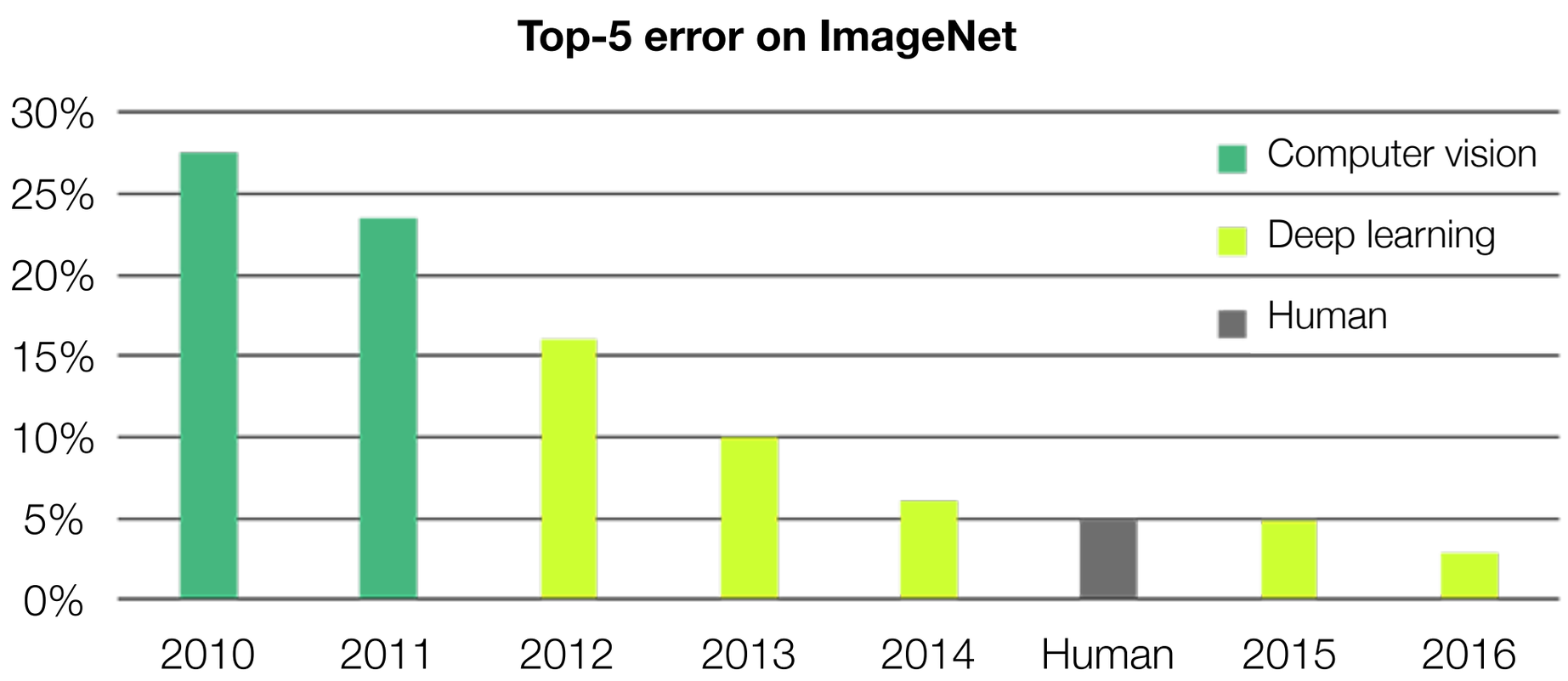
Understanding Black-box Predictions with Influence Functions

