of interest $\mathbf{w}^*$ and S^*
- Recovering either one recovers the other
  - If we know S^*, do least squares on it to get $\mathbf{w}^*$
  - If we know $\mathbf{w}^*$, points with zero error gives S^*
- Hmm … what if we only have "good" estimates?
  - Can a good estimate S of S^* get me a good estimate $\mathbf{w}$ of $\mathbf{w}^*$?
  - Can that good estimate $\mathbf{w}$ get me a still better estimate of S^*?



AM-RR: Alt. Min. for Rob. Reg.

AM-RR

1. Data $X \in \mathbb{R}^{d \times n}, \mathbf{y} \in \mathbb{R}^n$, # bad pts k
2. Initialize $S^1 \leftarrow [1 : n-k]$
3. For $t = 1, 2, \ldots, T$

$$\mathbf{w}^{t+1} = \arg\min_{\mathbf{w}} \left\| \mathbf{y}_{S^t} - X_{S^t}^\top \mathbf{w} \right\|_2^2$$

$$S^{t+1} = \arg\min_{|S| = n-k} \left\| \mathbf{y}_S - X_S^\top \mathbf{w}^{t+1} \right\|_2^2$$

4. Repeat until convergence

Maintain an "active" set S^t: points that we believe are clean

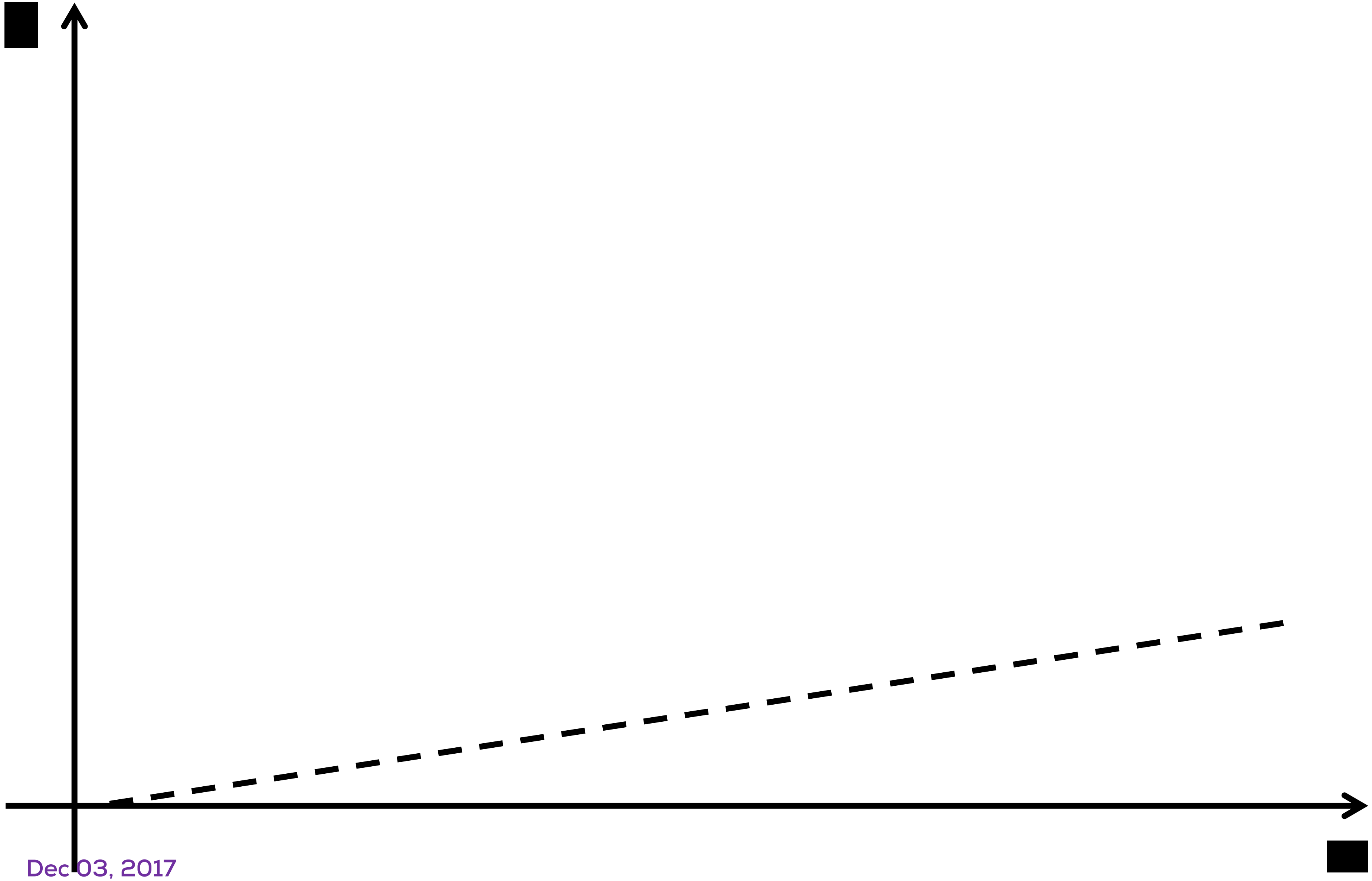
Solve a least squares problem on active set

Find points which seem least corrupted with respected to $\mathbf{w}^{t+1}$

Residual $\mathbf{r}^{t+1} = \mathbf{y} - X\mathbf{w}^{t+1}$
Find points with least res.

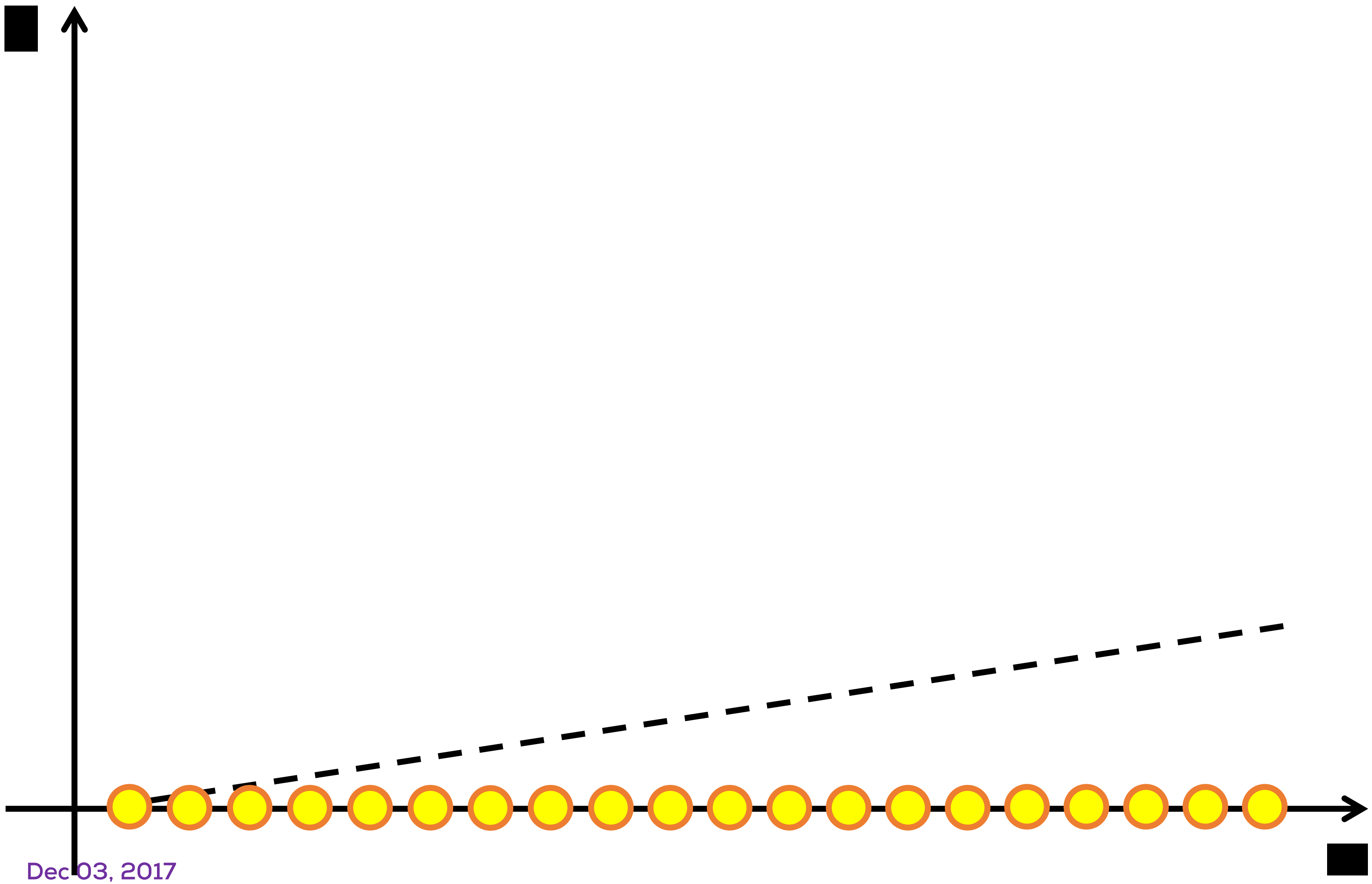
AM-RR at work

AM-RR at work



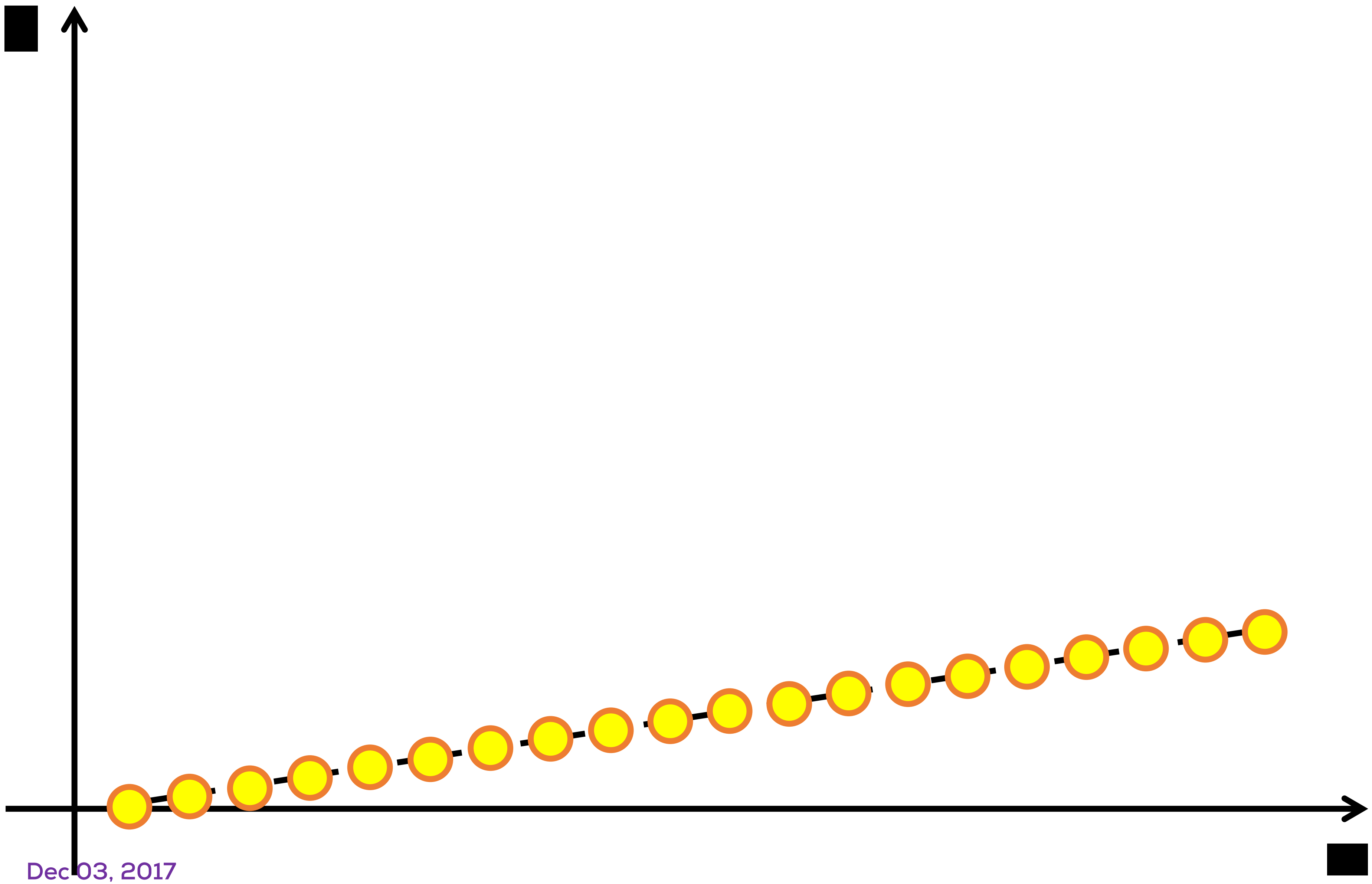
[redacted]

AM-RR at work



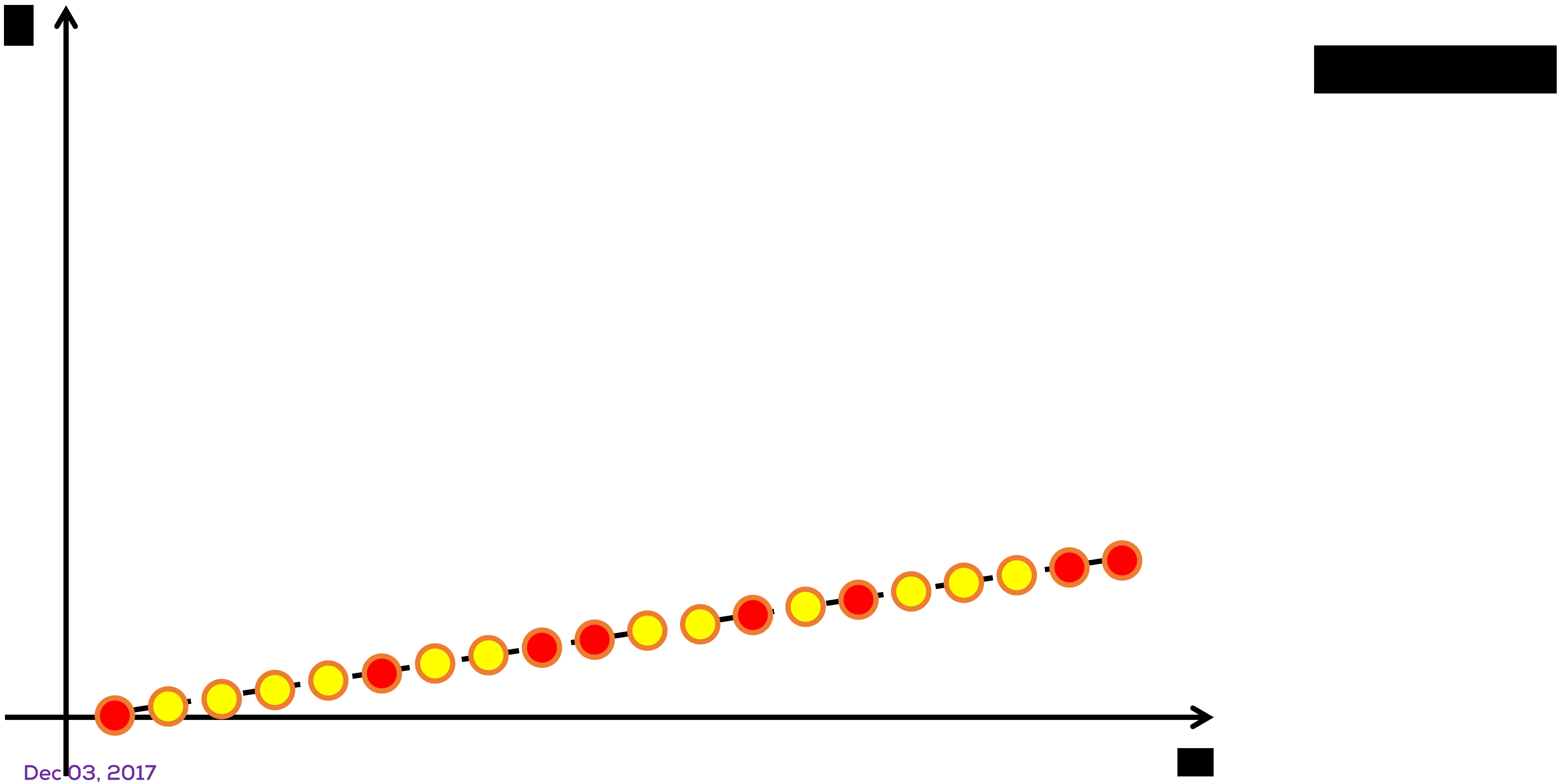
[redacted]

