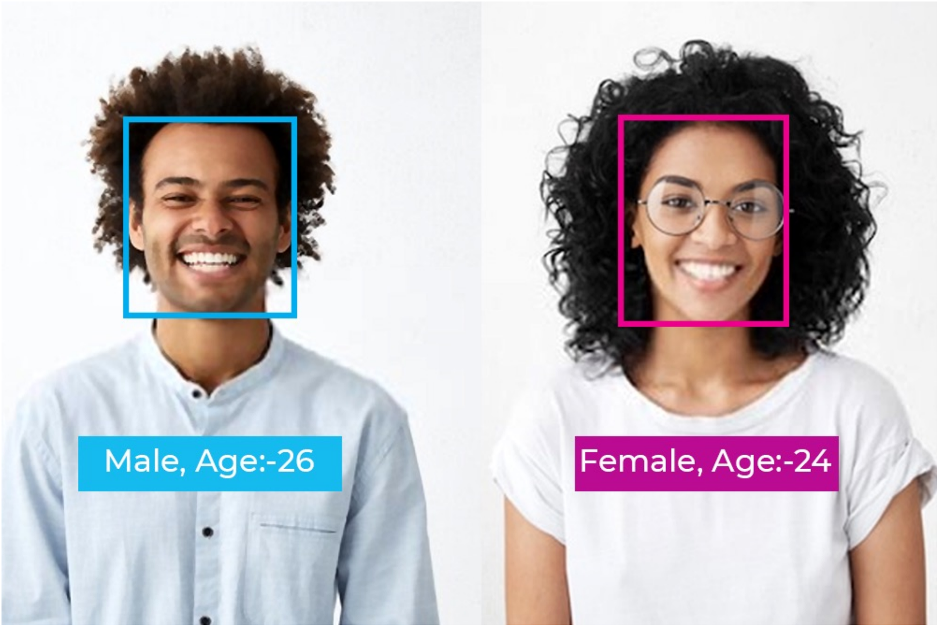
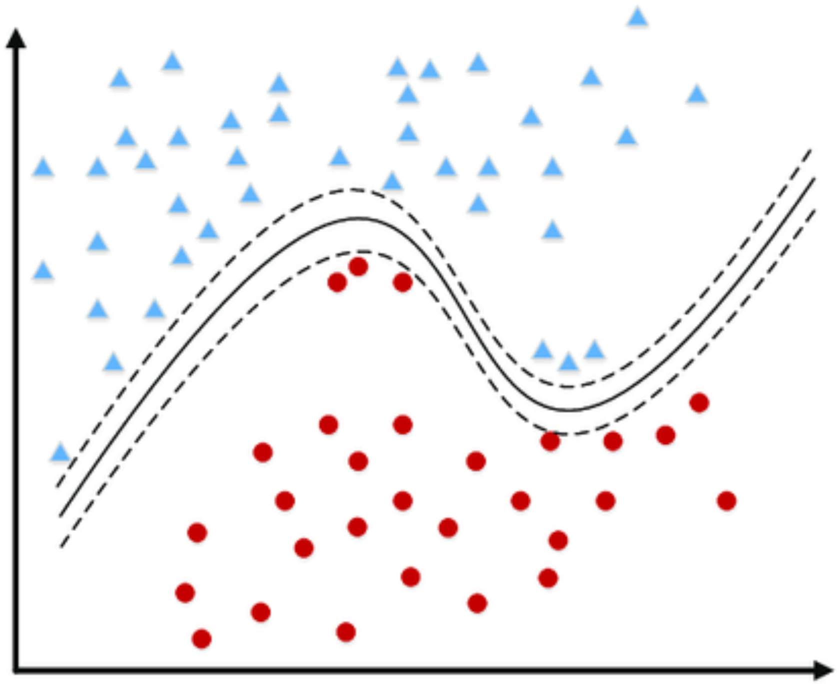
Computationally Efficient Learning under Noisy Data

Christos Tzamos

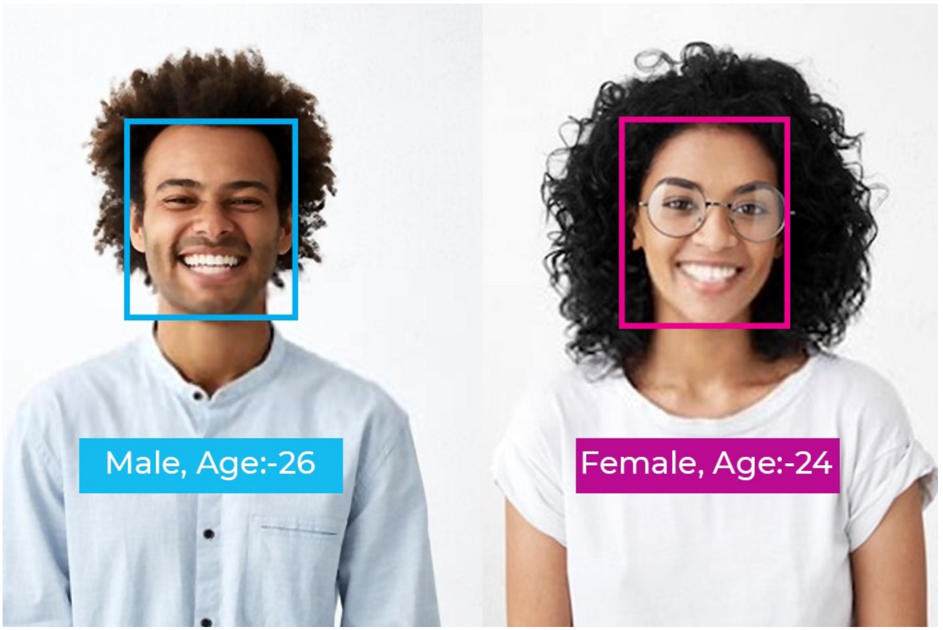
University of Athens & Archimedes AI

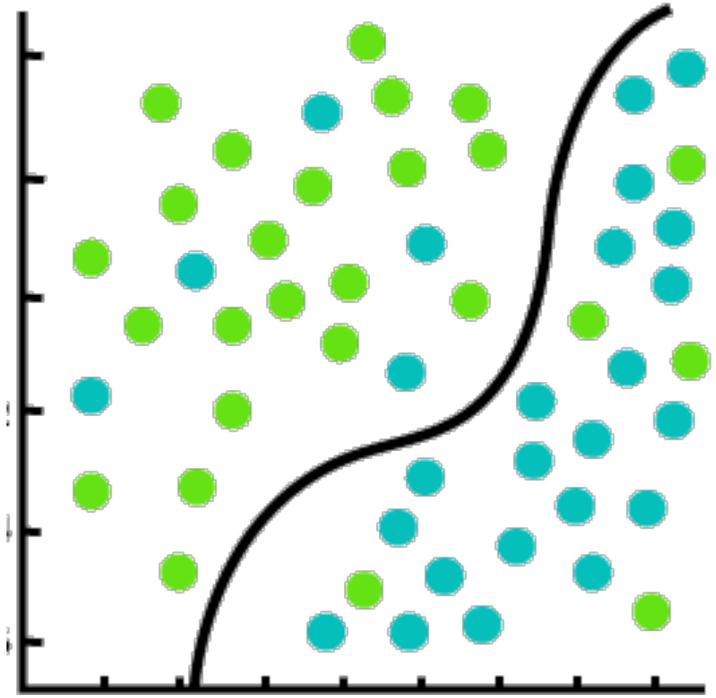
Classification





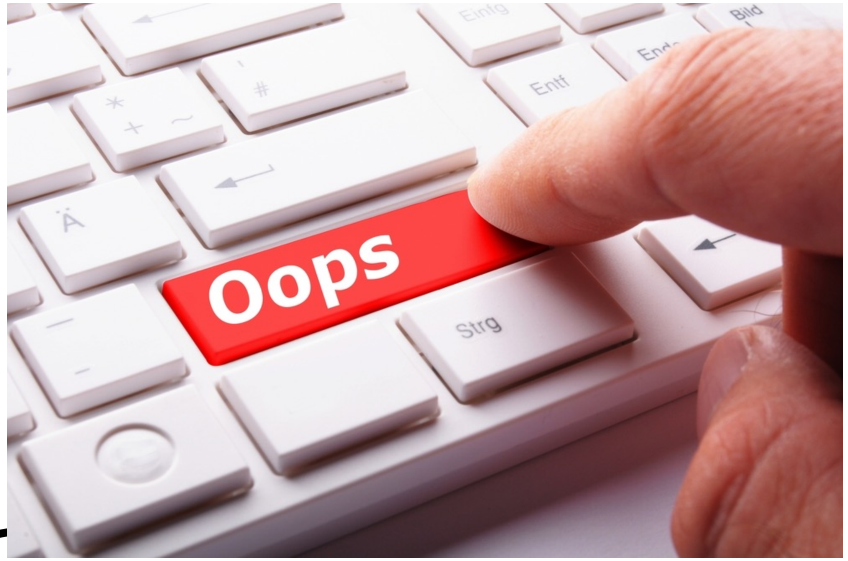
Classification with Noise

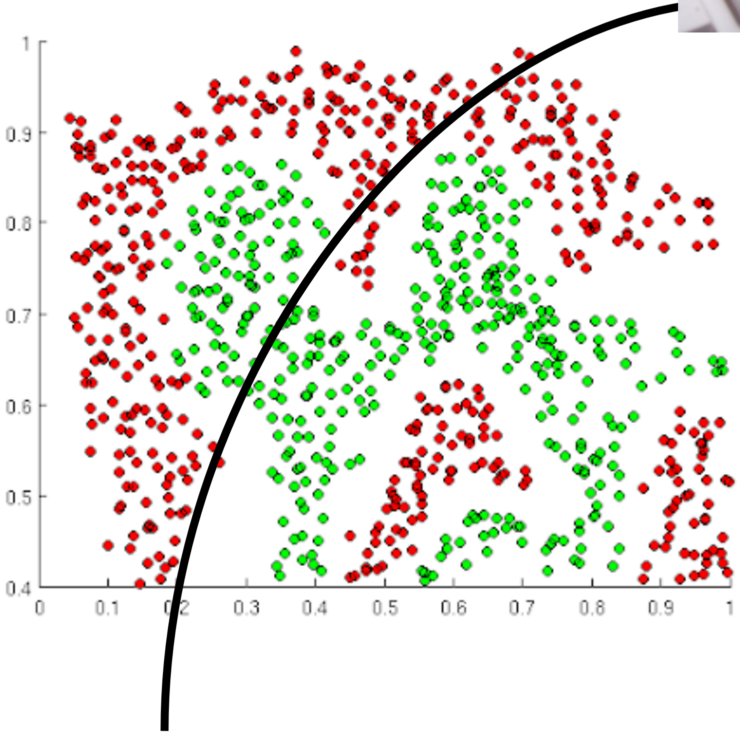




Why noise?