Pang Wei Koh
Percy Liang



[1] Defense Systems Information Analysis Center

Given a high-accuracy, black-box model, and a prediction from it, can we answer…

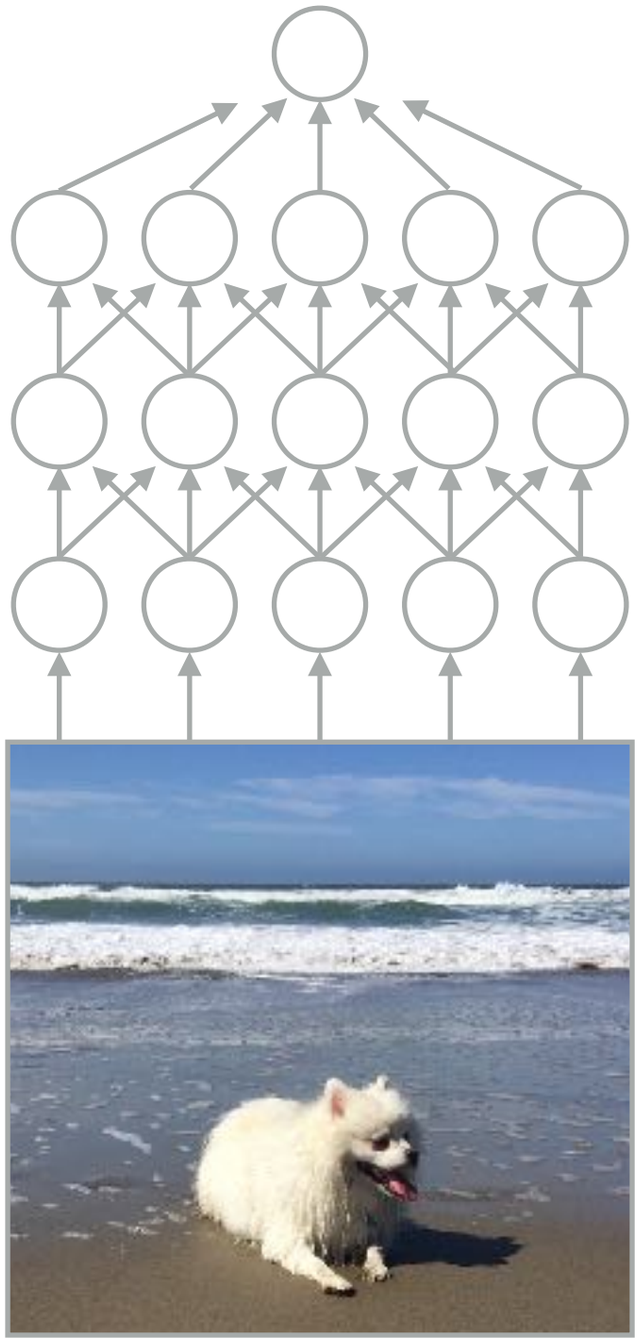
Why did the model make this prediction?

Why did the model make this prediction?

- Make better decisions [1]
- Improve the model [2]
- Discover new science [3]
- Provide end-users explanations [4]