AM-RR at work

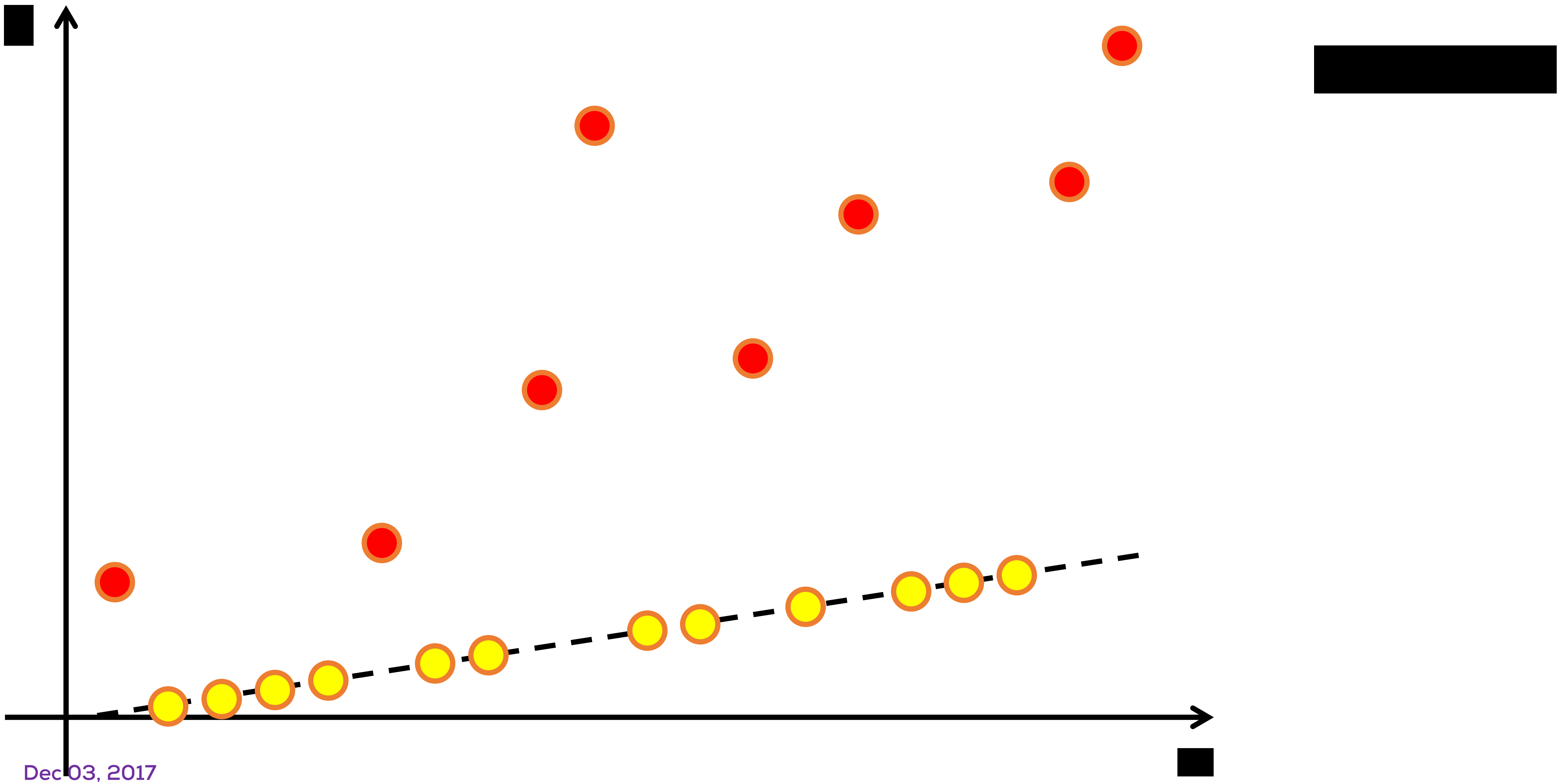


[redacted]

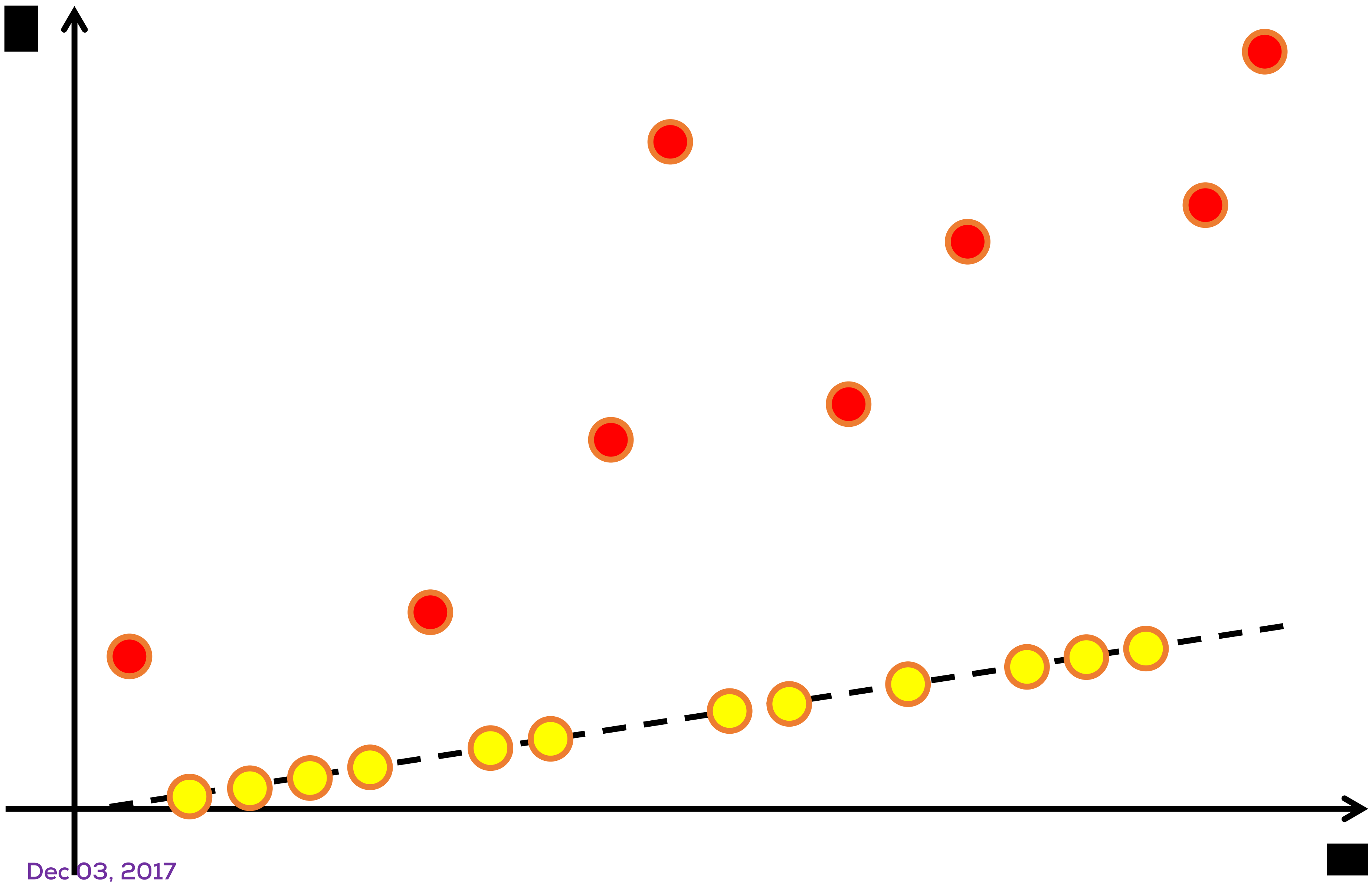
AM-RR at work



AM-RR at work



AM-RR at work

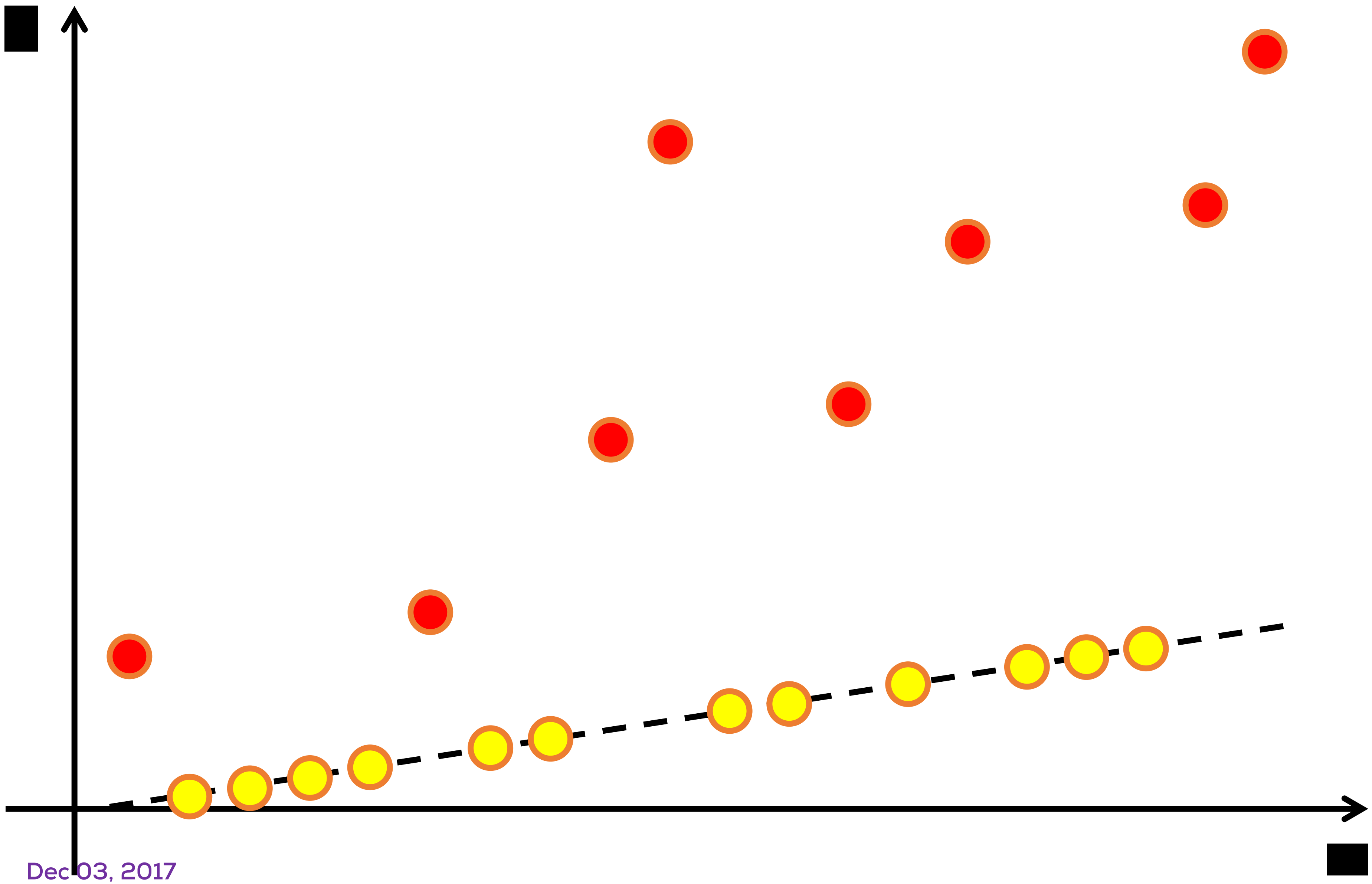


[redacted]

Given $\hat{\mathbf{w}}$, easy to identify points that *look* like ●

Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

AM-RR at work

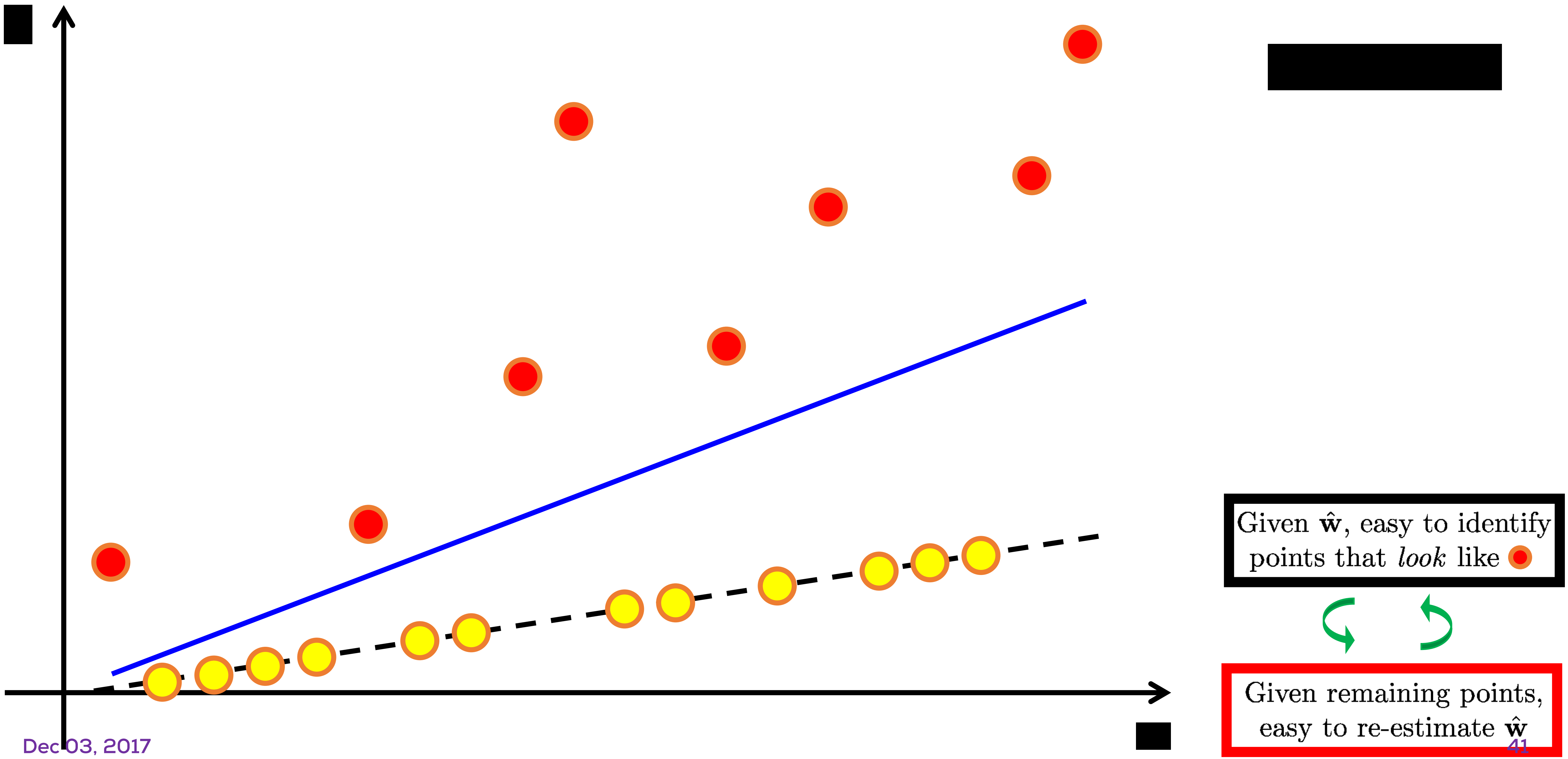


[redacted]

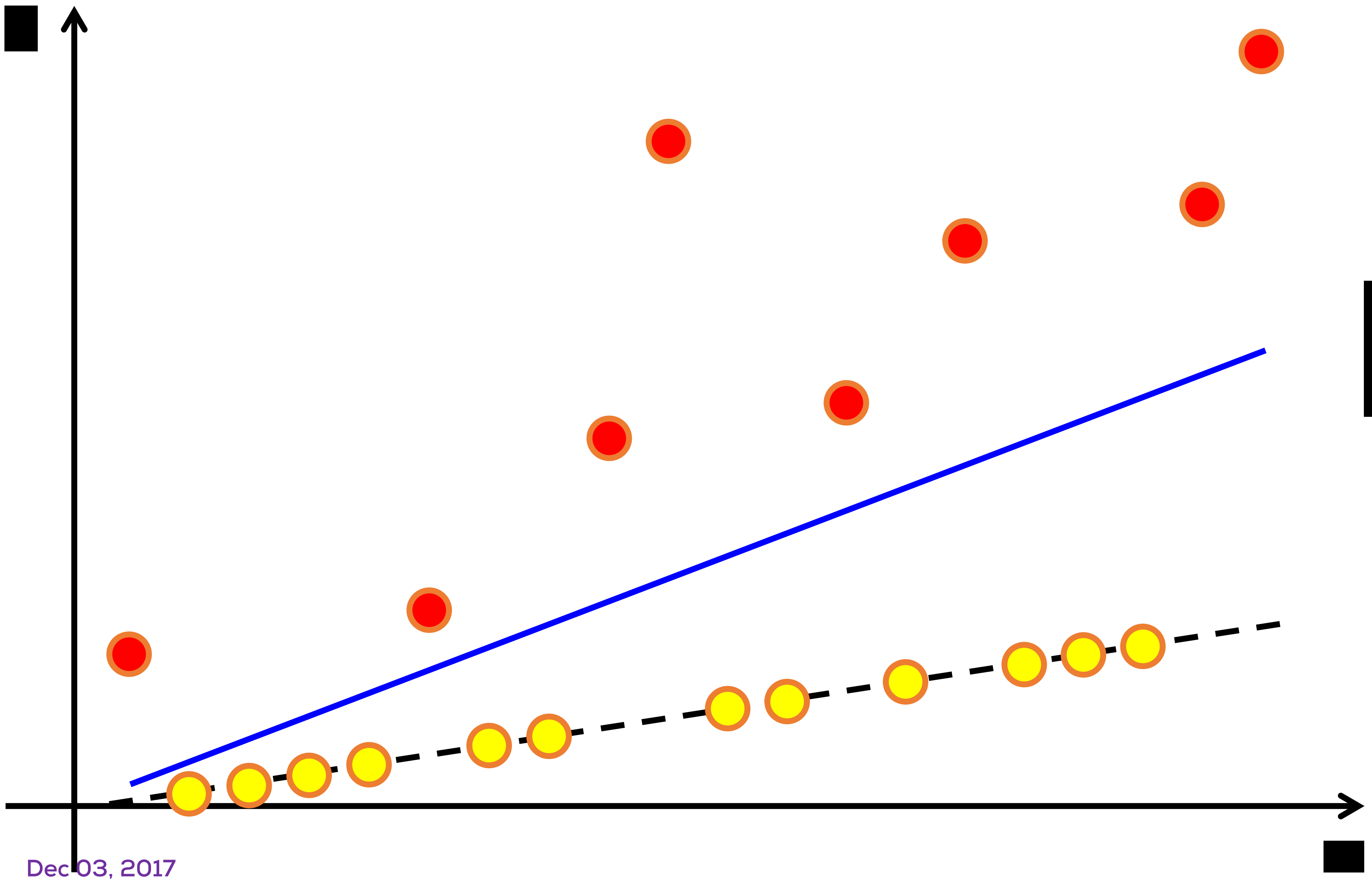
Given $\hat{\mathbf{w}}$, easy to identify points that *look* like ●

Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

AM-RR at work


[redacted]
[redacted]
Given $\hat{\mathbf{w}}$, easy to identify points that *look* like
Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$
[redacted]

AM-RR at work



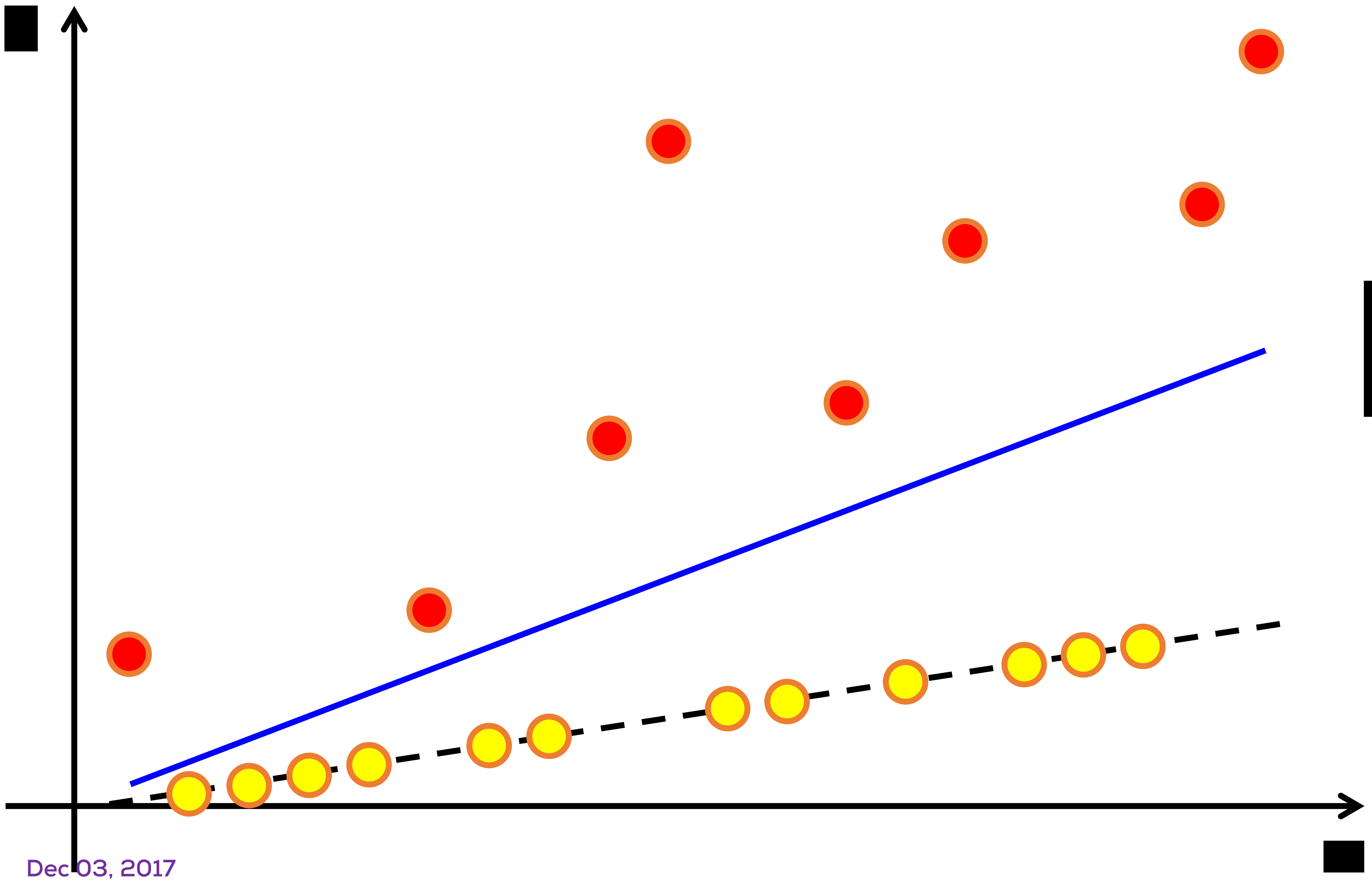
[redacted]

[redacted]

Given $\hat{\mathbf{w}}$, easy to identify points that *look* like ●

Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

AM-RR at work



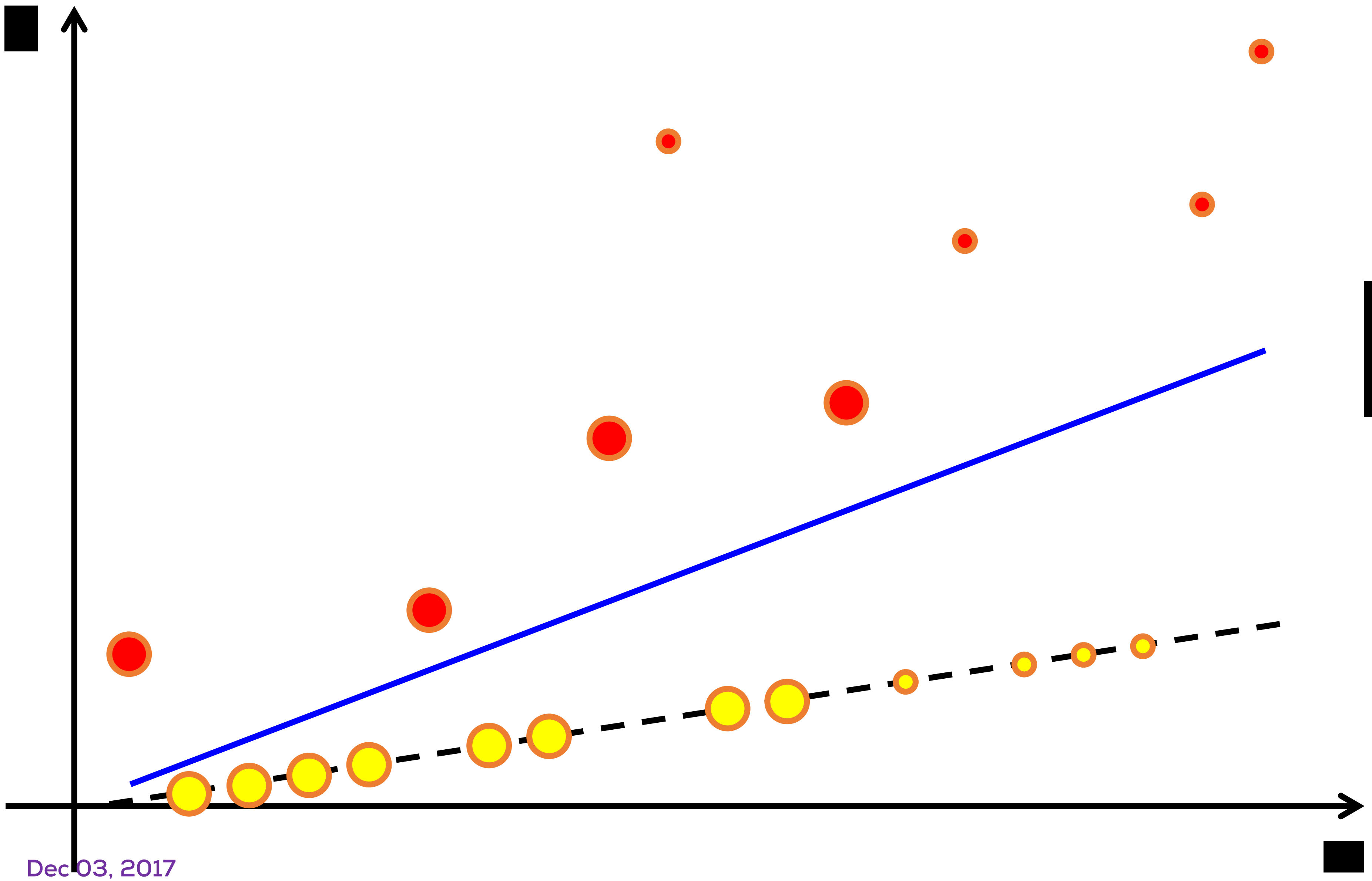
[redacted]

[redacted]

Given $\hat{\mathbf{w}}$, easy to identify points that *look* like ●

Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

AM-RR at work



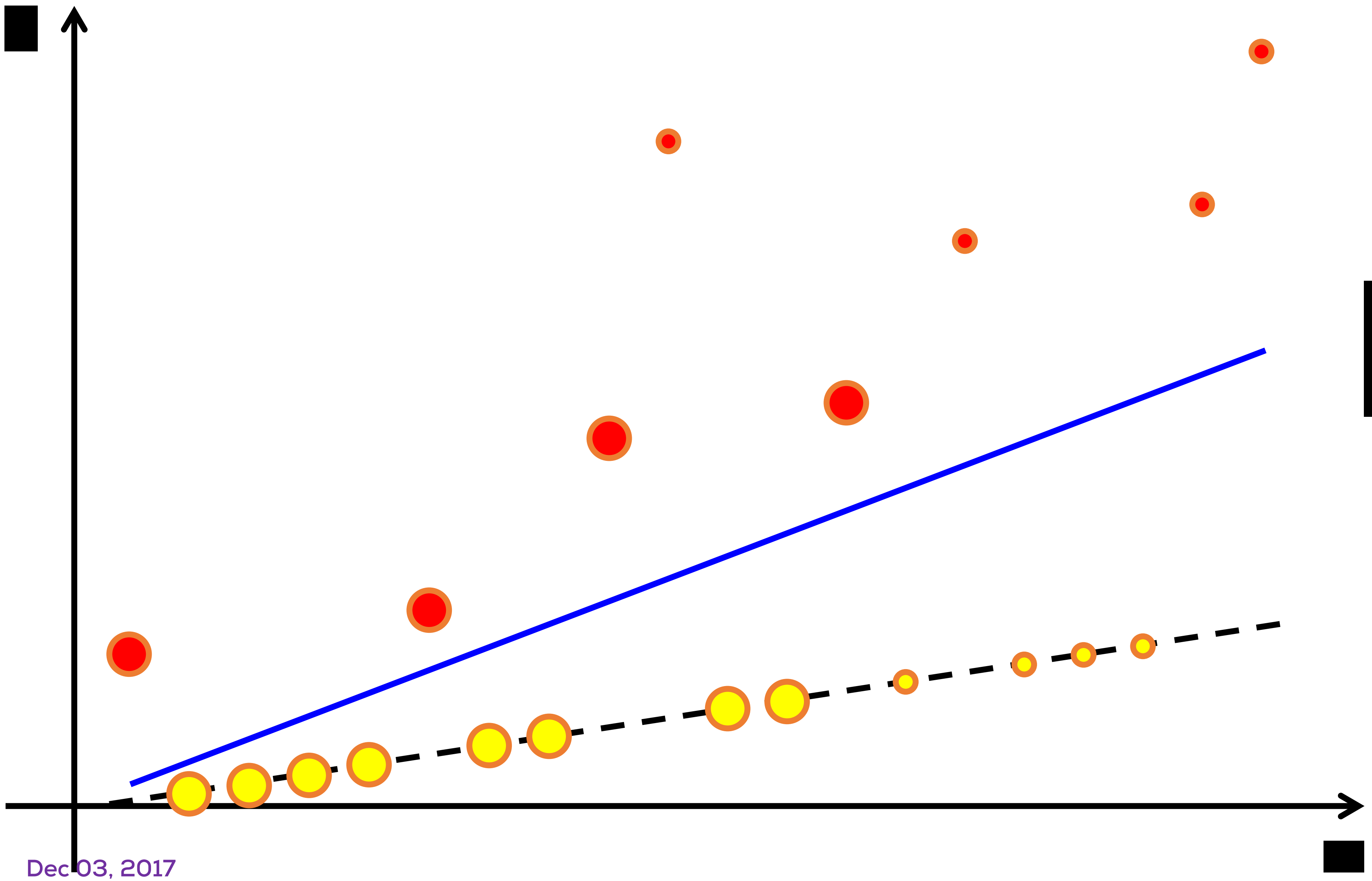
[redacted]

[redacted]

Given $\hat{\mathbf{w}}$, easy to identify points that *look* like ●

Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

AM-RR at work



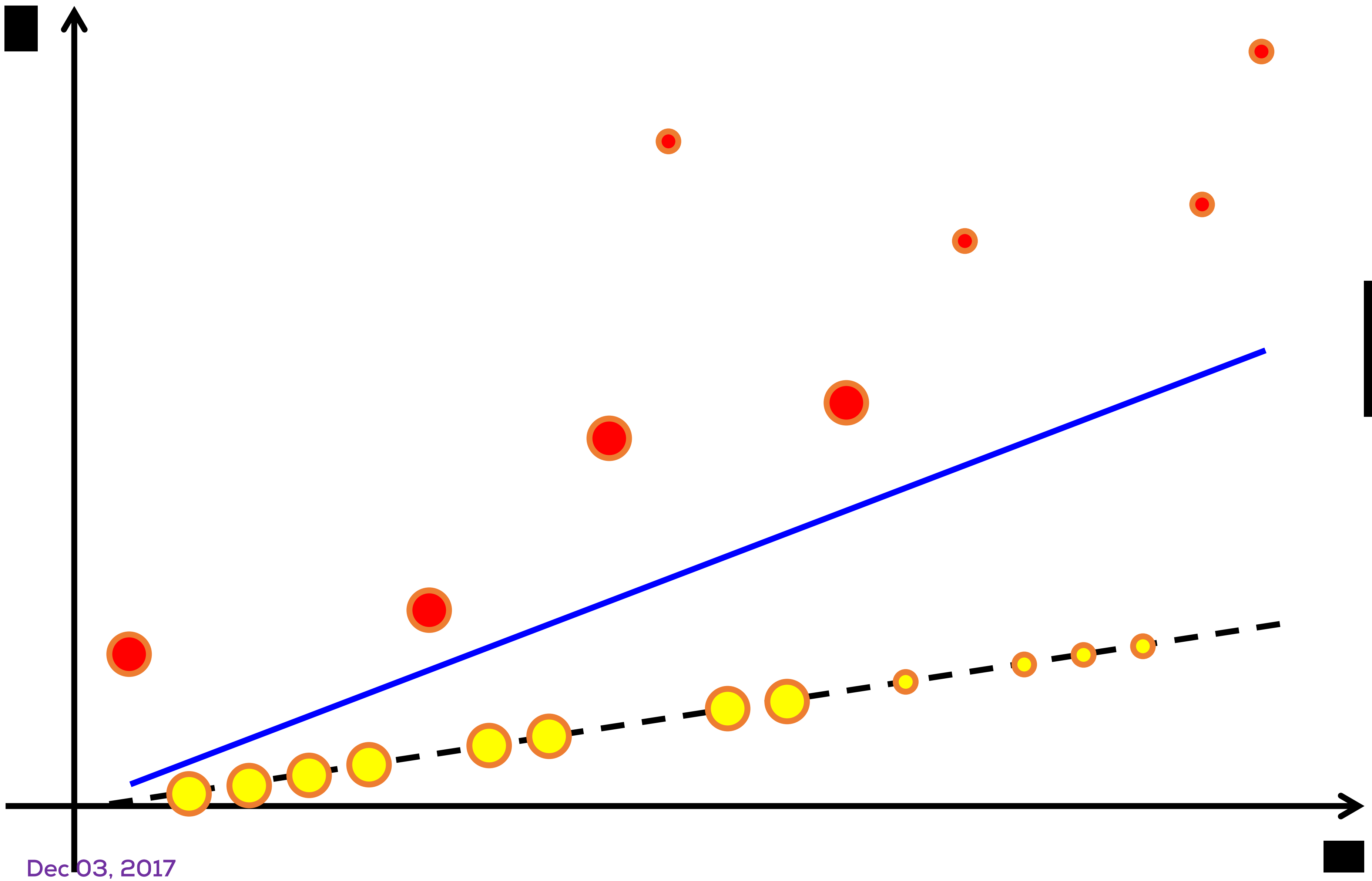
[redacted]

[redacted]