- Human Mistakes (crowdsourcing)
- Measurement error
- Model error
- ...





Imperfect Data

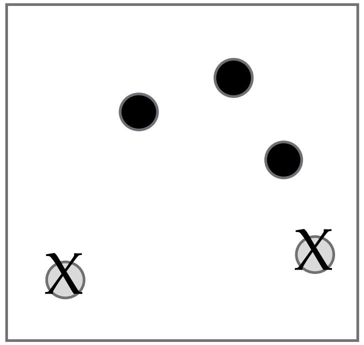
ML datasets in practice are huge and imperfect

$\Longrightarrow$ we need provably efficient and robust algorithms

Wrong Labels

Dog

Hidden Biases



Data are Truncated

Coarse Labels

Animal

Current Status: Fragile with Noise

- Data cleaning
- Models fragile to Various Attacks: Adversarial Examples / Data Poisoning
- High noise applications, e.g. Signal Processing

How to obtain robust classifiers?

The need for theory

Noise can come in many different shapes from many different sources

Techniques for one setting may not be applicable in others

- Must understand which settings are solvable and how to approach them
- Theory can also guide the development for novel more robust techniques

Theoretical Setup

- Data generating distribution
  - Examples (x, y) are drawn from a distribution D
  - Focus on binary classification where $y = +1$ or -1

- Train classifier $h: X \rightarrow \{+1, -1\}$ on a random set of N samples $(x_1, y_1), \ldots, (x_N, y_N)$

- Goal: Minimize the probability of error on a fresh sample (x, y) from D

- Noiseless: $\Pr[h(x) \neq y] \leq \varepsilon$
- Agnostic: $\Pr[h(x) \neq y] \leq OPT + \varepsilon$
  - where OPT is the best model c from a class C in that minimizes $\Pr[c(x) \neq y]$

Statistically

For a class C with **VC dimension** d ($\simeq$ d **parameters**) we can learn with error ε as long as we have sufficiently many samples

- **Noiseless**, where the best model gets 0 error
  - $N = \Theta(d/\varepsilon)$
- **Agnostic**, where the best model gets OPT > 0 error
  - $N = \Theta(d/\varepsilon^2)$

Statistically, the **agnostic** case is not much harder than the **noisless** case!

Computational Challenges

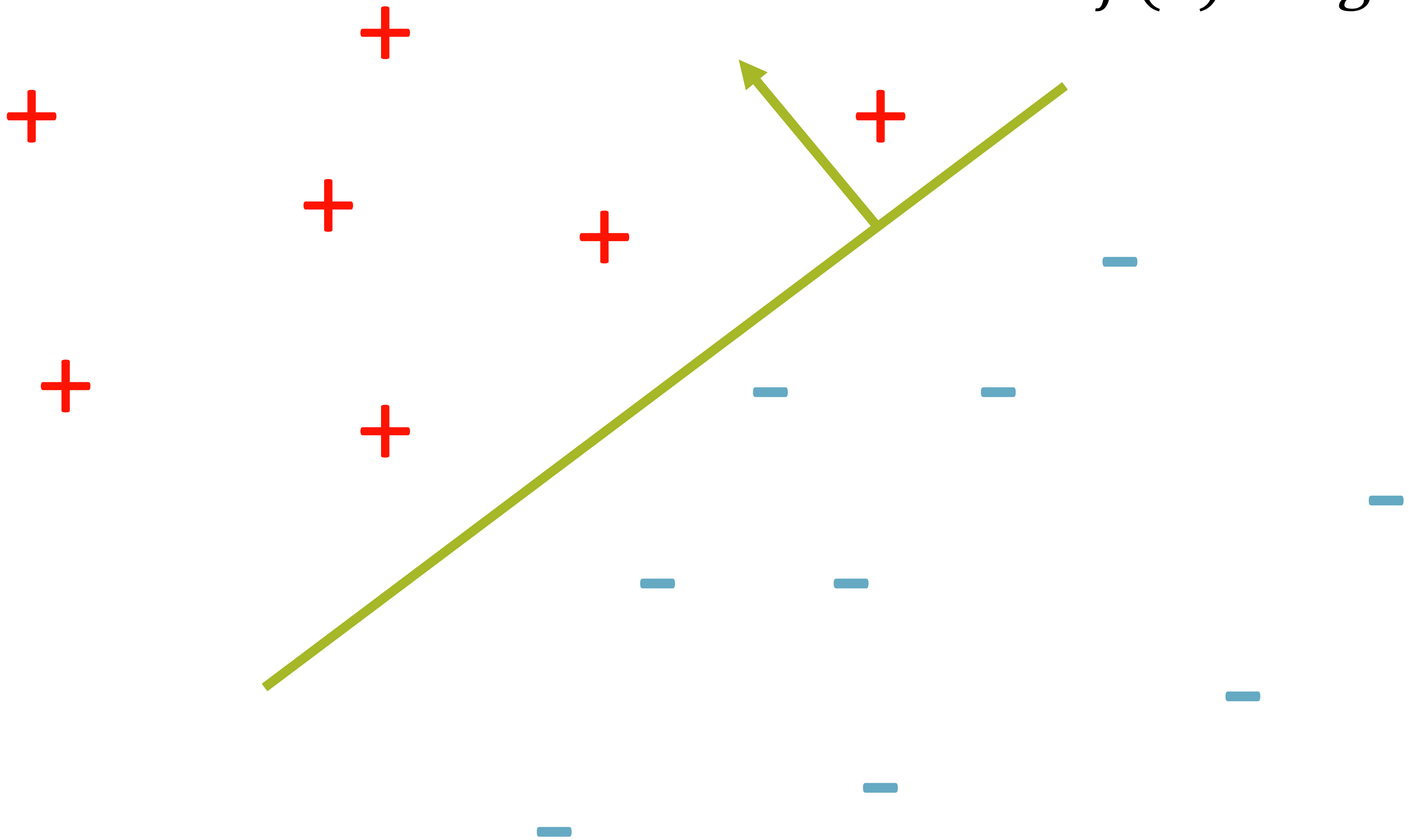
Finding a good set of parameters computationally efficiently is highly non-trivial!

- Optimization: find good parameters via local search methods
  - Gradient Descent, Second order methods, …
  - Do they converge to good parameters?
  - Proper learning

- Learning theory: train any classifier that performs at least as good
  - Improper learning
  - Overparameterization

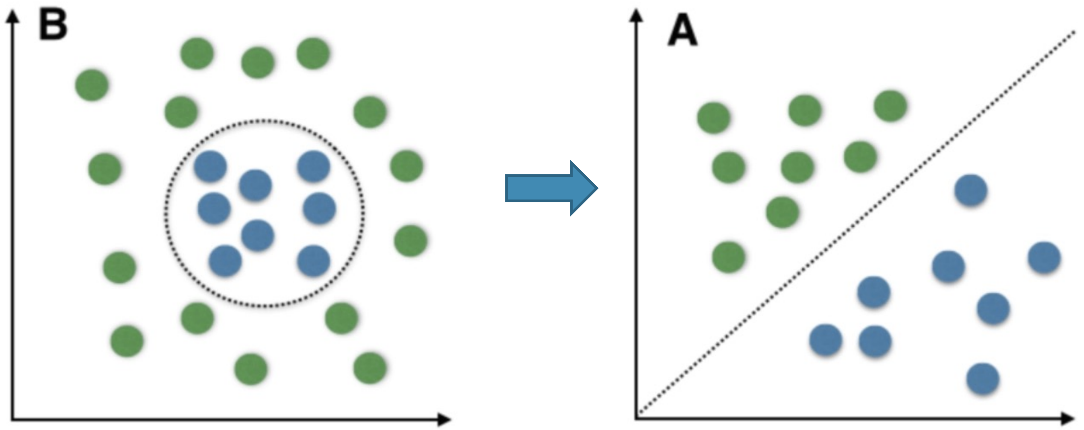
Linear Classification

$$f(x) = sgn(w^* \cdot x)$$



Why Linear Classification?