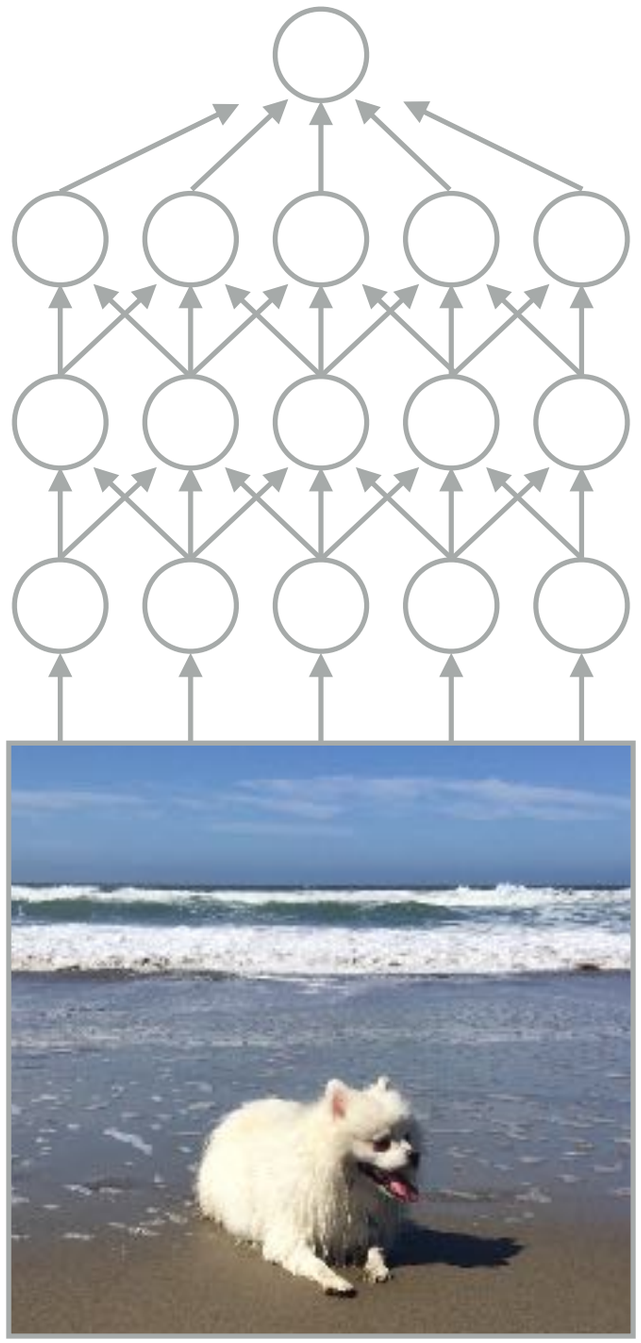
[1] Lakkaraju, Bach, and Leskovec, 2016
[2] Amershi et al., 2015
[3] Shrikumar, Greenside, and Kundaje, 2017
[4] Goodman & Flaxman, 2016

“Dog”



“Dog”

What inputs maximally activate these neurons? [1]



Can we represent this model with a simpler one? [2-3, 9]

Which part of the input was most responsible for this prediction? [4-9]

[1] Girshick et al., 2014
[2] Zeiler and Fergus, 2013
[3] Ribeiro, Singh, and Guestrin, 2016
[4] Bastani, Kim, and Bastani, 2017
[5] Simonyan, Vedaldi, and Zisserman, 2013
[6] Li, Monroe, and Jurafsky, 2016
[7] Shrikumar, Greenside, and Kundaje, 2017
[8] Sundararajan, Taly, and Yan, 2017
[9] Leino et al., 2018

Fish

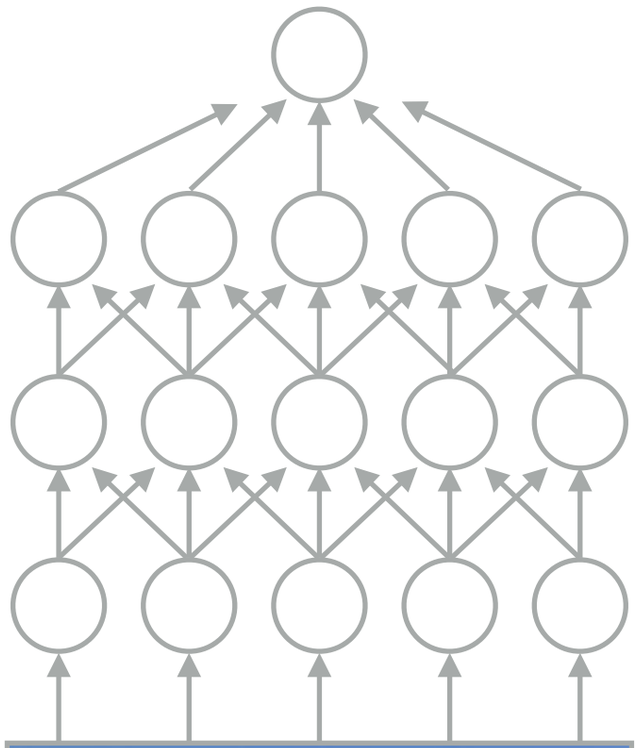
Dog

Dog

Training data

Training

“Dog”



Why did the model make this prediction?

Which training points were most responsible for this prediction?