Given $\hat{\mathbf{w}}$, easy to identify points that *look* like ●

Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

AM-RR at work



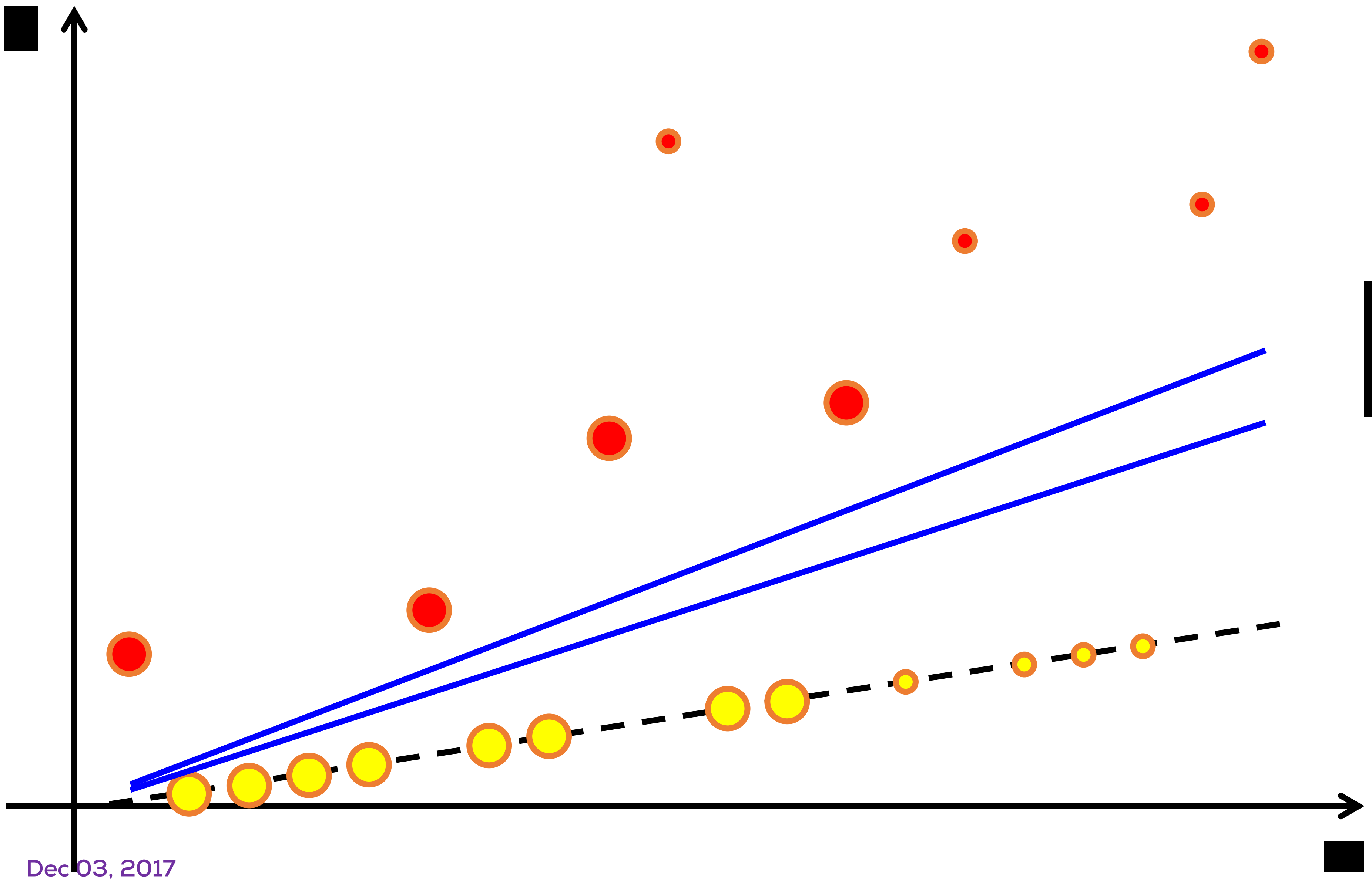
[redacted]

[redacted] $\hat{\mathbf{w}}$

Given $\hat{\mathbf{w}}$, easy to identify points that *look* like ●

Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

AM-RR at work



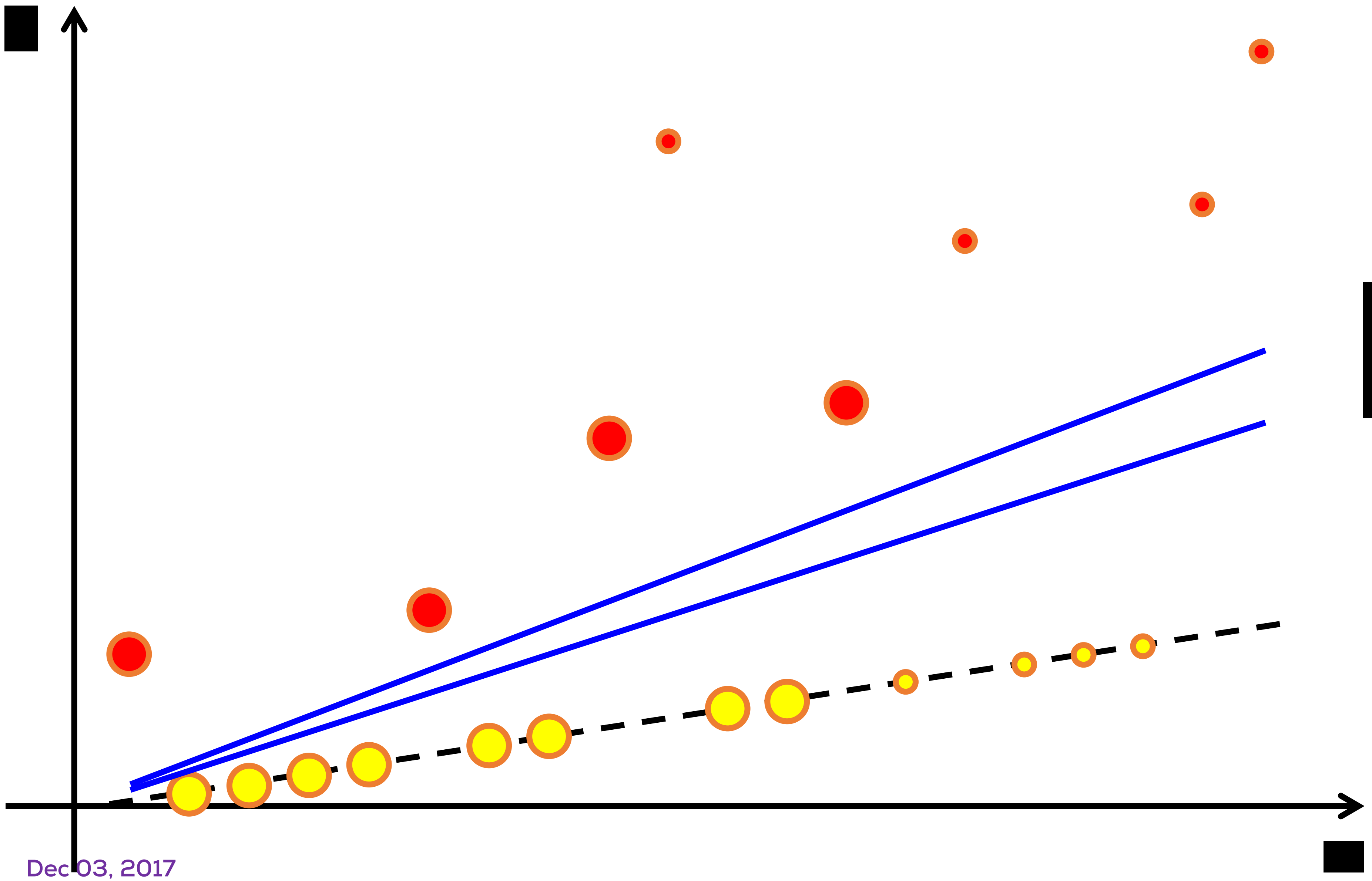
[redacted]

[redacted]

Given $\hat{\mathbf{w}}$, easy to identify points that *look* like ●

Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

AM-RR at work



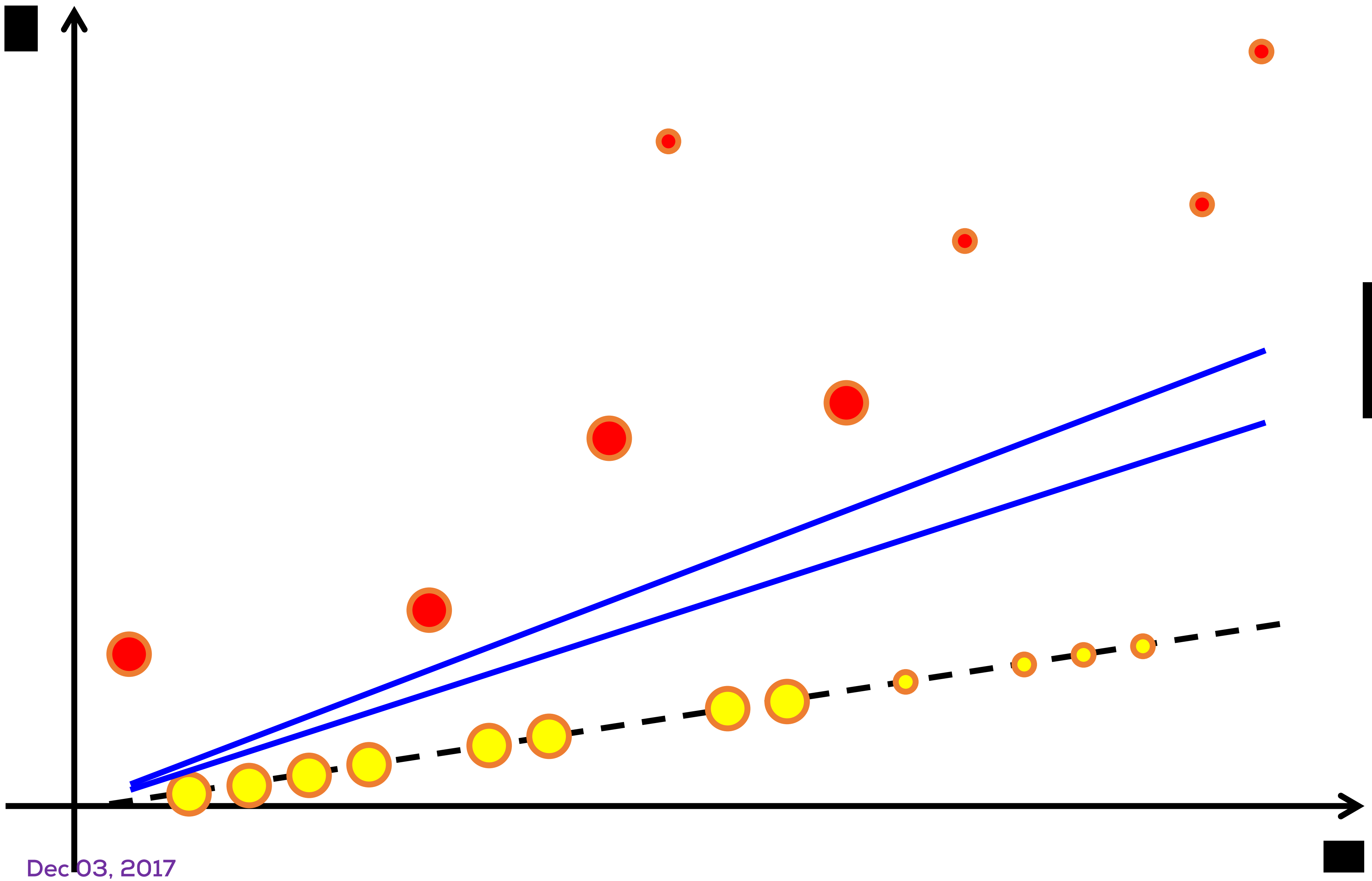
[redacted]

[redacted]

Given $\hat{\mathbf{w}}$, easy to identify points that *look* like ●

Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

AM-RR at work



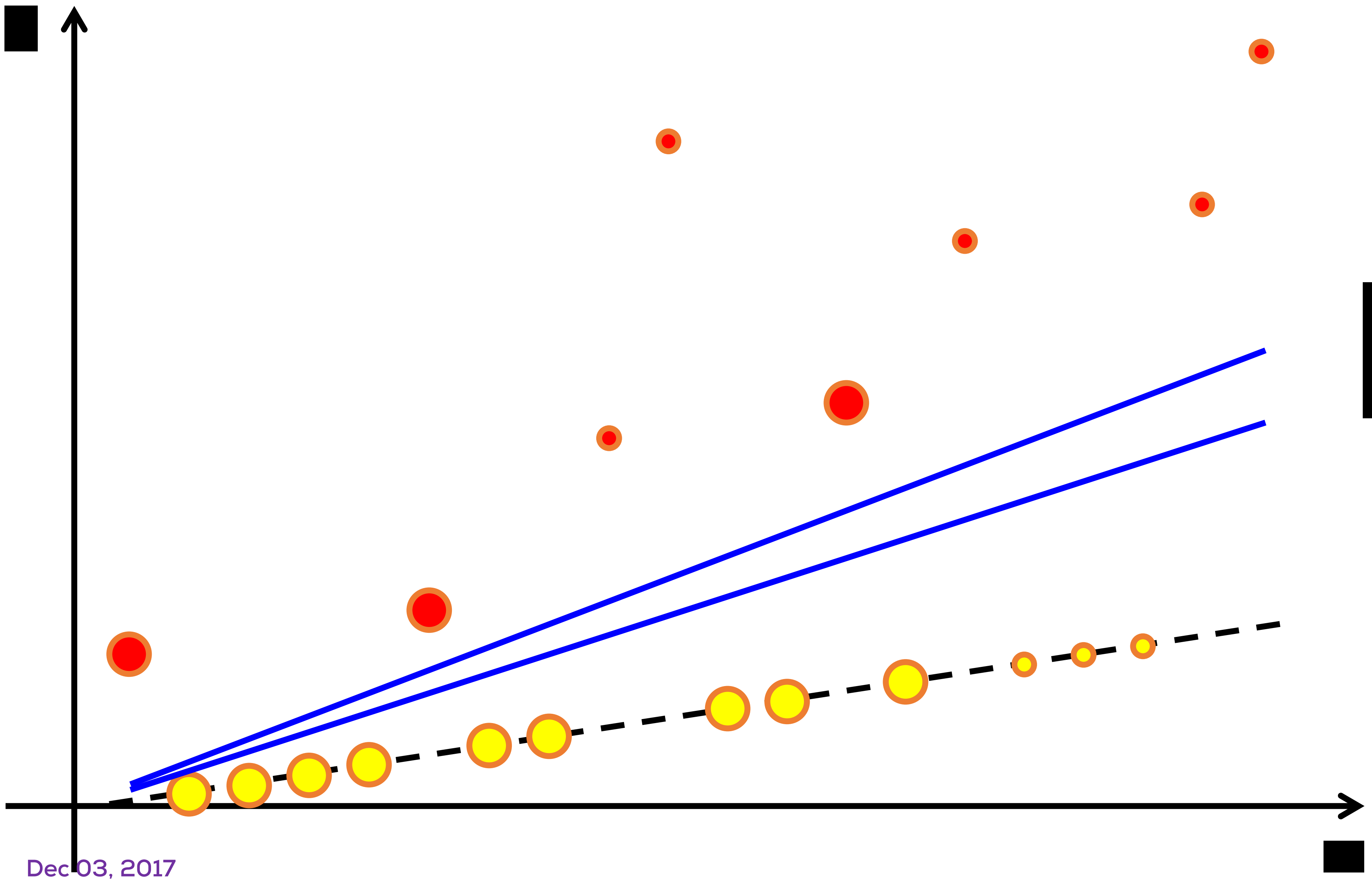
[redacted]

[redacted]

Given $\hat{\mathbf{w}}$, easy to identify points that *look* like ●

Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

AM-RR at work



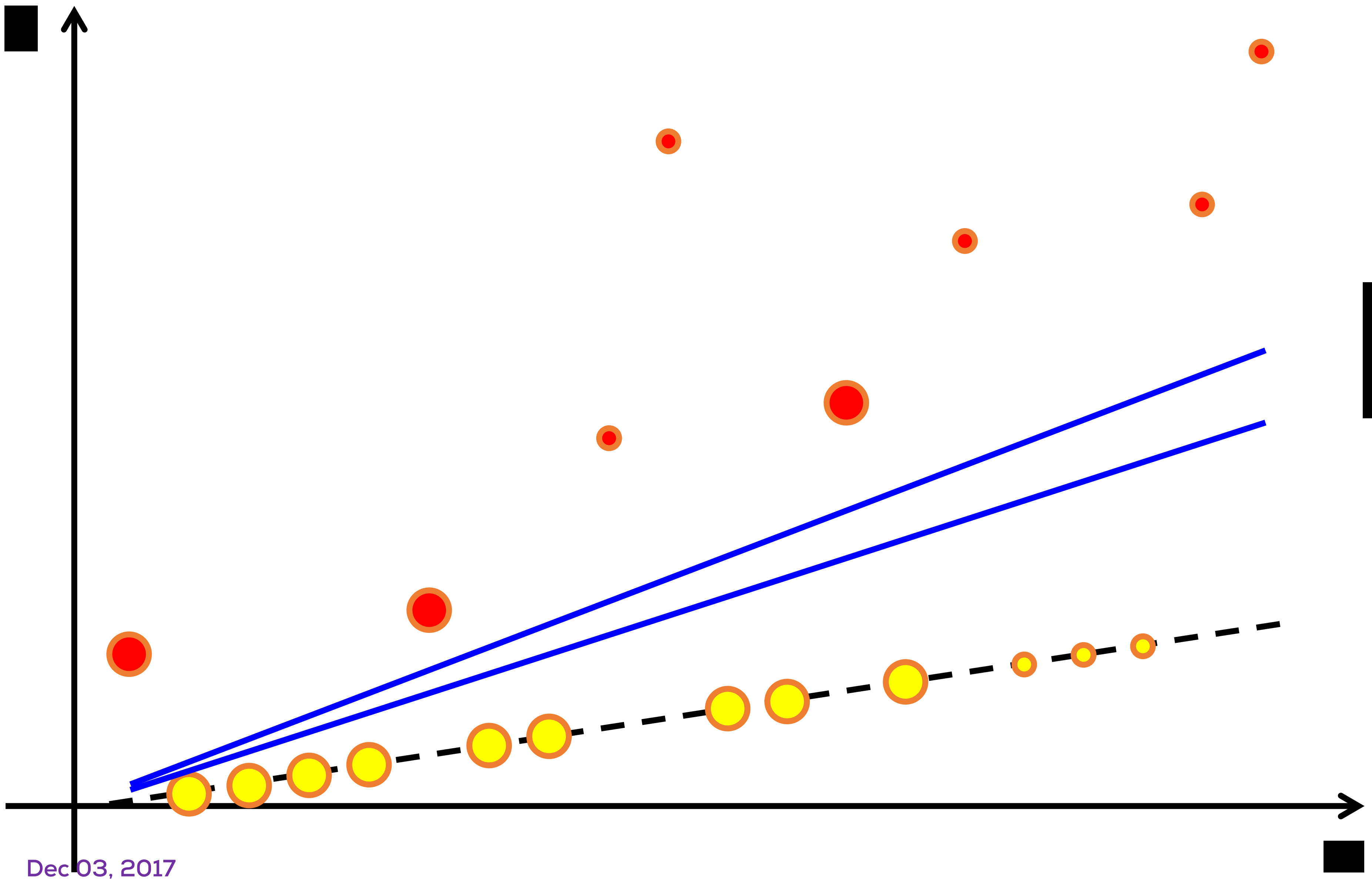
[redacted]