- More complex classifiers can be seen as linear classification over more complex features

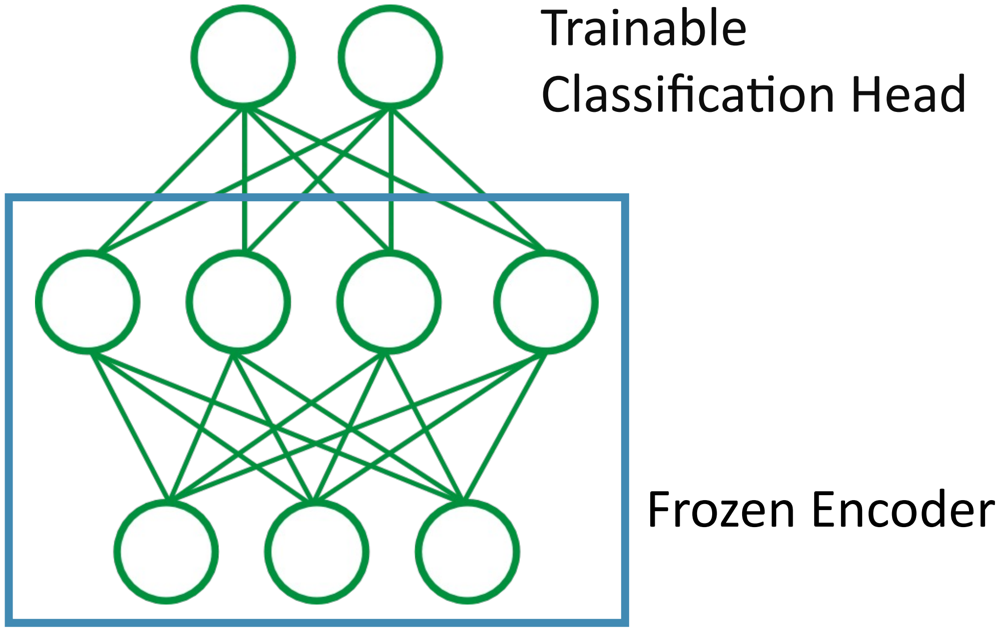


Features (x_1, x_2)

$(x_1 - 5)^2 + (x_2 - 3)^2 \leq 1$

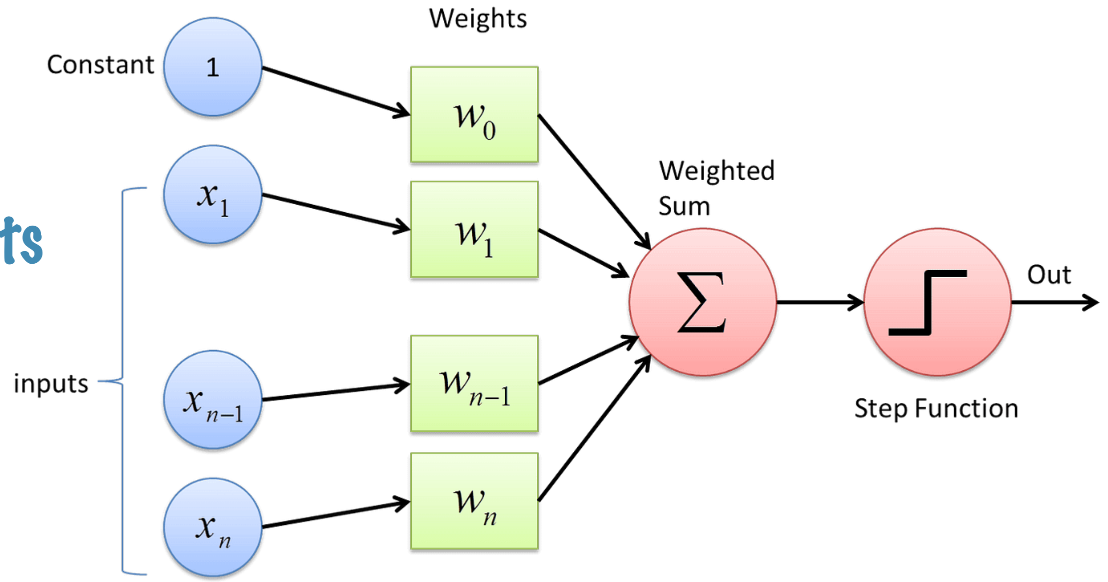
Features $(1, x_1, x_2, x_1 x_2, x_1^2, x_2^2)$

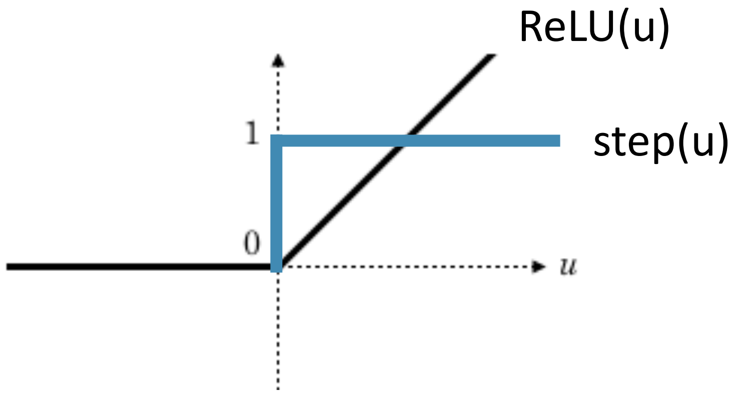
$x_1^2 - 10x_1 + x_2^2 - 6x_2 \leq -33$



Perceptron for Linear Classification

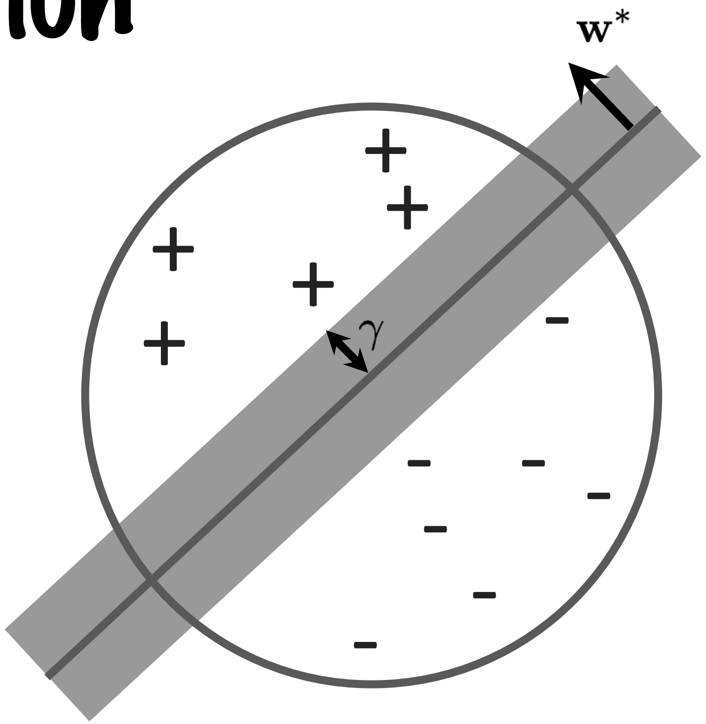
- Perceptron [Rosenblatt '58]
  - An iterative method for updating the weights of a linear function
  - For every misclassified example (x,y) set:
    - $w' \leftarrow w + y \cdot x$
  - Can be seen as gradient descent on the objective
    - $g(w) = E[\ \mathrm{ReLU}(-y \cdot w \cdot x)\ ]$
  - This is a convex proxy for $E[\mathrm{step}(-y \cdot w \cdot x)]$





Algorithms for Linear Classification

- Linear Classification with margin γ
  - the **Perceptron** algorithm [Rosenblatt'58] finds a perfect linear separator in $O(1/\gamma^2)$ iterations.
  - Linear Programming via **Ellipsoid** finds a perfect linear separator in $O(\log(1/\gamma))$ iterations.
- Major Open Problem in CS:
  - Is there an algorithm that doesn't depend on γ?
  - I.e. strongly polynomial time



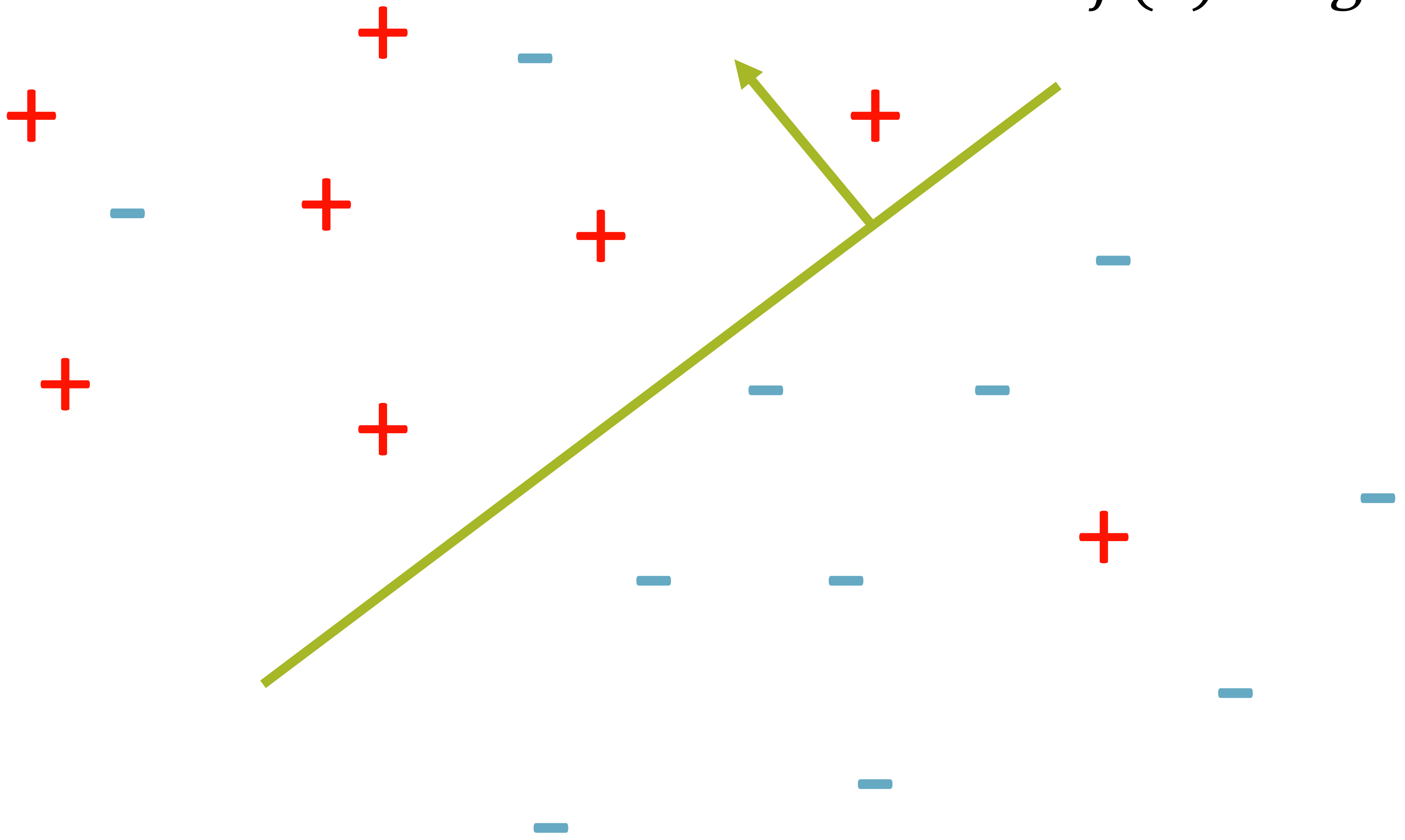
[DiakonikolasKaneT STOC'23]