Fish

Dog

Dog

Training data $z_1, z_2, \ldots, z_n$

Fish

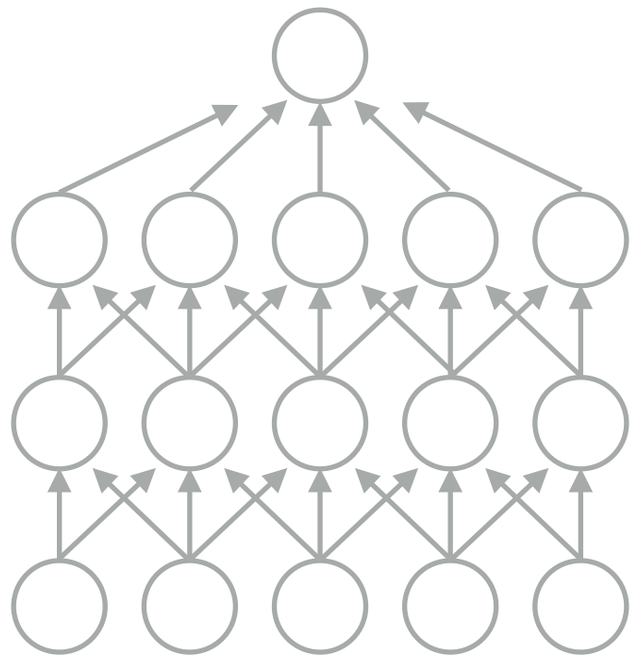
Dog

Dog

Training data $z_1, z_2, \ldots, z_n$

Pick $\hat{\theta}$ to minimize $\frac{1}{n}\sum_{i=1}^{n} L(z_i, \theta)$

“Dog”



$\hat{\theta}$

Fish

Dog

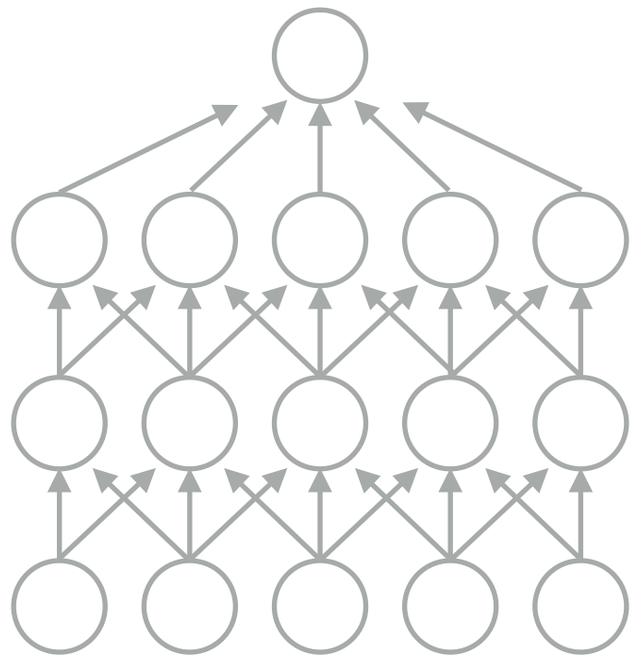
Dog

Training data $z_1, z_2, \ldots, z_n$

z_{train}

Pick $\hat{\theta}$ to minimize $\frac{1}{n}\sum_{i=1}^{n} L(z_i, \theta)$

“Dog”