[redacted]

Given $\hat{\mathbf{w}}$, easy to identify points that *look* like ●

Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

AM-RR at work



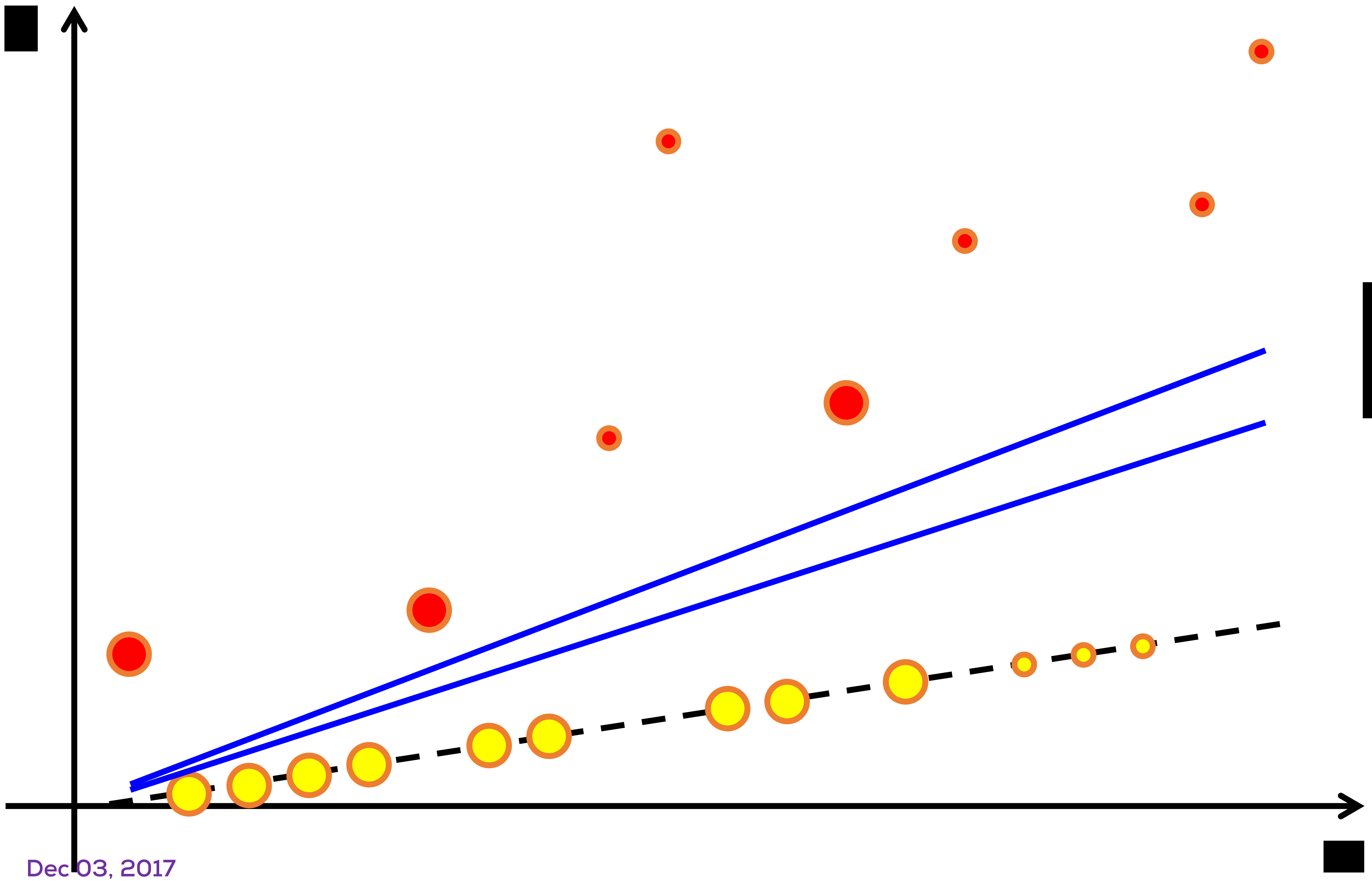
[redacted]

[redacted] $\hat{\mathbf{w}}$

Given $\hat{\mathbf{w}}$, easy to identify points that *look* like ●

Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

AM-RR at work



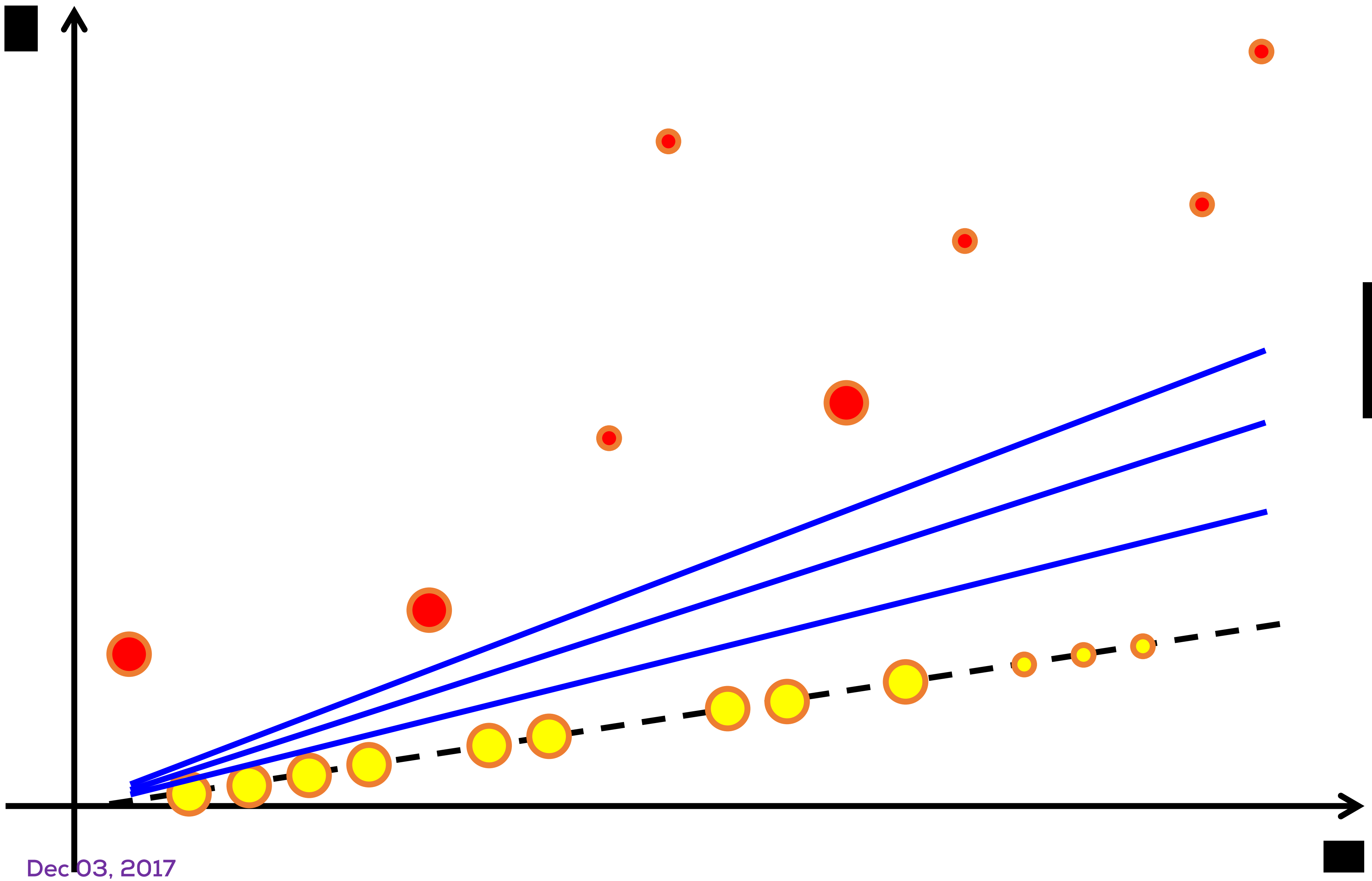
[redacted]

[redacted]

Given $\hat{\mathbf{w}}$, easy to identify points that *look* like ●

Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

AM-RR at work



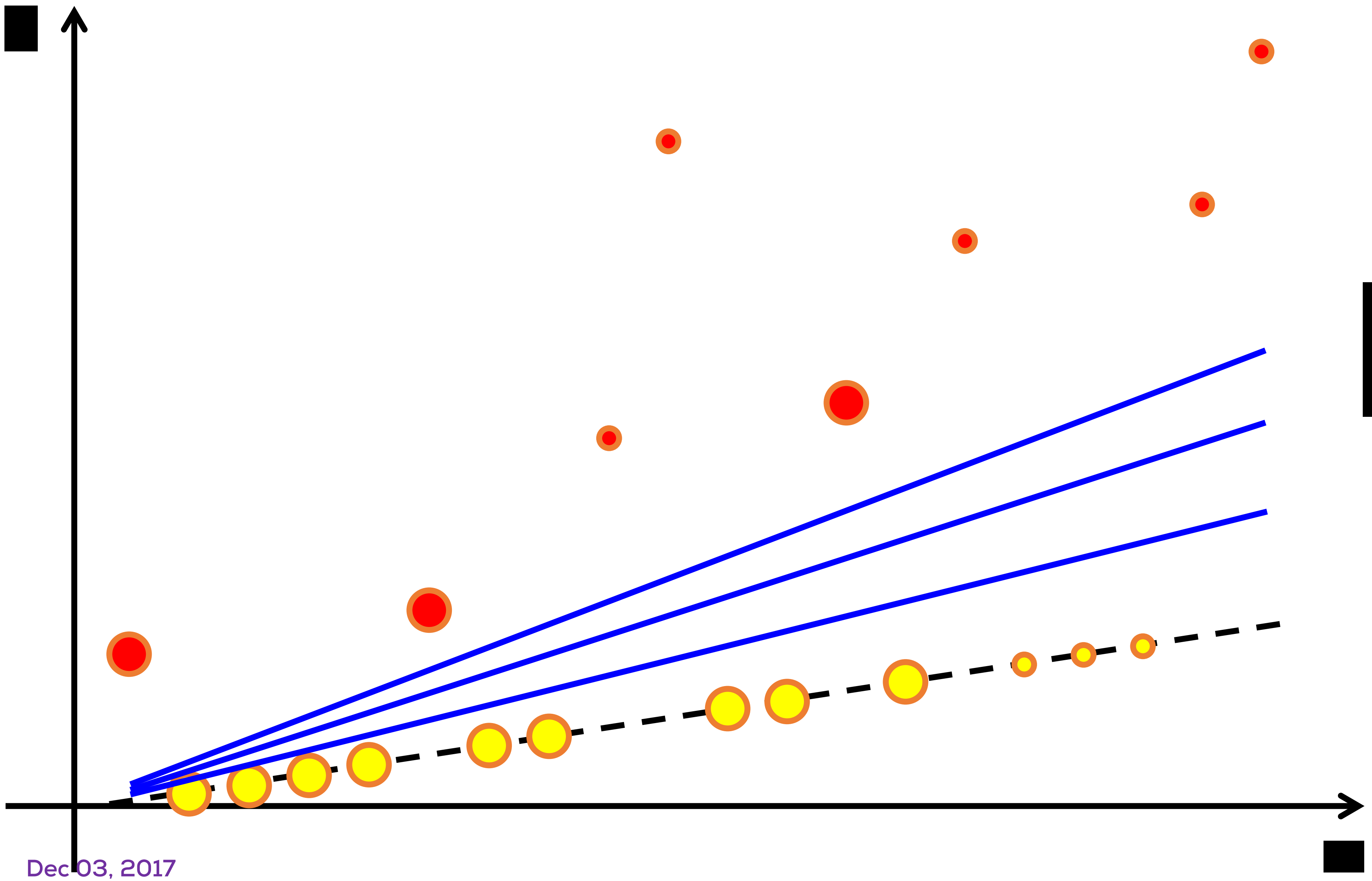
[redacted]

[redacted] $\hat{\mathbf{w}}$

Given $\hat{\mathbf{w}}$, easy to identify points that *look* like ●

Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

AM-RR at work



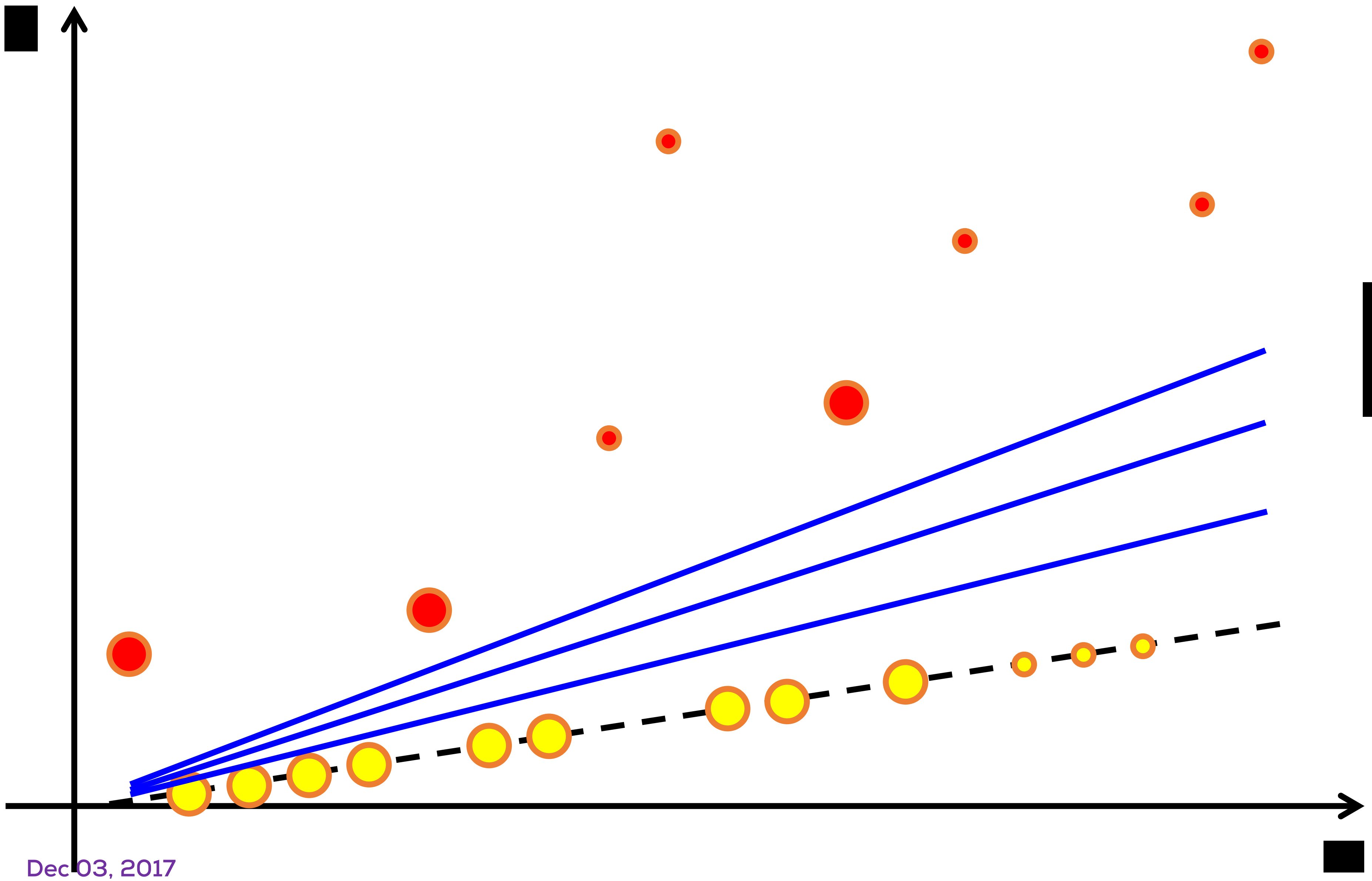
[redacted]