Can improperly learn linear classifiers in strongly polynomial time

(with a decision list of *d log n* linear classifiers)

Linear Classification with Noise

$$f(x) = sgn(w^* \cdot x)$$



Linear Classification with Noise

- Strong Negative Result

[Guruswami-Raghevendra'06, Feldman et al.'06, Daniely'16]
Even if only 1% of the data are corrupted, it is even computationally intractable to compute a classifier with 49% error.

^ even improper

Too Pessimistic: Applies for some adversarially chosen setting
Hopefully can do something better in practice

Milder Cases: Escaping Impossibility

- Non-adversarial settings, more structure
- Structure on x:
  - Data are gaussian / Large margin
- Structure on y:
  - Separable but random noise was added

Today's Menu

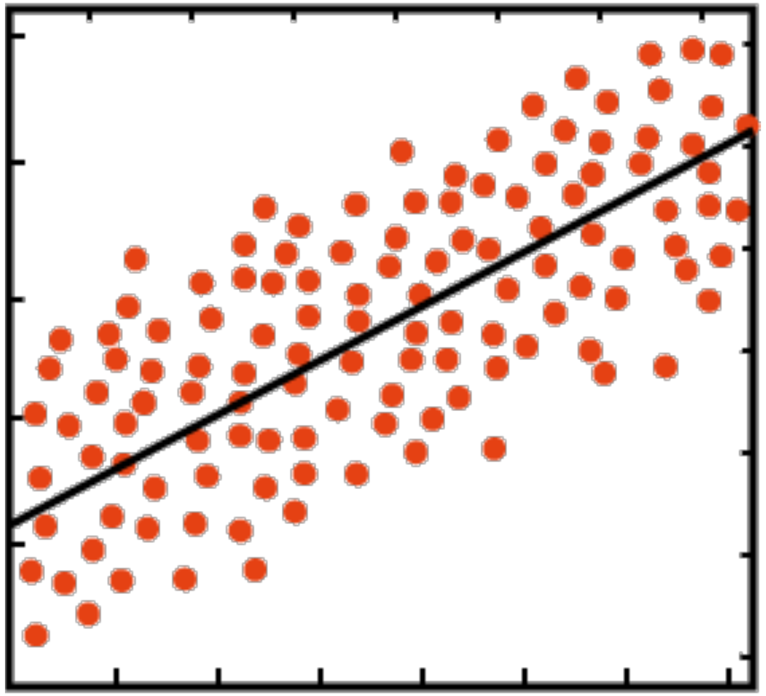
- Structure on x: Gaussian Data
- Structure on y: Noise Model
- Structure on both x and y

- Main techniques:
  - From Classification to Polynomial Regression
  - Debiasing Statistical Queries
  - Iterative Peeling
  - Localization
  - Certificate Framework

Structure on x

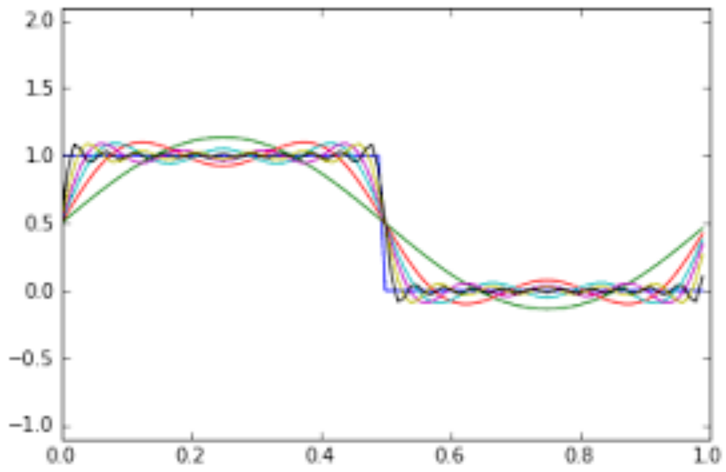
Structure on x

- A generic approach:
  - Treat classification as regression
  - i.e. minimize $E[\ (f(x) - y)^2\ ]$ and then look at $\text{sign}(\ f(x)\ )$
- If f(x) is flexible enough it can fit the +1,-1 labels
- This does not work in general but does so when the data are structured



Structure on x: Gaussian Data

- When the data are drawn from a Gaussian
  - Polynomial Regression works as polynomials can approximate the step function arbitrarily well!
  - [Kalai, Klivans, Mansour, Servedio '05]



- However, we need high polynomial degree $(1/\varepsilon^2)$
  - [Diakonikolas, Kane, Pittas, Zarifis '21]

- Runtime is $d^{\mathrm{poly}(1/\varepsilon)}$ but also the sample complexity is similarly high

Polynomial Regression for Other Classes

- A classifier class is approximable by polynomial regression if it has low complexity
- [Kalai, Klivans, Mansour, Servedio '08] measure the complexity in terms of a concept called Gaussian Surface Area

Concept Class	Gaussian Surface Area	Sample Complexity
Polynomial threshold functions of degree k	$O(k)$ [Kan11]	$d^{O(k^2)}$
Intersections of k halfspaces	$O(\sqrt{\log k})$ [KOS08]	$d^{O(\log k)}$
General convex sets	$O(d^{1/4})$ [Bal93]	$d^{O(\sqrt{d})}$

Still runtime is $d^{\mathrm{poly}(1/\varepsilon)}$ and the sample complexity is similarly high

Today's Menu

- Structure on x: Gaussian Data ✓
- Structure on y: Noise Model
- Structure on both x and y

- Main techniques:
  - From Classification to Polynomial Regression ✓
  - Debiasing Statistical Queries
  - Iterative Peeling
  - Localization
  - Certificate Framework

Structure on y

Structure on y: The Generative Process

- Ground Truth: $f(x) = sgn(w^* \cdot x)$
- Sample $x \sim D$
- Generate Noisy label of x

$$y = \begin{cases} -f(x) & w.p.\ \eta(x) \\ f(x) & o/w \end{cases}$$

Goal: Find hypothesis $h(x)$

$$\Pr[h(x) \neq y] \leq \underbrace{\Pr[f(x) \neq y]}^{\text{OPT}} + \epsilon$$

Random Classification Noise

- Introduced by [Angluin-Laird'88]
- The simplest noise model: equal probability of flips η (say 1%)

$$y = \begin{cases} -f(x) & w.p.\ \eta \\ f(x) & o/w \end{cases}$$

- Common baseline in practice

- [Blum-Frieze-Kannan-Vempala'96] gave a computationally efficient algorithm for this problem

How to learn under RCN?

- While individual examples can be noisy, aggregate statistics over the data can be denoised
- Statistical Queries [Kearns '98]: For a given function q compute
  - $\mathbf{E}[q(x, y)]$ over the distribution of data
- Nearly all existing algorithms can be implemented using statistical queries
- They can thus directly work for Random Classification Noise

Perceptron Using Statistical Queries

- Noiseless case:
  - The perceptron updates weights using a misclassified example (x,y) set:
    - $w' \leftarrow w + y \cdot x$
  - One can replace the single example with $E[\,(1-y)\,x \mid wx > 0]$
  - The $1-y$ term ignores all examples with $y=1$ and averages over all the misclassified examples with $y = -1$ in the region $wx > 0$
- RCN case with noise 1%:
  - Compute instead $E[\,(0.98-y)\,x \mid wx > 0]$
  - For examples that are positive $E[y] = 0.98$ and thus are ignored in expectation.
  - For the remaining examples $E[y] = -0.98$ and thus this expectation still averages over misclassified examples.

Denoising RCN

- The same principle can be applied for pretty much any problem
- There is a generic way of denoising Statistical Queries

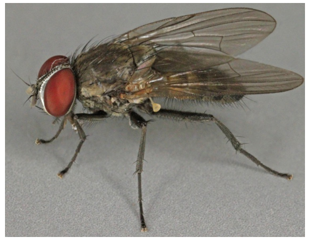
- Yet this idea can be unrealistic in practice because it assumes that the amount of noise is known for every example a priori.