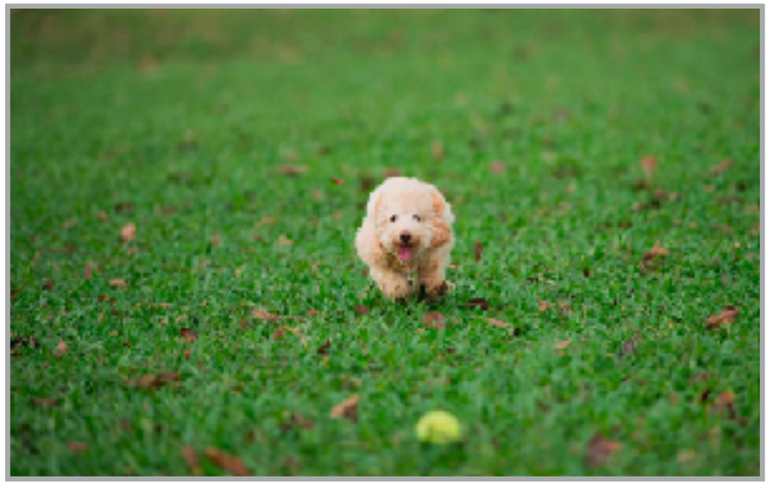
$\hat{\theta}$

Pick $\hat{\theta}$ to minimize $\frac{1}{n}\sum_{i=1}^{n} L(z_i, \theta)$

Fish

Dog

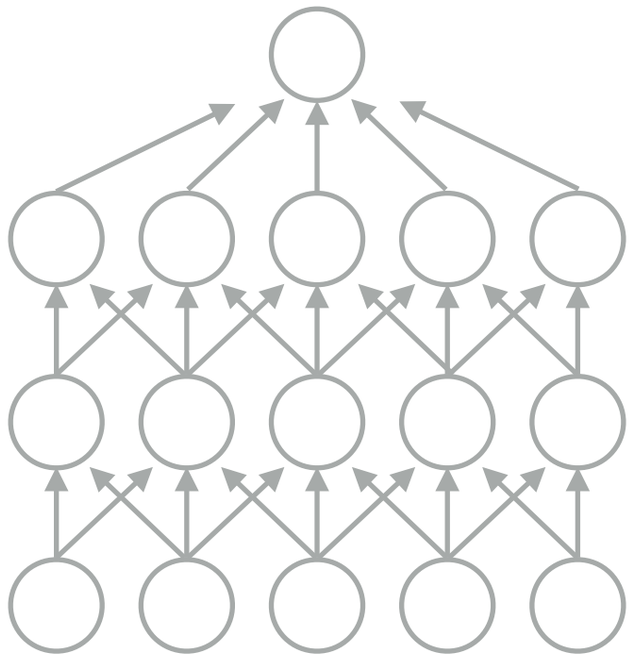
Dog



Training data $z_1, z_2, \ldots, z_n$

z_{train}

“Dog”