[redacted]

Given $\hat{\mathbf{w}}$, easy to identify points that *look* like ●

Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

AM-RR at work



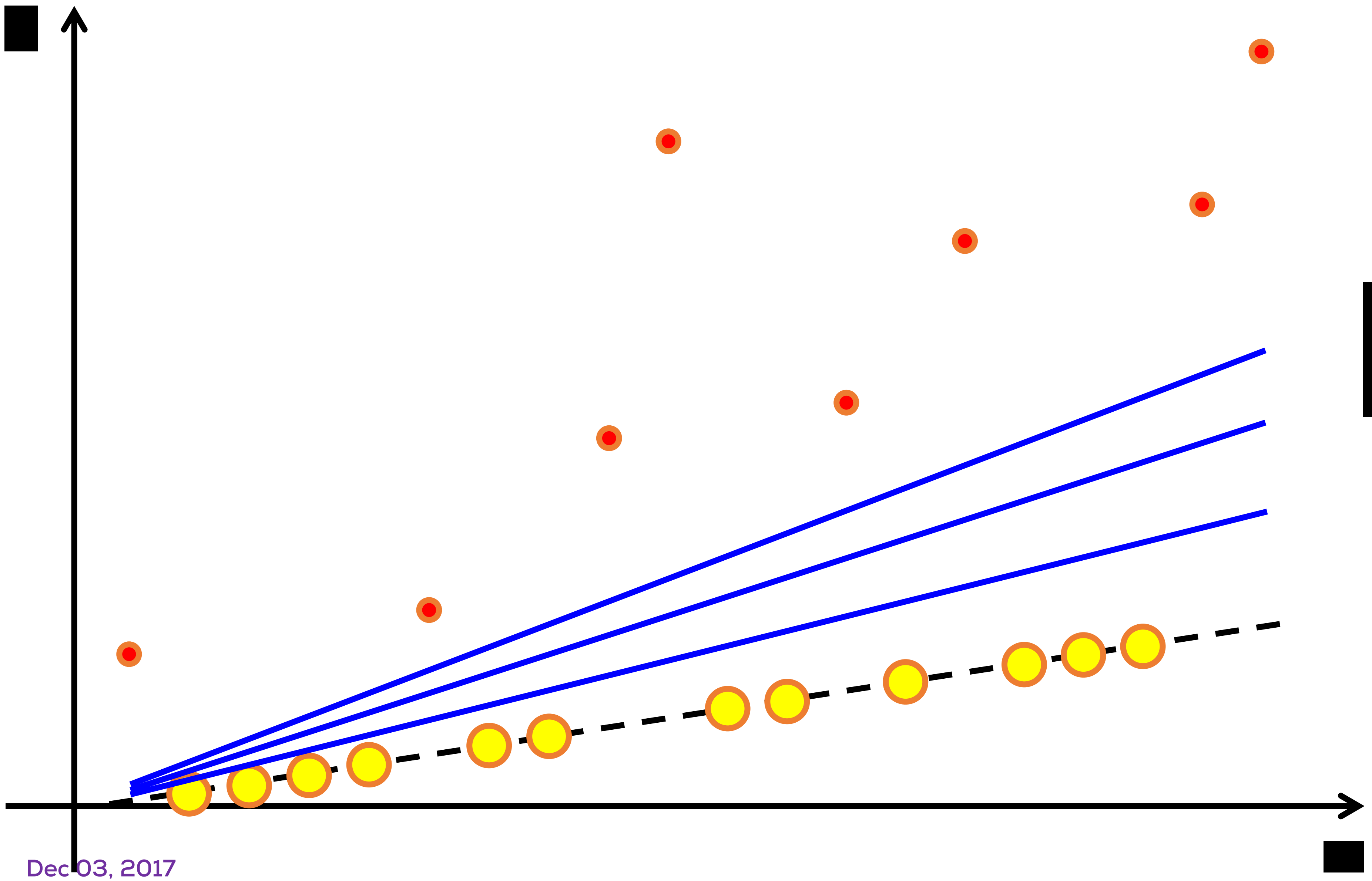
[redacted]

[redacted]

Given $\hat{\mathbf{w}}$, easy to identify points that *look* like ●

Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

AM-RR at work



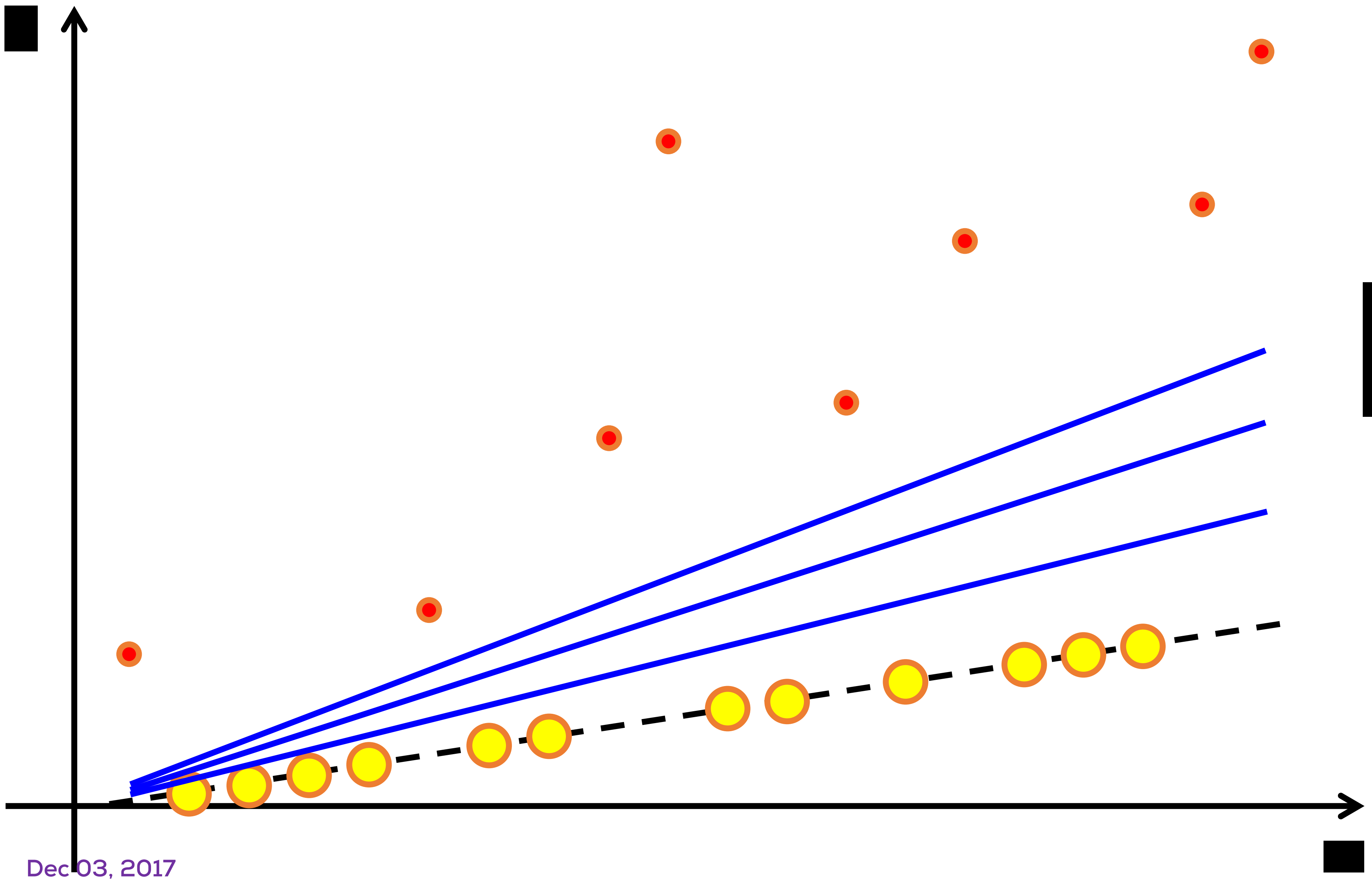
[redacted]

[redacted]

Given $\hat{\mathbf{w}}$, easy to identify points that *look* like ●

Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

AM-RR at work



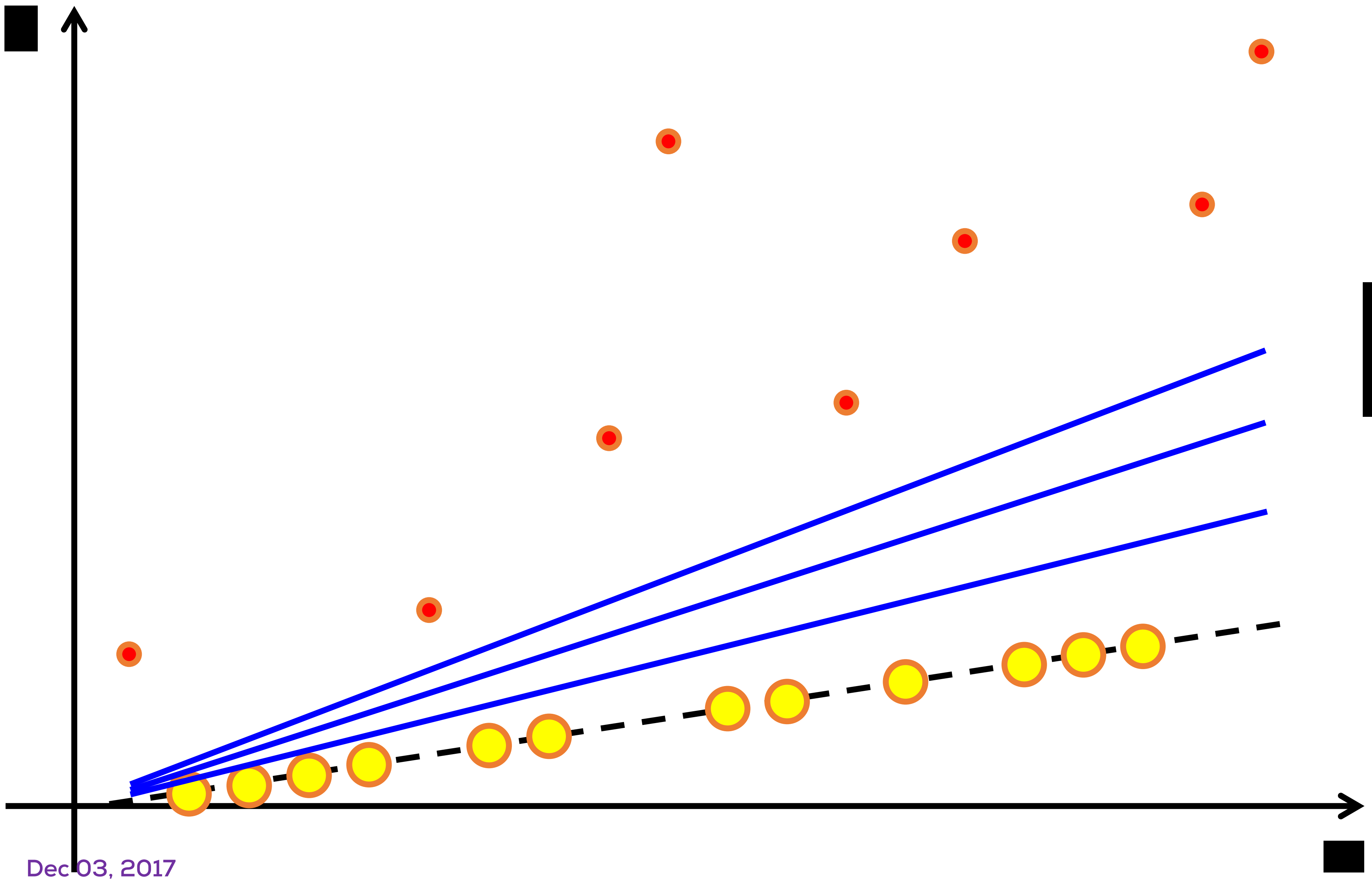
[redacted]

[redacted]

Given $\hat{\mathbf{w}}$, easy to identify points that *look* like

Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

AM-RR at work



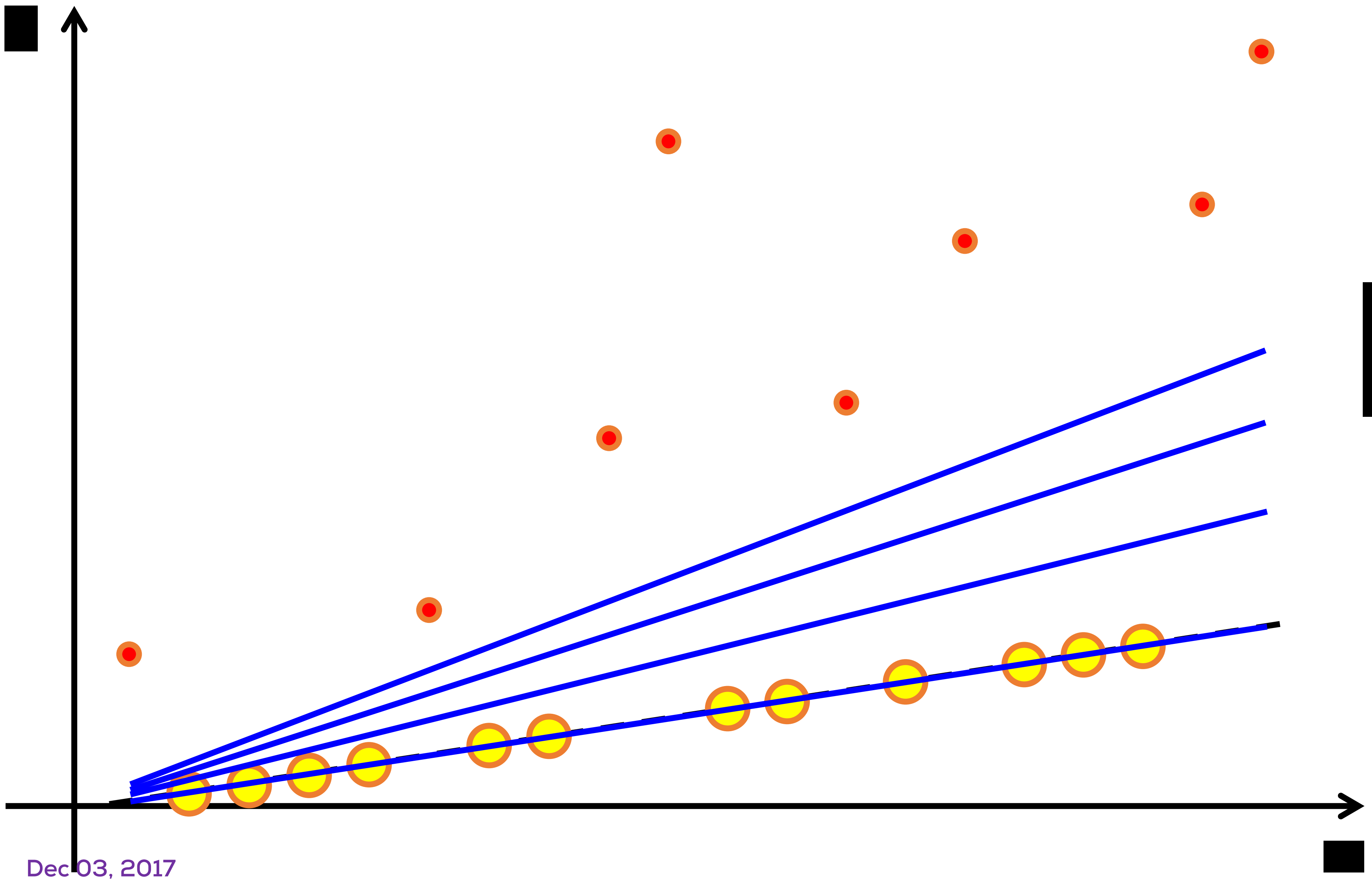
[redacted]

[redacted]