- Even if it is unknown, since this is only a single parameter one could try all possible values (up to some discretization)

Semi-Random noise models

- The uniform noise assumption is often unrealistic
- The error rate varies depending on the example

Bee



Fly

Bee “Noisier”

$\eta(x)$ =30%

Semi-Random noise models

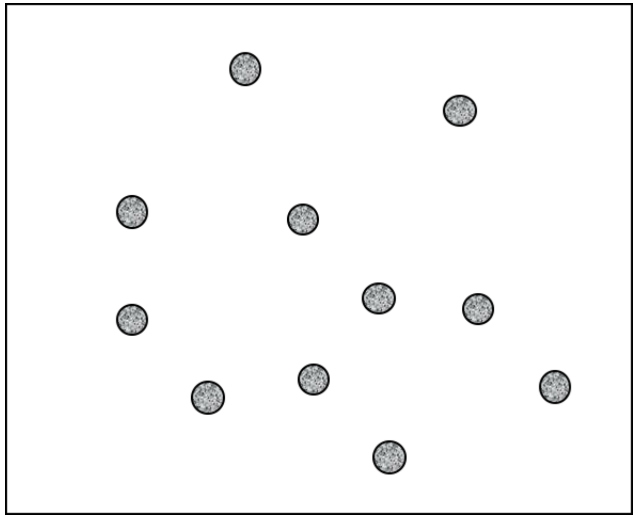
- Ground Truth: $f(x) = sgn(w^* \cdot x)$
- Sample $x \sim D$
- Generate Noisy label of x

$$y = \begin{cases} -f(x) & w.p.\ \eta(x) \\ f(x) & o/w \end{cases}$$

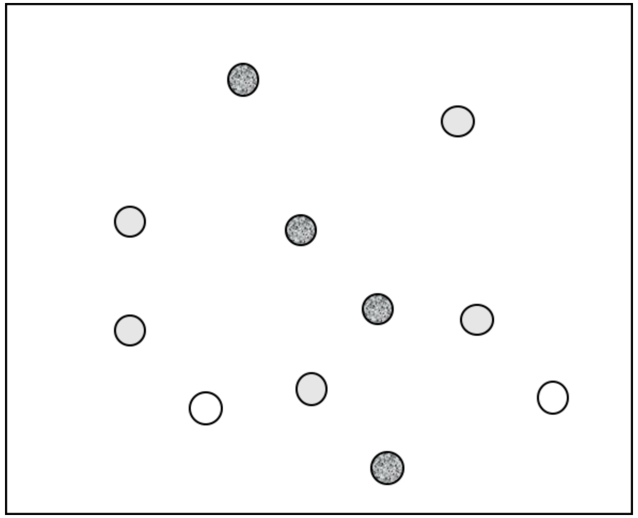
Massart Noise, also known as Malicious misclassification noise [Sloan'88, Rivest-Sloan'94]: $\eta(x) \le \eta \le 1/2$

Every label is randomly flipped with probability at most 1% but the exact probabilities are adversarially chosen

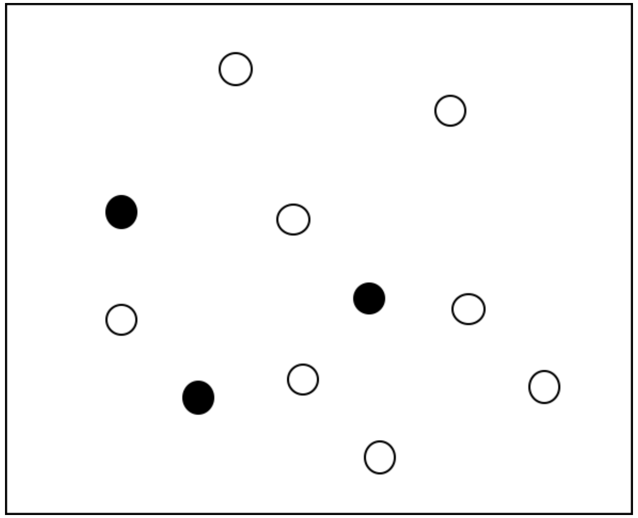
Summary of Noise Models



RCN
Noise Rate **exactly** 1%



Massart
Noise Rate **at most** 1%



Agnostic
Arbitrary 1% fraction

Results for Massart Noise

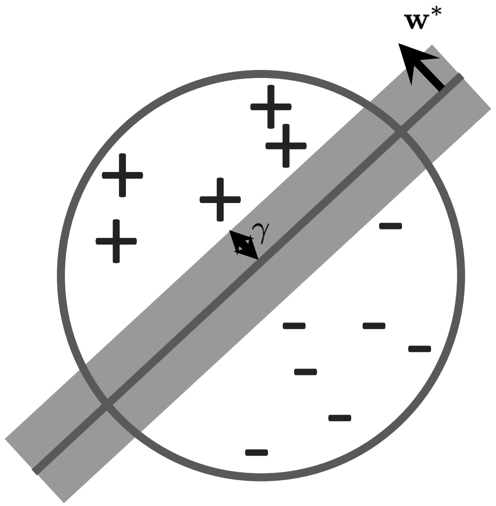
First efficient algorithm for linear separators with Massart noise.

[DiakonikolasKaneT NeurIPS'19 Best Paper]

With a d-dimensional dataset corrupted with Massart noise at most η, we can compute a hypothesis with misclassification error $\eta+\epsilon$ in time poly(d, $1/\epsilon$)

Approach

Target Vector $\mathbf{w}^*$

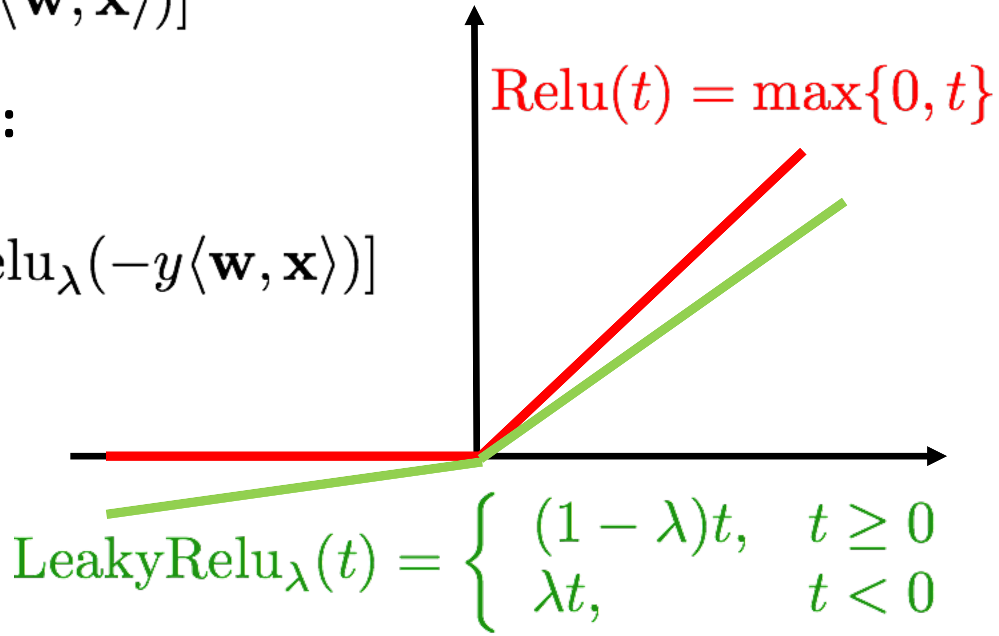


- **Realizable Case:**
  (Perceptron =) SGD on convex surrogate
  $$L_0(\mathbf{w}) = \mathbf{E}_{(\mathbf{x},y)\sim\mathcal{D}}[\mathrm{Relu}(-y\langle \mathbf{w}, \mathbf{x}\rangle)]$$

- **Random Classification Noise:**
  SGD on convex surrogate
  $$L_\lambda(\mathbf{w}) = \mathbf{E}_{(\mathbf{x},y)\sim\mathcal{D}}[\mathrm{LeakyRelu}_\lambda(-y\langle \mathbf{w}, \mathbf{x}\rangle)]$$
  for $\lambda \approx \eta$

In both cases:

$$L(\mathbf{w}) \geq 0 \text{ and } L(\mathbf{w}^*) = 0$$



Approach for Massart Noise

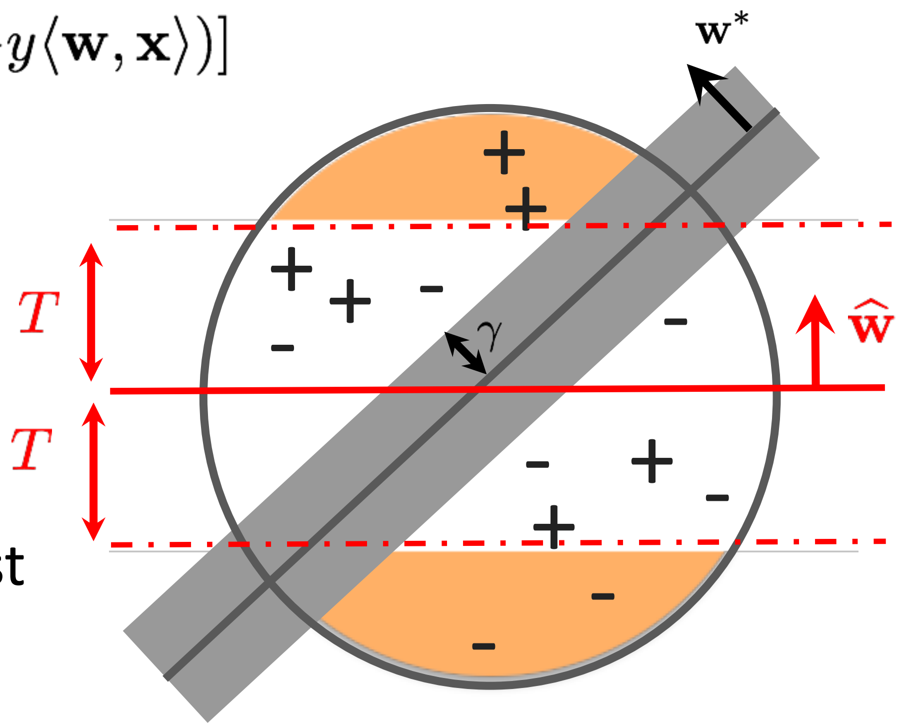
Lemma 1: No convex surrogate works.