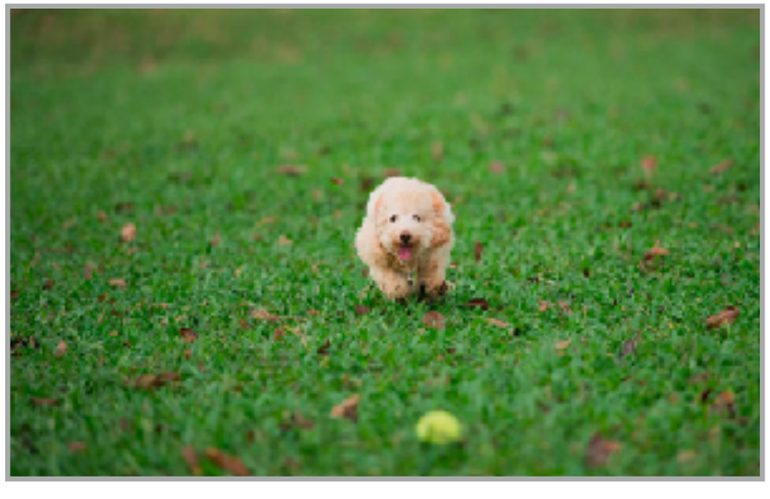


$\hat{\theta}$

Fish

Dog

Dog



Training data $z_1, z_2, \ldots, z_n$

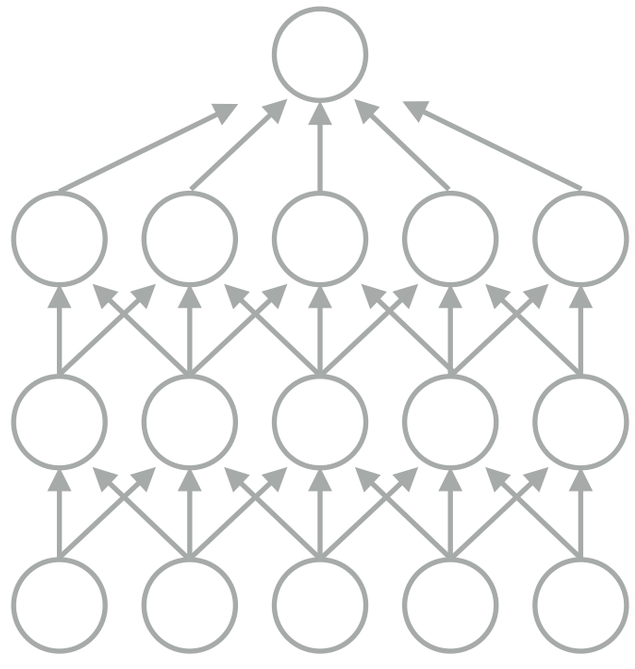
Pick $\hat{\theta}$ to minimize $\frac{1}{n}\sum_{i=1}^{n} L(z_i, \theta)$

Pick $\hat{\theta}_{\epsilon, z_{\text{train}}}$ to minimize

$\frac{1}{n}\sum_{i=1}^{n} L(z_i, \theta) + \epsilon L(z_{\text{train}}, \theta)$

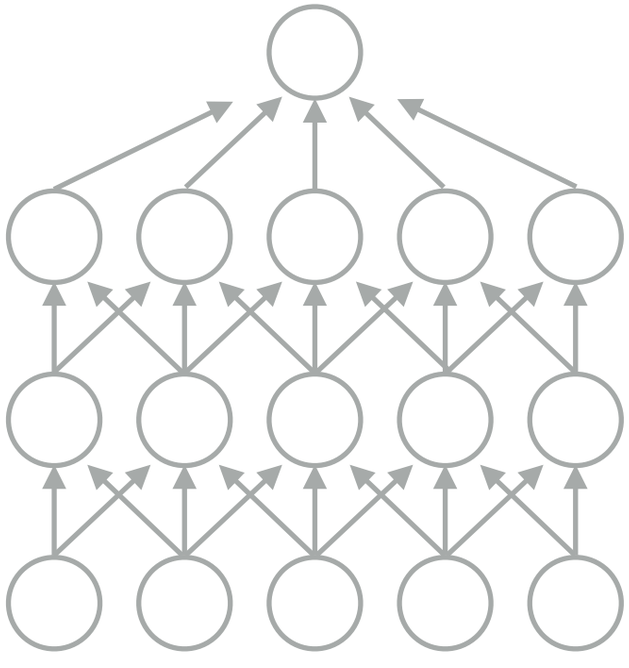
z_{train}

“Dog”



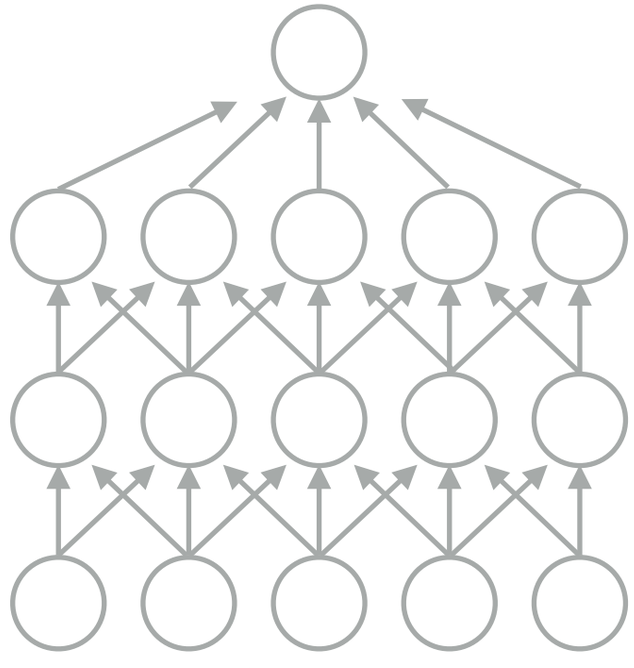
$\hat{\theta}_{\epsilon, z_{\text{train}}}$

“Dog” (79% confidence)



vs.