Given $\hat{\mathbf{w}}$, easy to identify points that *look* like ●

Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

AM-RR at work



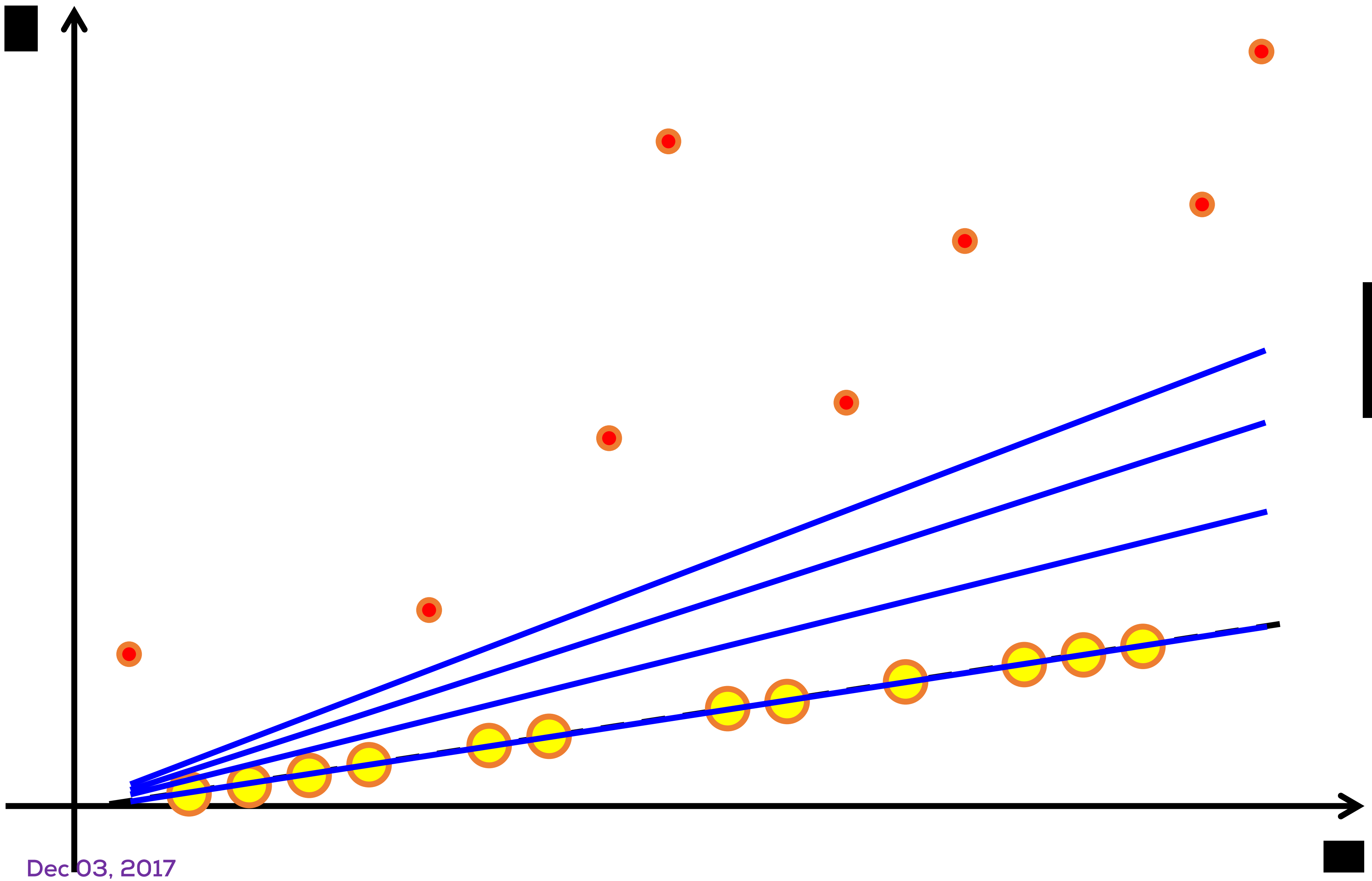
[redacted]

[redacted]

Given $\hat{\mathbf{w}}$, easy to identify points that *look* like ●

Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

AM-RR at work



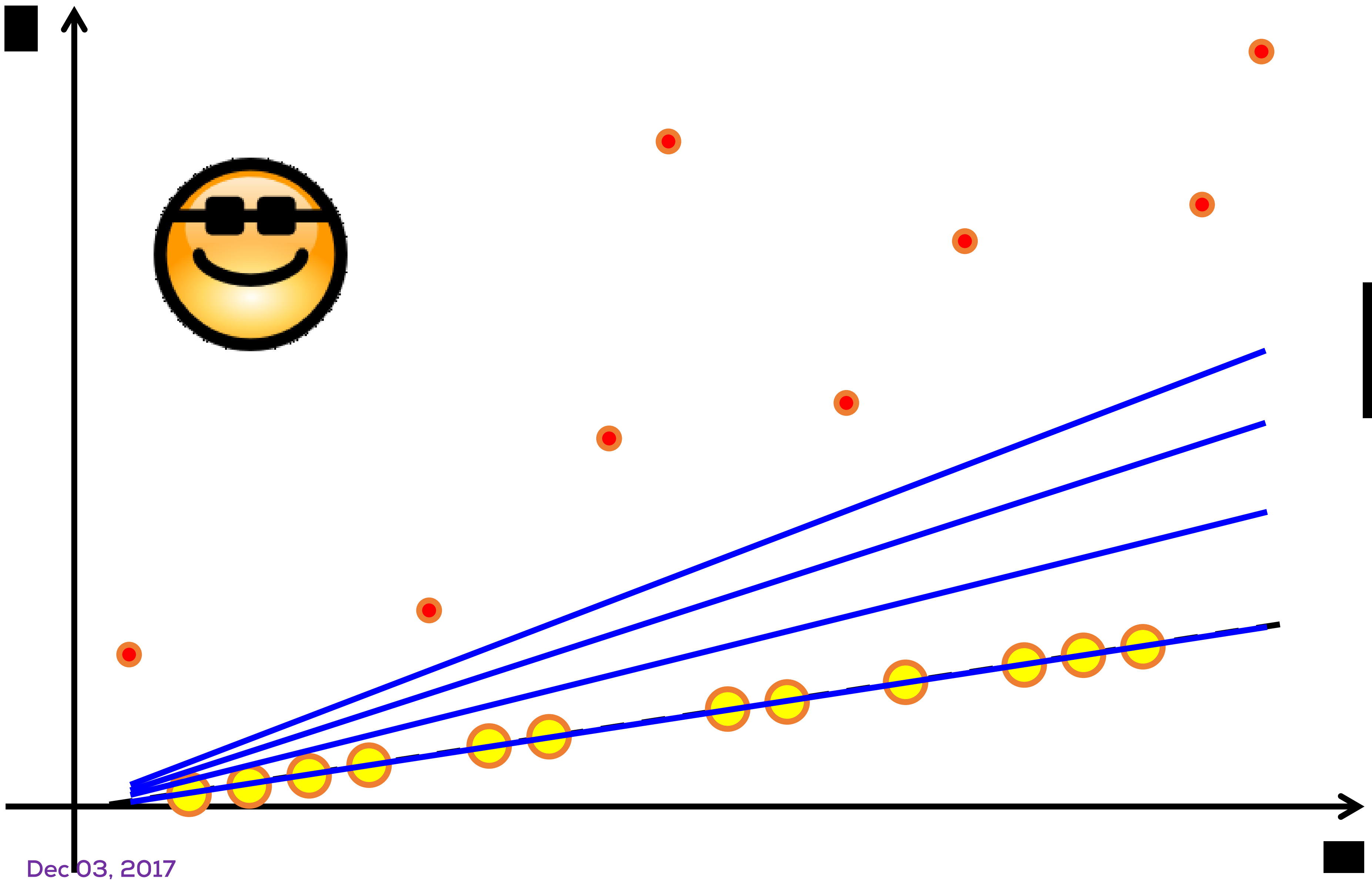
[redacted]

[redacted]

Given $\hat{\mathbf{w}}$, easy to identify points that *look* like ●

Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

AM-RR at work



[redacted]

[redacted]

Given $\hat{\mathbf{w}}$, easy to identify points that *look* like ●

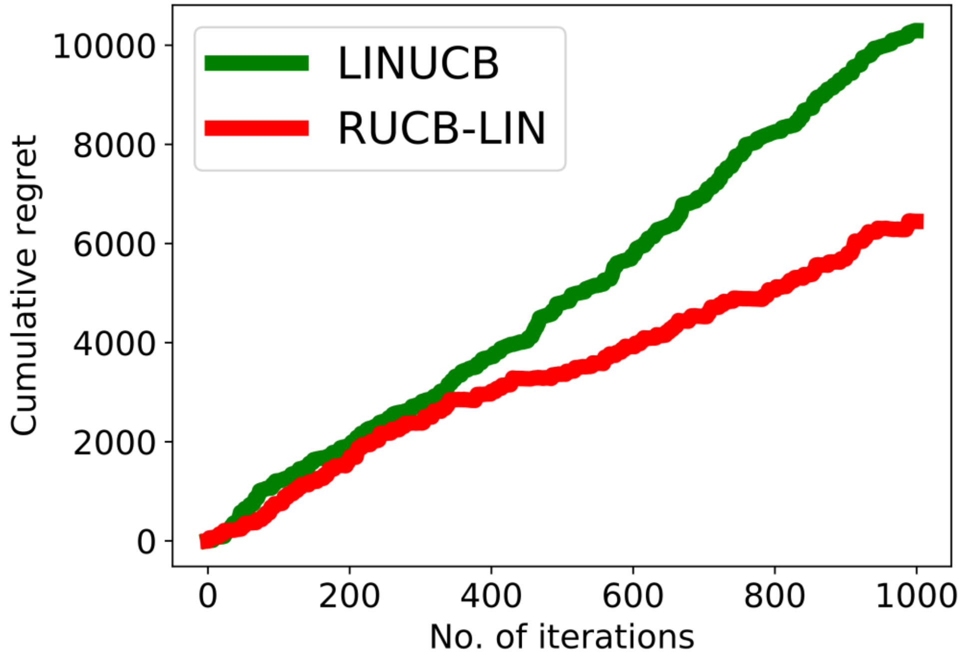
Given remaining points, easy to re-estimate $\hat{\mathbf{w}}$

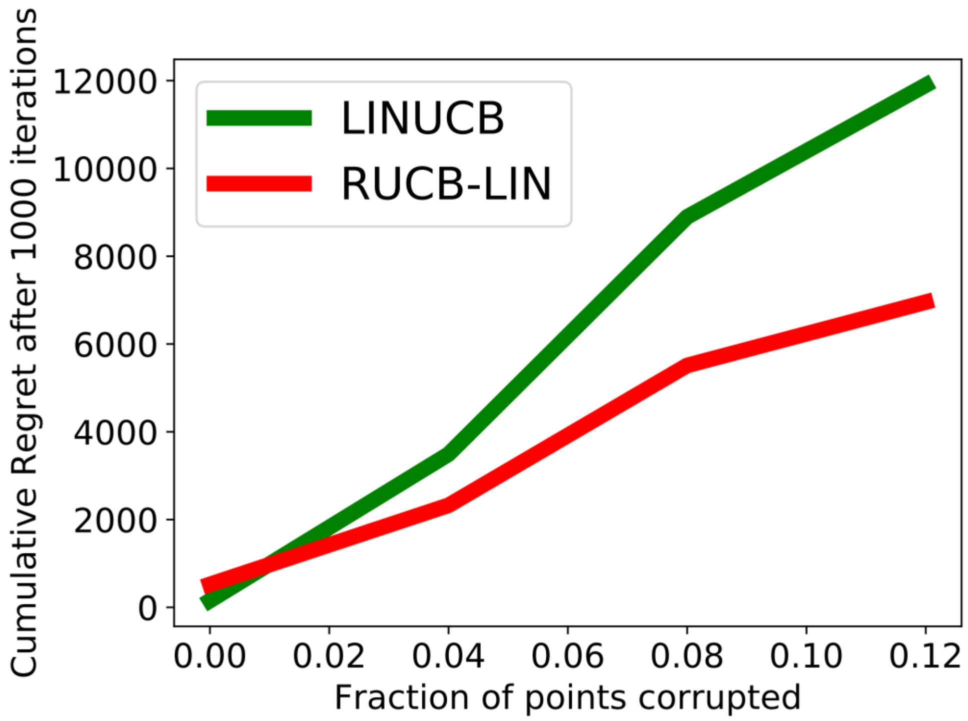
AM-RR at work

ORIGINAL | DISTORTED | OLS | AM-RR

Bhatia et al, 2015

AM-RR at work





- The y-axis plots the "regret" of the algorithms.
- The regret of an algorithm informally captures the amount of "lost opportunities" due to recommending items user doesn't like
- AM-RR based recommendations incur substantially less regret

Feel free to doze off ☺

Convergence proofs ahead!

Why AM-RR works?

- Some presuppositions are necessary
- It had better be possible to uniquely identify $\mathbf{w}^*$ using data in S^*
- This requires X_{S^*} to be well-conditioned
- To simplify things, assume all sets S of size $n-k$ well conditioned
- This can be ensured if for some $c > 0$, every $\mathbf{v} \in \mathbb{R}^d$

$$\left\|X_S^\top \mathbf{v}\right\|_2^2 \geq c \cdot \|\mathbf{v}\|_2^2$$

- We will assume the above holds with $c = \alpha \cdot (n-k)$
- This holds w.h.p. for all S if X is sampled from sub-Gaussian dist.
- Note that since all covariates are unit norm, $\left\|\mathbf{x}^i\right\|_2 \leq 1$, also have

$$\left\|X_S^\top \mathbf{v}\right\|_2^2 \leq |S| \cdot \|\mathbf{v}\|_2^2, \text{ for all } S \subseteq [n]$$

The Proof

- AM-RR solves least squares on the active set S^t hence
$$\mathbf{w}^{t+1} = C_{S_t}^{-1} X_{S_t} \mathbf{y}_{S_t}$$
- However $\mathbf{y}_{S_t} = X_{S_t}^\top \mathbf{w}^* + \mathbf{b}_{S_t}^*$ which gives us
$$\mathbf{w}^{t+1} = \mathbf{w}^* + C_{S_t}^{-1} X_{S_t} \mathbf{b}_{S_t}^*$$
- This gives us the residuals as
$$\mathbf{r}^{t+1} = \mathbf{y} - X^\top \mathbf{w}^{t+1} = \mathbf{b}^* + X C_{S^t}^{-1} X_{S^t} \mathbf{b}_{S^t}^*$$
- AM-RR chooses S^{t+1} to be the $n-k$ points with least residual, so
$$\left\| r_{S^{t+1}}^{t+1} \right\|_2^2 \leq \left\| r_{S^*}^{t+1} \right\|_2^2$$
- Notice that since $\mathbf{r}^{t+1}$ is simply $\mathbf{b}^*$ plus an error term, once error goes down, my active set will only contain clean points!

Nice! $\mathbf{w}^{t+1}$ is $\mathbf{w}^*$ plus an error term

Nice! $\mathbf{r}^{t+1}$ is $\mathbf{b}^*$ plus an error term

The Proof

- Elementary manipulations and applying well conditioned-ness

$$\left\|\mathbf{b}^*_{S^{t+1}}\right\|_2^2 \leq \frac{k^2}{\alpha^2(n-k)^2} \cdot \left\|\mathbf{b}^*_{S^t}\right\|_2^2 + \frac{2k}{\alpha(n-k)} \cdot \left\|\mathbf{b}^*_{S^t}\right\|_2^2 \cdot \left\|\mathbf{b}^*_{S^t}\right\|_2^2$$

- Solving this quadratic equation gives us

$$\left\|\mathbf{b}^*_{S^{t+1}}\right\|_2 \leq \frac{\left(\sqrt{2}+1\right)k}{\alpha(n-k)} \cdot \left\|\mathbf{b}^*_{S^t}\right\|_2$$

- Thus, if $k \leq \frac{\alpha}{3+\alpha} \cdot n$, then after $t \geq \log \frac{\|\mathbf{b}^*\|_2}{\epsilon}$ we get $\|\mathbf{w}^t - \mathbf{w}^*\|_2 \leq \epsilon$
- A better way to rewrite the above is $k \leq \frac{n}{3\kappa+1}$
- The quantity $\kappa = \frac{1}{\alpha}$ captures the *condition number* of the problem

A Generalized AM-RR

Weaker result for sake of clarity ☺

- Loss function: $\ell(\mathbf{w}; y^i, \mathbf{x}^i)$ with certain nice properties
  - Positivity: $\ell(\mathbf{w}; y^i, \mathbf{x}^i) \geq 0$
  - Realizability: $\ell(\mathbf{w}^*; y^i, \mathbf{x}^i) = 0$ for all $i \in S^*$
  - Normalization: $\left|\ell(\mathbf{w}^1; y^i, \mathbf{x}^i) - \ell(\mathbf{w}^2; y^i, \mathbf{x}^i)\right| \leq \frac{\beta}{2} \cdot \|\mathbf{w}^1 - \mathbf{w}^2\|_2^2$
  - Well-conditioned-ness:

    $\ell(\mathbf{w}^1; y^i, \mathbf{x}^i) \geq \ell(\mathbf{w}^2; y^i, \mathbf{x}^i) + \langle \nabla\ell(\mathbf{w}^2; y^i, \mathbf{x}^i), \mathbf{w}^1 - \mathbf{w}^2 \rangle + \frac{\alpha}{2} \cdot \|\mathbf{w}^1 - \mathbf{w}^2\|_2^2$

Plays same role as assump. $\left\|\mathbf{x}^i\right\|_2 \leq 1$

Strong Convexity

- Denote $f(\mathbf{w}, S) = \sum_{i \in S} \ell(\mathbf{w}; y^i, \mathbf{x}^i)$
- Actually, weaker requirement needed: for all $S \subseteq [n]$

$$f(\mathbf{w}^1, S) \geq f(\mathbf{w}^2, S) + \langle \nabla f(\mathbf{w}^2, S), \mathbf{w}^1 - \mathbf{w}^2 \rangle + \frac{\alpha|S|}{2} \cdot \|\mathbf{w}^1 - \mathbf{w}^2\|_2^2$$

$$|f(\mathbf{w}^1, S) - f(\mathbf{w}^2, S)| \leq \frac{\beta|S|}{2} \cdot \|\mathbf{w}^1 - \mathbf{w}^2\|_2^2$$

A Generalized AM-RR

AM-RR-gen

1. Data $X \in \mathbb{R}^{d \times n}, \mathbf{y} \in \mathbb{R}^n$, # bad pts k
2. Initialize $S^1 \leftarrow [1 : n - k]$
3. For $t = 1,2, \dots, T$

$$\mathbf{w}^{t+1} = \arg\min_{\mathbf{w}} f(\mathbf{w}, S^t)$$

$$S^{t+1} = \arg\min_{|S| = n-k} f(\mathbf{w}^{t+1}, S)$$

4. Repeat until convergence

Since f is a convex function, it is usually easy to solve this

Find the $n - k$ points with the least loss in terms of $\ell(\mathbf{w}; y^i, \mathbf{x}^i)$

Why AM-RR-gen works!