But…

Lemma 2: Let $\widehat{\mathbf{w}}$ be the minimizer of

$$L_\lambda(\mathbf{w}) = \mathbf{E}_{(\mathbf{x},y)\sim\mathcal{D}}[\mathrm{LeakyRelu}_\lambda(-y\langle \mathbf{w}, \mathbf{x}\rangle)]$$

for $\lambda \approx \eta$.
Then, $\widehat{\mathbf{w}}$ must get error-rate less than $\eta + \epsilon$ for points far from $\widehat{\mathbf{w}}$



IDEA: Use $\widehat{\mathbf{w}}$ as a classifier for those points and recurse on the rest
Iterative Peeling

Results for Massart Noise

First efficient algorithm for linear separators with Massart noise.

[DiakonikolasKaneT NeurIPS'19 Best Paper]

With a d-dimensional dataset corrupted with Massart noise at most η, we can compute a hypothesis with misclassification error $\eta+\varepsilon$ in time poly(d, $1/\varepsilon$)

Is this the same as getting error OPT + ϵ ?

No, OPT = $\mathbf{E}[\eta(x)]$ which can be smaller than η

Results for Massart Noise

First efficient algorithm for linear separators with Massart noise.

[DiakonikolasKaneT NeurIPS'19 Best Paper]

With a d-dimensional dataset corrupted with Massart noise at most η, we can compute a hypothesis with misclassification error $\eta+\varepsilon$ in time poly(d, $1/\varepsilon$)

Can we get OPT + ϵ efficiently?

No without assumptions on the distribution D that generates x

Distribution Free

Computationally Challenging:

Super-polynomial SQ Lower Bounds

[Chen Koehler Moitra Yau '20]

[Diakonikolas Kane '20]

[Nasser Tiegel '22]

Today's Menu

- Structure on x: Gaussian Data ✓
- Structure on y: Noise Model ✓
- Structure on both x and y

- Main techniques:
  - From Classification to Polynomial Regression ✓
  - Debiasing Statistical Queries ✓
  - Iterative Peeling ✓
  - Localization
  - Certificate Framework

Structure on both x and y

Massart Noise + Gaussian Data

Long line of work

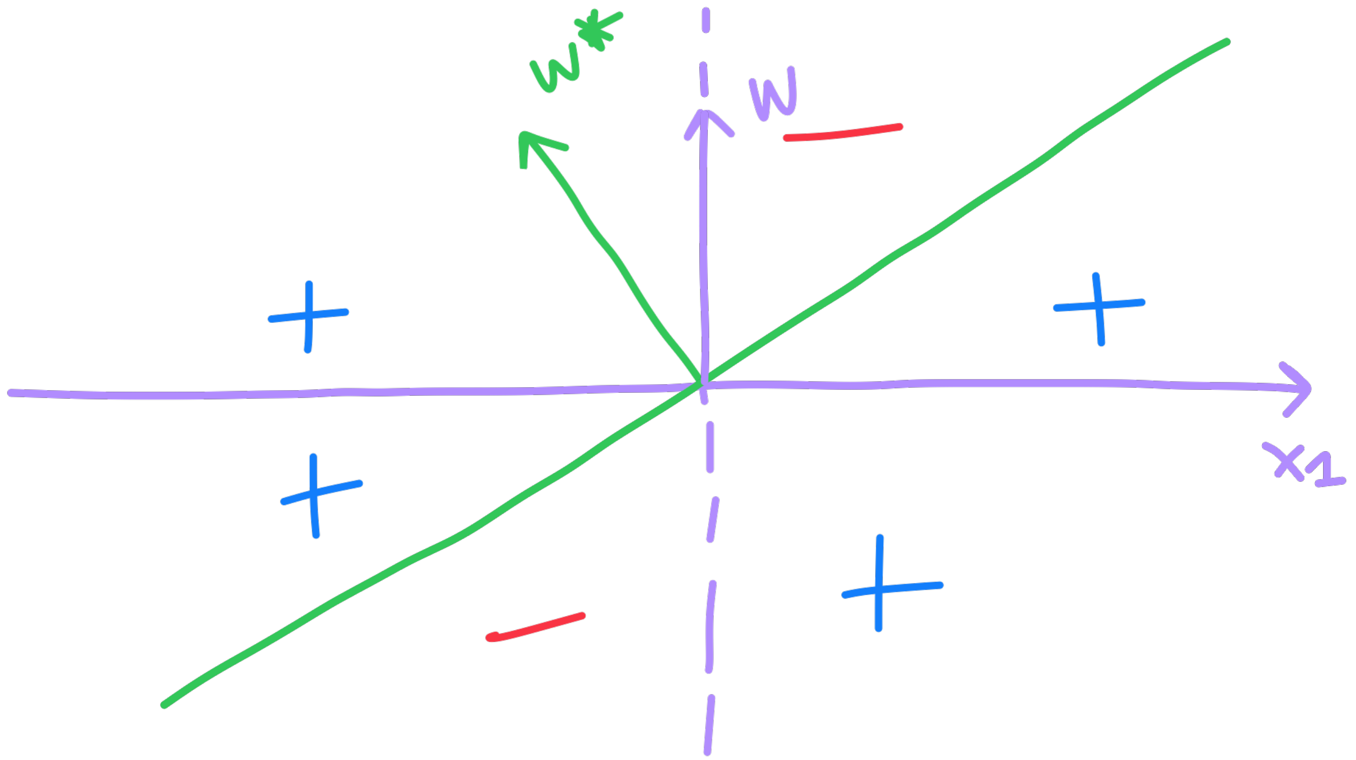
[Awasthi Balcan Haghtalab Urner ’15]
[Awasthi Balcan Haghtalab Zhang ’16]
[Balcan Zhang ’17]
[Yang Zhang ’17]
[Zhang Liang Charikar ’17]
[Diakonikolas Kontonis Zarifis Tzamos ’20]
[Zhang Shen Awasthi ’20]
[Zhang Li ’21]

Gaussian x-Marginal
Extends to other well-behaved distributions like log-concave

poly$\left(\frac{d}{(1-2\eta)\epsilon}\right)$ samples and runtime

Key Technique: Localization

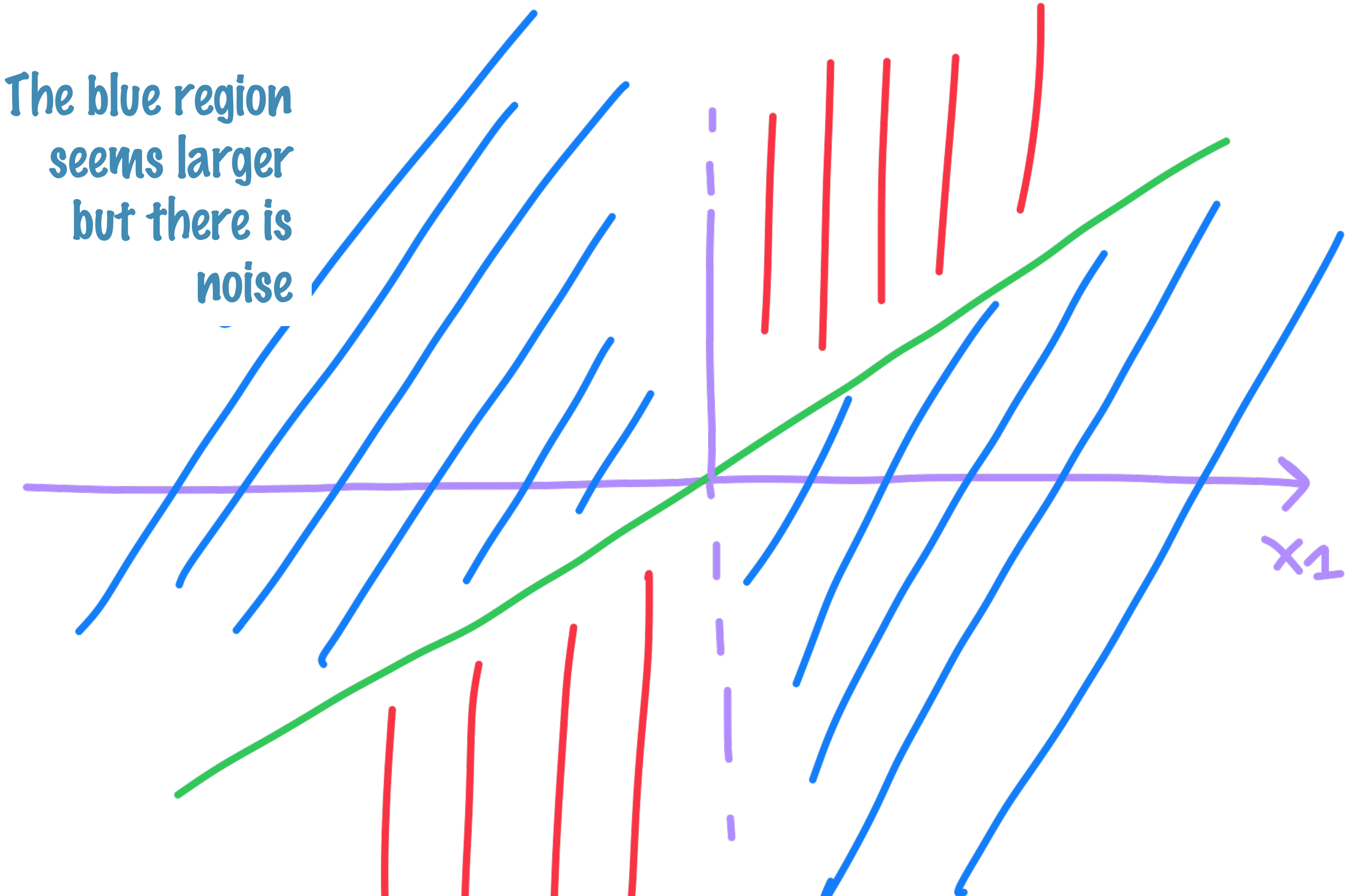
- Given any w, need to update the weight to move closer to w*
- Consider setting w′ = w + E[y x]