“Dog” (82% confidence)



$\hat{\theta}$

$\hat{\theta}_{\epsilon, z_{\text{train}}}$

Test input

Why did the model make this prediction?

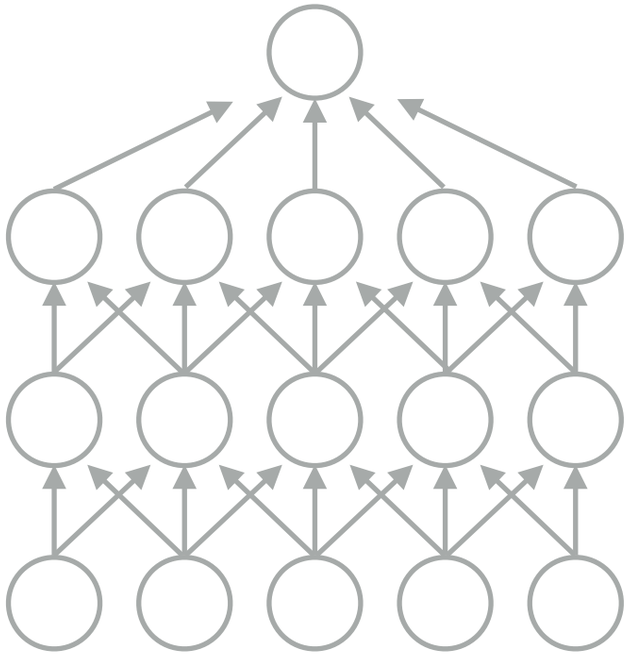
↓

Which training points were most responsible for this prediction?

↓

How would the prediction change if we upweighted each training point?

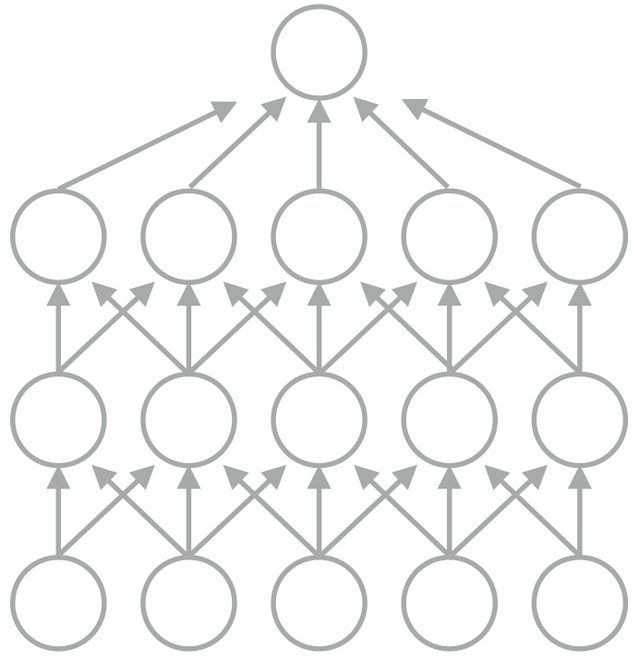
"Dog" (79% confidence)



$\hat{\theta}$

vs.

"Dog" (82% confidence)



$\hat{\theta}_{\epsilon, z_{\text{train}}}$

What is $L(z_{\text{test}}, \hat{\theta}_{\epsilon, z_{\text{train}}}) - L(z_{\text{test}}, \hat{\theta})$?

Motivation

> Influence functions

Applications

Conclusion

Influence functions

- Introduced in the 1970s in the field of robust statistics
  (e.g., Jaeckel, 1972; Cook, 1977; Cook and Weisberg, 1982)

Influence functions

- Introduced in the 1970s in the field of robust statistics
  (e.g., Jaeckel, 1972; Cook, 1977; Cook and Weisberg, 1982)

- Consider an estimator T that acts on a distribution F

- How much does T change if we perturb F?

Influence functions

- Goal: Measure change in $L(z_{\text{test}}, \hat{\theta}_{\epsilon, z_{\text{train}}})$ as we increase ϵ.