- Since $\mathbf{w}^{t+1}$ minimizes $f(\mathbf{w}, S^t)$, we must have $\nabla f(\mathbf{w}^{t+1}, S^t) = \mathbf{0}$
- Applying well conditioned-ness then gives us

$$\|\mathbf{w}^* - \mathbf{w}^{t+1}\|_2^2 \leq \frac{2}{\alpha(n-k)}\left(f(\mathbf{w}^*, S^t) - f(\mathbf{w}^{t+1}, S^t)\right) \leq \frac{2}{\alpha(n-k)} f(\mathbf{w}^*, S^t)$$

- Since S^{t+1} minimizes $f(\mathbf{w}^{t+1}, S)$, we have $f(\mathbf{w}^{t+1}, S^{t+1}) \leq f(\mathbf{w}^{t+1}, S^*)$

$$f(\mathbf{w}^{t+1}, S^{t+1} \backslash S^*) \leq f(\mathbf{w}^{t+1}, S^* \backslash S^{t+1})$$

- Since $|S^{t+1} \backslash S^*| \leq k$ and $|S^* \backslash S^{t+1}| \leq k$, normalization gives us

$$f(\mathbf{w}^*, S^{t+1}) = f(\mathbf{w}^*, S^{t+1} \backslash S^*) \leq \beta k \cdot \|\mathbf{w}^* - \mathbf{w}^{t+1}\|_2^2$$

- Putting things together gives us $f(\mathbf{w}^*, S^{t+1}) \leq \frac{2\beta k}{\alpha(n-k)} f(\mathbf{w}^*, S^t)$
- For $k \leq \frac{n}{3\kappa+1}$ we get linear convergence ($\kappa = \frac{\beta}{\alpha}$ is condition number)

AM-RR with Kernels!

- Calculations get messy so just a sketch-of-an-algo for now ☺
- Data $(\mathbf{x}^i, y^i)$ and kernel K with associated feature map $\phi: \mathbb{R}^d \to \mathcal{H}$
$$K(x^i, x^j) = \langle \phi(\mathbf{x}^i), \phi(\mathbf{x}^j) \rangle$$
- Model a bit different $y^i = \langle \mathbf{W}, \phi(\mathbf{x}^i) \rangle + \mathbf{b}_i^*$, for some $\mathbf{W} \in \mathcal{H}$
- Can only hope to recover component of $\mathbf{W}$ in $\text{span}(\phi(\mathbf{x}^1), \dots, \phi(\mathbf{x}^n))$
- A simplified algo that uses AM-RR as a black box
    - Receive the data and create new features out of "empirical kernel map"
    - Create $\mathbf{z}^i = (K(\mathbf{x}^i, \mathbf{x}^1), K(\mathbf{x}^i, \mathbf{x}^2), \dots, K(\mathbf{x}^i, \mathbf{x}^n)) \in \mathbb{R}^n$
    - Perform AM-RR with covariates $\mathbf{z}^i$, responses y^i
- Making this bullet-proof needs more work

Please consider waking up

Open problems ahead ☺

Non-toy Problems for relaxed introspection

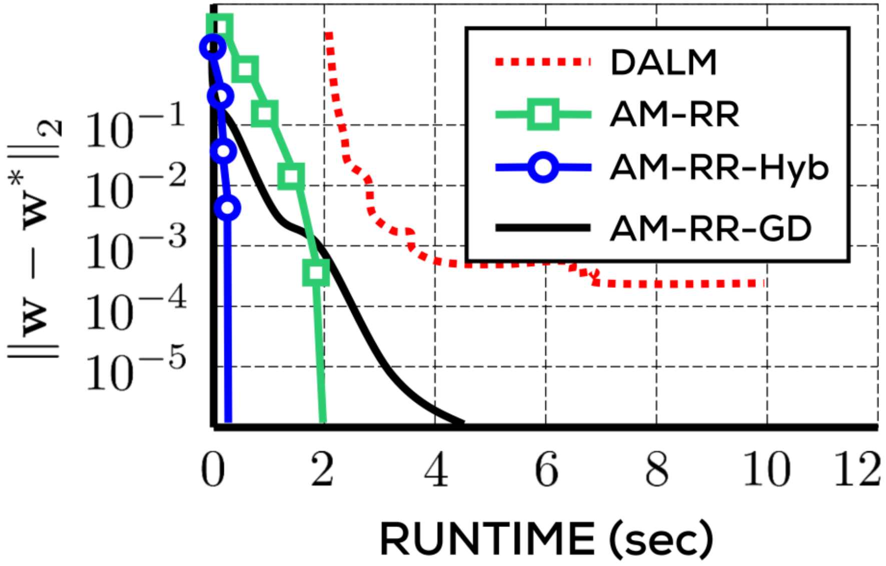
- Presence of dense noise $\mathbf{y} = X\mathbf{w}^* + \boldsymbol{\epsilon} + \mathbf{b}$
  - Corruptions are still sparse $\|\mathbf{b}\|_0 \leq k$
  - Dense noise is Gaussian $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, I)$
  - Can we ensure *consistent recovery* i.e. $\lim_{n \to \infty} \|\hat{\mathbf{w}} - \mathbf{w}^*\|_2 = 0$?
- Adversary Model
  - Fully adaptive: $\mathbf{b}$ chosen with knowledge of $X, \mathbf{w}^*, \boldsymbol{\epsilon}$
  - Fully oblivious: $\mathbf{b}$ chosen without knowledge of $X, \mathbf{w}^*, \boldsymbol{\epsilon}$
  - Partially oblivious: $\mathbf{b}$ chosen with knowledge of $X, \mathbf{w}^*$ or just X
- Breakdown point
  - Can we tolerate up to $k = \frac{n}{2} - 1$ corruptions, $k = n - d \log d$ corruptions?
  - What if adversary is fully oblivious?

Non-toy Problems for relaxed introspection

- Non-linear/structured Models
  - What if the linear model $y \approx \langle \mathbf{w}, \mathbf{x} \rangle$ is not appropriate?
  - Rob. Reg. with sparse models?, kernels?, deep-nets?
- Loss Function
  - The least squares loss function $(y - \langle \mathbf{w}, \mathbf{x} \rangle)^2$ may not suit all applications
  - Robust regression with other loss functions? We already saw an example
- Speed
  - I am solving $\log \frac{1}{\epsilon}$ reg. problems to solve one corrupted reg. problem
  - Can I invoke the least squares solver less frequently?
- Feature corruption
  - What if the features $\mathbf{x}^i$ are corrupted (too)?
  - Can pass the “blame” for corruption onto y^i but does not always work
  - Distributed corruption: each $\mathbf{x}^i$ has only few of d coordinates corrupted

We do have some answers

- We can ensure consistent recovery of $\mathbf{w}^*$ in the presence of dense Gaussian noise and sparse corruptions if adversary is fully oblivious (Bhatia et al, 2017)
- We can handle sparse linear models/kernels for a fully adaptive adversary (but no dense noise) (Bhatia et al, 2015)
- Feature corruptions can be handled (Chen et al, 2013)
- AM-RR can be sped up quite a bit
  - Unless active set S^t stable, do gradient steps
  - May replace with SGD/ConjGD steps
  - In practice, very few least squares calls
  - Extremely rapid execution in practice
- AM-RR can be extended to other losses



We do have some answers

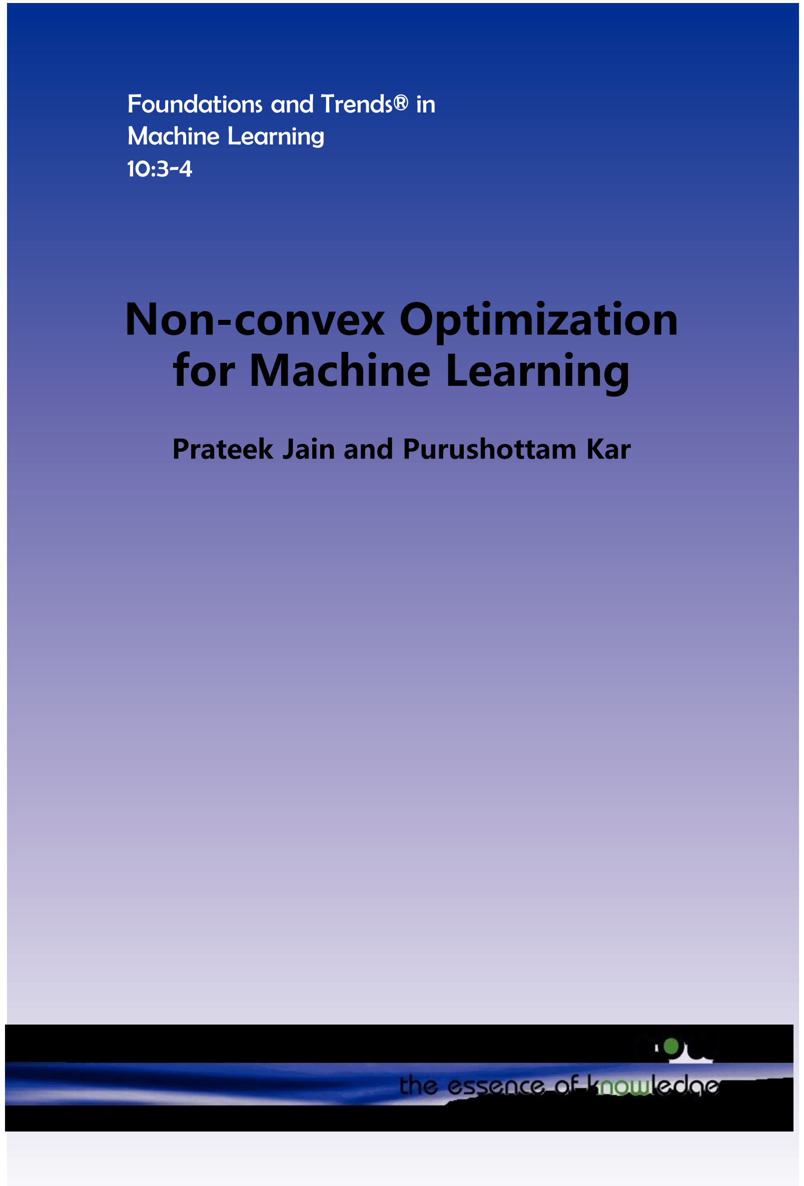
- A "softer" approach also can be shown to work
  - Instead of throwing away points, down-weigh them
  - Points with small residual get up-weighed
  - Perform weighted least-squares estimation (IRLS)
- Our breakdown points are quite bad ☺
  - For simple linear models with no feature corruption, best result $k \approx \frac{n}{100}$
  - For feature corruption even worse $k \approx \frac{n}{d}$
- Biggest missing piece: answering several questions together e.g. consistent recovery with kernel models in the presence of an adaptive adversary and feature corruption?

That's All!

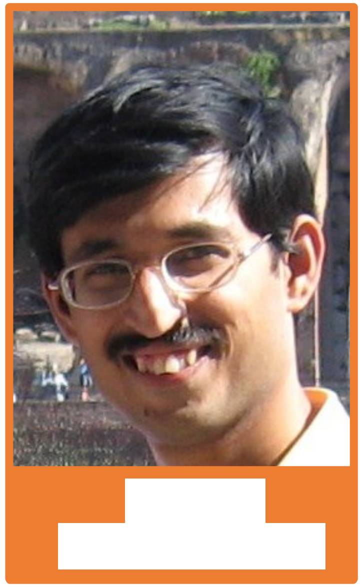
Shameless Ad Alert!

Non-convex Optimization For Machine Learning

- A new monograph!
- Accessible but comprehensive
- Foundations: PGD/SGD, Alt-Min, EM
- Apps: sparse rec., matrix comp., rob. reg.
- 130 references, 50 exercises, 20 figures
- Official publication available from now publishers https://tinyurl.com/ncom-book
- For benefit of students, an arXiv version https://tinyurl.com/ncom-arxiv
  Grateful to now publishers for this!
- Don't Relax!



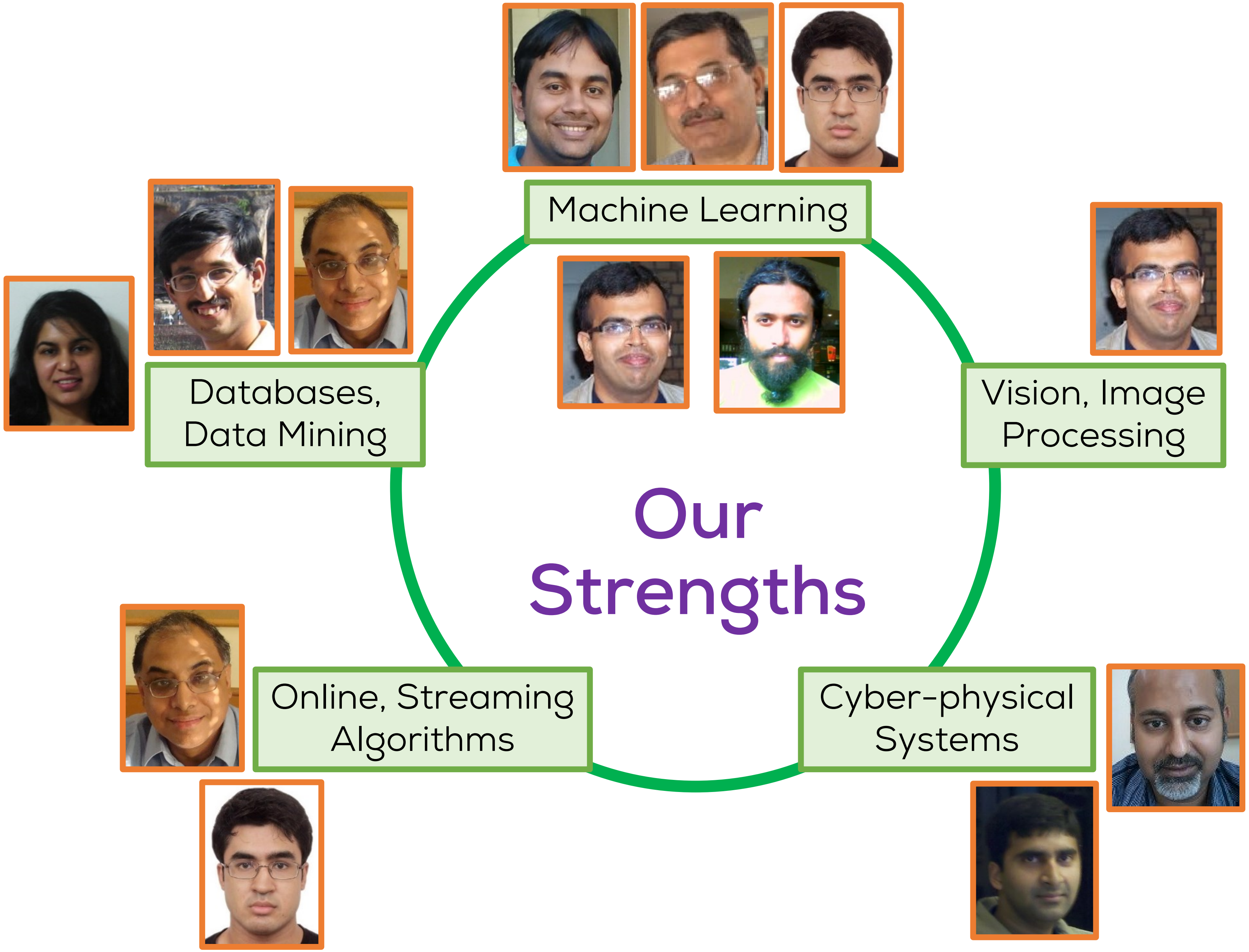
The Data Sciences Gang @ CSE, IITK



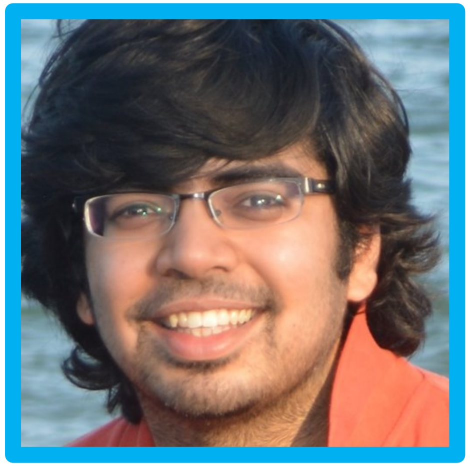





Machine Learning
Databases, Data Mining
Vision, Image Processing
Our Strengths
Online, Streaming Algorithms
Cyber-physical Systems

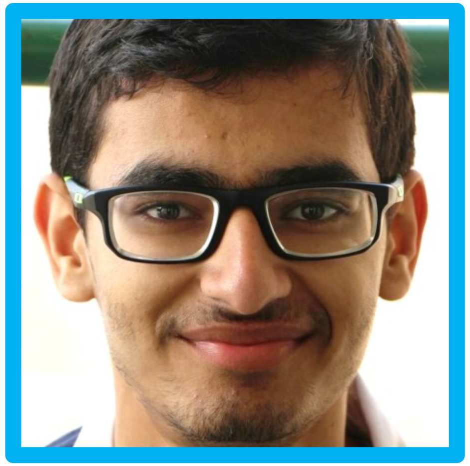
Gratitude

Prateek Jain
MSR



Kush Bhatia
UC Berkeley

Govind Gopakumar
IIT Kanpur



Sayash Kapoor
IIT Kanpur

Kshitij Patel
IIT Kanpur