Key Technique: Localization

- Given any w, need to update the weight to move closer to w*
- Consider setting $w' = w + E[y x]$

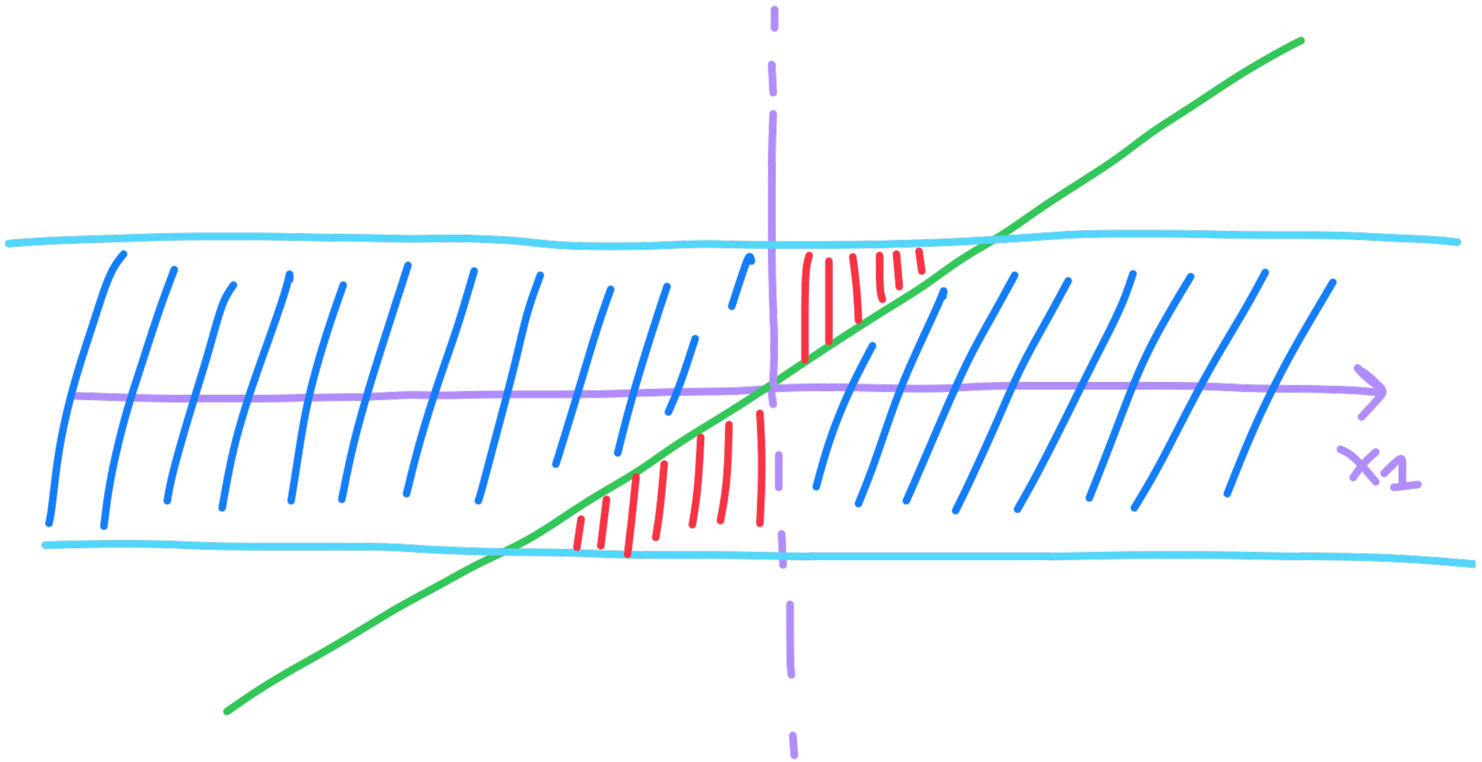
The blue region seems larger but there is noise



Might not turn in the correct direction

Key Technique: Localization

- Given any w, need to update the weight to move closer to w*
- Consider setting w' = w + E[y x | such that |w.x| < ρ]



Turns in the correct direction!

Massart Noise + Gaussian Data

Long line of work
[Awasthi Balcan Haghtalab Urner ’15]
[Awasthi Balcan Haghtalab Zhang ’16]
[Balcan Zhang ’17]
[Yang Zhang ’17]
[Zhang Liang Charikar ’17]
[Diakonikolas Kontonis Zarifis Tzamos ’20]
[Zhang Shen Awasthi ’20]
[Zhang Li ’21]

$\text{poly}\left(\frac{d}{(1-2\eta)\epsilon}\right)$ samples and runtime

Gaussian x-Marginal
Extends to other well-behaved distributions like log-concave

Assumptions

- Noise Rate $\eta < 1/2$ for all x
- Homogeneous Halfspaces

$$f(x) = \text{sgn}(w^* \cdot x)$$

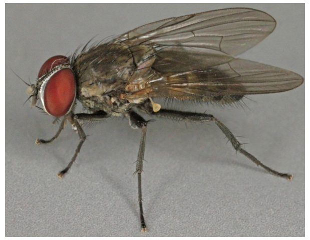
vs

$$f(x) = \text{sgn}(w^* \cdot x + t^*)$$

What about random labels?

Actual Imagenet example
[Vasudevan, et al.'22]

Bee



Fly

Bee "Noisier"
30%

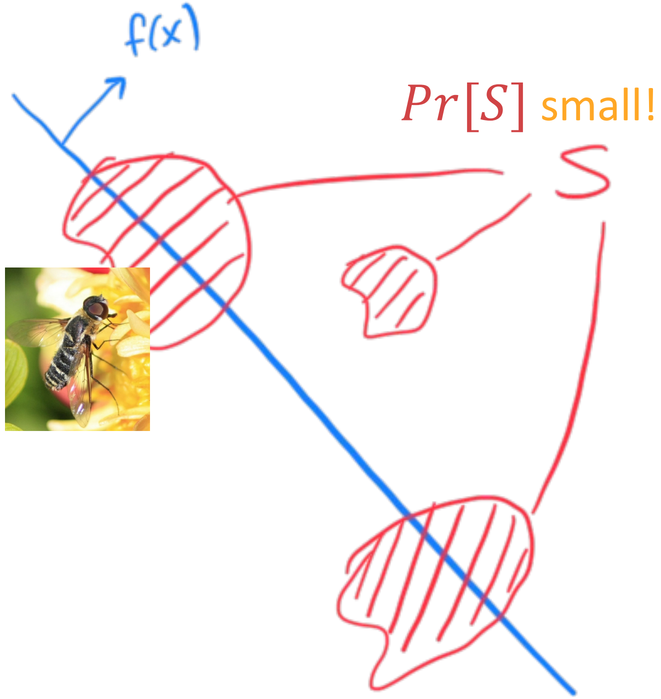
Fly or Bee?

Massart Noise
[Massart, Nedelec '06]

$$\eta(x) \leq \eta \leq 1/2$$

Non-expert human annotators often flip (almost) random coins for harder examples [Klebanov, Beigman '09]

General Massart Noise



For all $x \in S$: $\eta(x) = 1/2$

$Pr[f(x) \neq y] = Pr[S]/2$

Want to find a halfspace with error

$Pr[h(x) \neq y] \leq Pr[S]/2 + \epsilon$

Homogeneous vs General

- Homogeneous: $\mathrm{sgn}(w^* \cdot x)$ vs General: $\mathrm{sgn}(w^* \cdot x + t^*)$

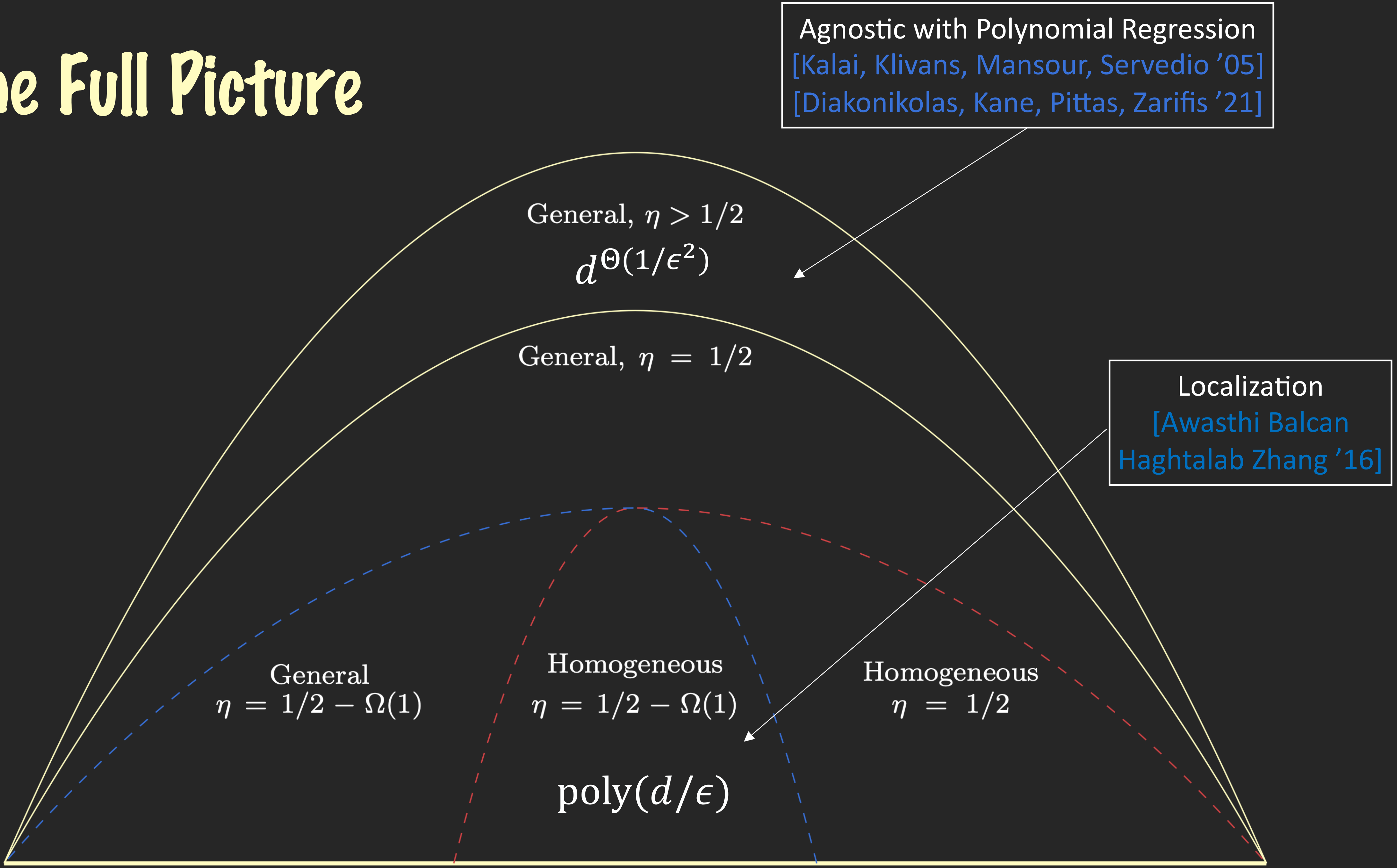
Adding a threshold shouldn't be a problem…

Homogeneous vs General

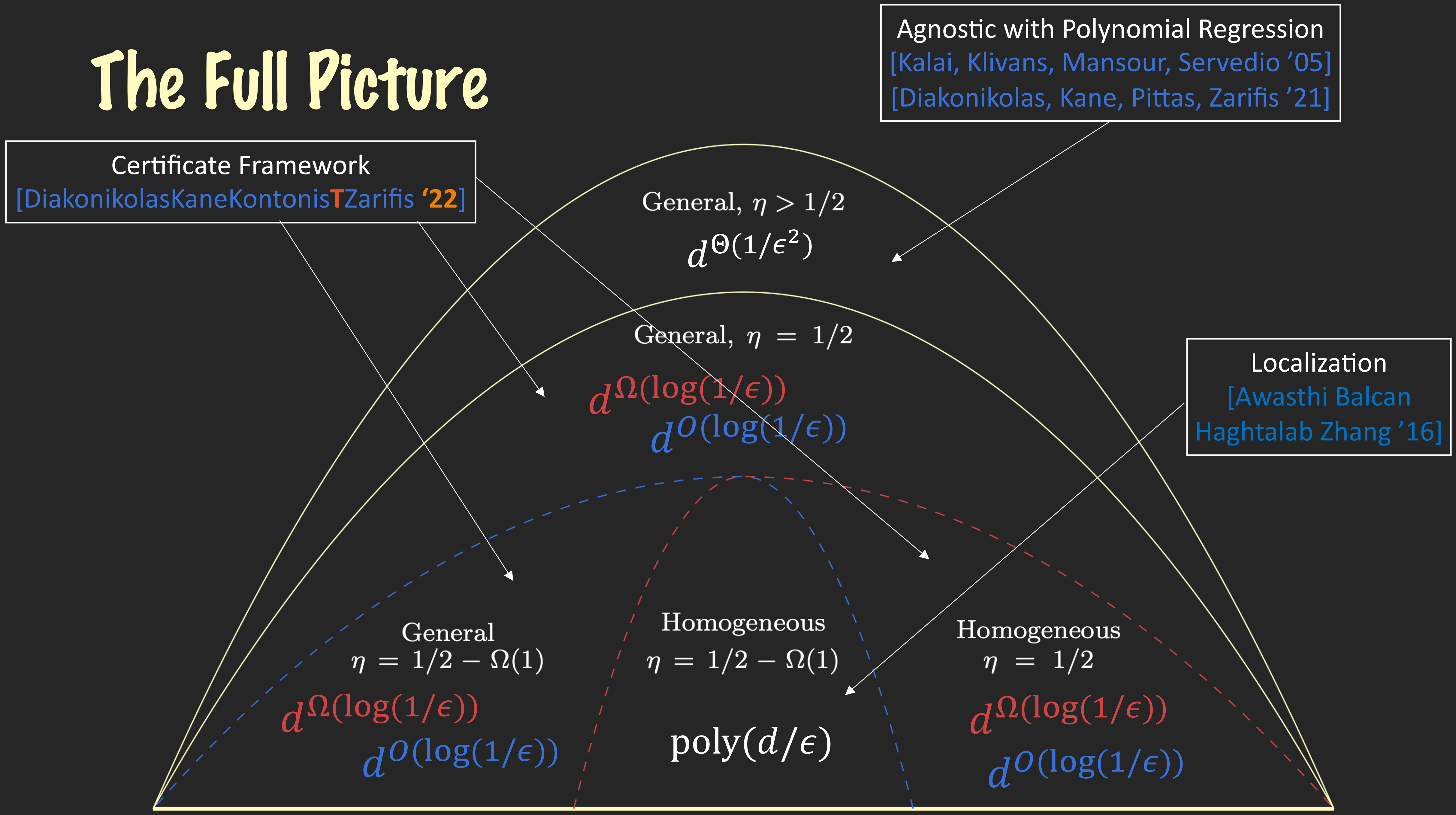
- Homogeneous: $\mathrm{sgn}(w^* \cdot x)$ vs General: $\mathrm{sgn}(w^* \cdot x + t^*)$
- Adapt a Homogeneous learner to General:
- OK if the learner works in the Distribution-Free setting.
- Does not work in Distribution Specific!

  The -marginal of the transformed instance is not Gaussian N(0,I)

The Full Picture



The Full Picture



The certificate framework

The Certificate Framework

Ground Truth: $f(x) = sgn(w^* \cdot x)$

$$y = \begin{cases} -f(x) & w.p.\ \eta(x) \\ f(x) & w.p.\ 1 - \eta(x) \end{cases}$$

For Ground Truth w^*

for every $T(x) \geq 0$: $\mathbf{E}[w^* \cdot xy\, T(x)] \geq 0$

Proof

$$\mathbf{E}[y|x] = f(x)(1 - \eta(x)) - f(x)\eta(x) = (1\text{-}2\eta(x))\, f(x)$$

$$\mathbf{E}[w^* \cdot xy\, T(x)] = \mathbf{E}[w^* \cdot xf(x)(1 - 2\eta(x))\, T(x)]$$

$$\mathbf{E}[w^* \cdot xy\, T(x)] = \mathbf{E}[|w^* \cdot x|(1 - 2\eta(x))\, T(x)] \geq 0$$

The Certificate Framework

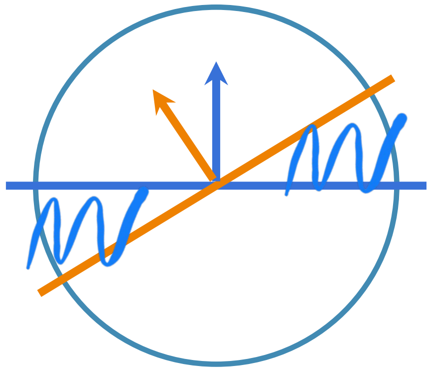
Ground Truth: $f(x) = sgn(w^* \cdot x)$

$$y = \begin{cases} -f(x) & w.p.\ \eta(x) \\ f(x) & w.p.\ 1 - \eta(x) \end{cases}$$

For $w \neq w^*$

exists $T(x) \geq 0$: $\mathbf{E}[w \cdot xy\, T(x)] < 0$

Proof



Pick $T(x) = 1\{(w \cdot x)f(x) < 0\}$

$$\mathbf{E}[w \cdot xy\, T(x)] = \mathbf{E}[w \cdot xf(x)(1 - 2\eta(x))\, 1\{w \cdot xf(x) < 0\}] < 0$$

Certificate Framework

[Diakonikolas Kontonis Tzamos Zarifis '20]

Ground Truth: $\ell^*(x) = w^* \cdot x + t^*$

for every $T(x) \geq 0$: $\mathbf{E}[\ell^*(x) y\, T(x)] \geq 0$

For $\ell(\cdot) \neq \ell^*(\cdot)$:

there exists $T(x) \geq 0$: $\mathbf{E}[\ell(x) y\, T(x)] < 0$

The Certificate Framework

[Diakonikolas K. Tzamos Zarifis '20]
[Diakonikolas Kane K. Tzamos Zarifis '21]
[Diakonikolas Kane K. Tzamos Zarifis '22]

Ground Truth: w^*

(LP) : Find w (with $\| w \|_2 = 1$)

for every $T(x) \geq 0$: $\mathbf{E}[w \cdot xy\, T(x)] \geq 0$

Separation Oracle

Given $w \neq w^*$

Efficiently Compute $T(x) \geq 0$: $\mathbf{E}[w \cdot xy\, T(x)] < 0$

The Certificate Framework

Separation Oracle

Given $w \neq w^*$

Efficiently Compute $T(x) \geq 0: \quad \mathbf{E}[w \cdot xy\, T(x)] < 0$

[Diakonikolas K. Tzamos Zarifis '20]

T(x) = Low-Degree O($\log(1/\epsilon)$) SoS Polynomial

[Diakonikolas Kane K. Tzamos Zarifis '21]

T(x) = Intersection of 4-Halfspaces

For the special case of Tsybakov noise obtain poly(d/eps)

Computationally Efficient Methods: Summary

- Debiasing Statistical Queries
- From Classification to Polynomial Regression
- Localization
- Iterative Peeling (for Massart Noise)
- Certificate Framework

- Good understanding of binary classification.
- The case of 3 or more classes is widely under-explored.