Influence functions

- Goal: Measure change in $L(z_{\text{test}}, \hat{\theta}_{\epsilon, z_{\text{train}}})$ as we increase ϵ.
- $\hat{\theta}_{\epsilon, z_{\text{train}}} \stackrel{\text{def}}{=} \arg\min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta) + \epsilon L(z_{\text{train}}, \theta)$.

Influence functions

- Goal: Measure change in $L(z_{\text{test}}, \hat{\theta}_{\epsilon, z_{\text{train}}})$ as we increase ϵ.
- $\hat{\theta}_{\epsilon, z_{\text{train}}} \stackrel{\text{def}}{=} \arg\min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta) + \epsilon L(z_{\text{train}}, \theta)$.
- Under smoothness assumptions,

$$\mathcal{I}_{\text{up,loss}}(z_{\text{train}}, z_{\text{test}}) \stackrel{\text{def}}{=} \left. \frac{dL(z_{\text{test}}, \hat{\theta}_{\epsilon, z_{\text{train}}})}{d\epsilon} \right|_{\epsilon=0}$$

Influence functions

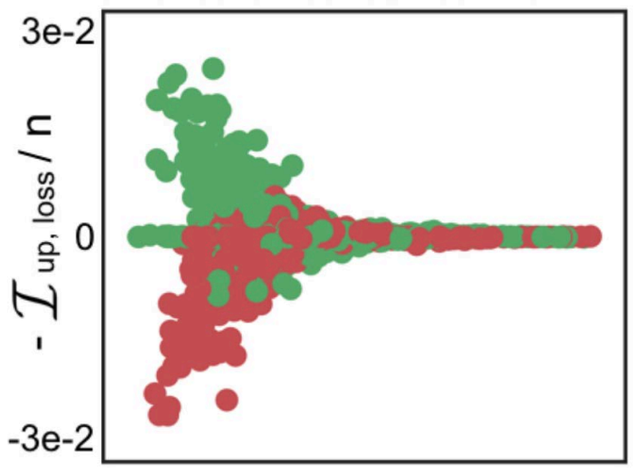
- Goal: Measure change in $L(z_{\text{test}}, \hat{\theta}_{\epsilon, z_{\text{train}}})$ as we increase ϵ.
- $\hat{\theta}_{\epsilon, z_{\text{train}}} \stackrel{\text{def}}{=} \arg\min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta) + \epsilon L(z_{\text{train}}, \theta)$.
- Under smoothness assumptions,

$$\mathcal{I}_{\text{up,loss}}(z_{\text{train}}, z_{\text{test}}) \stackrel{\text{def}}{=} \left. \frac{dL(z_{\text{test}}, \hat{\theta}_{\epsilon, z_{\text{train}}})}{d\epsilon} \right|_{\epsilon=0}$$
$$= -\nabla_\theta L(z_{\text{test}}, \hat{\theta})^\top H_{\hat{\theta}}^{-1} \nabla_\theta L(z_{\text{train}}, \hat{\theta}),$$

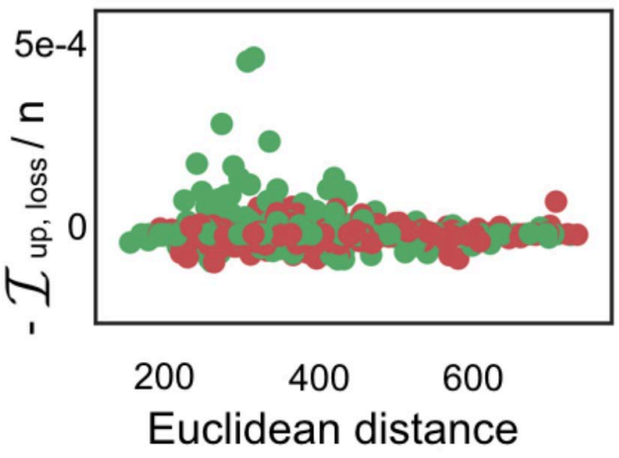
- where $H_{\hat{\theta}} \stackrel{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} \nabla_\theta^2 L(z_i, \hat{\theta})$.

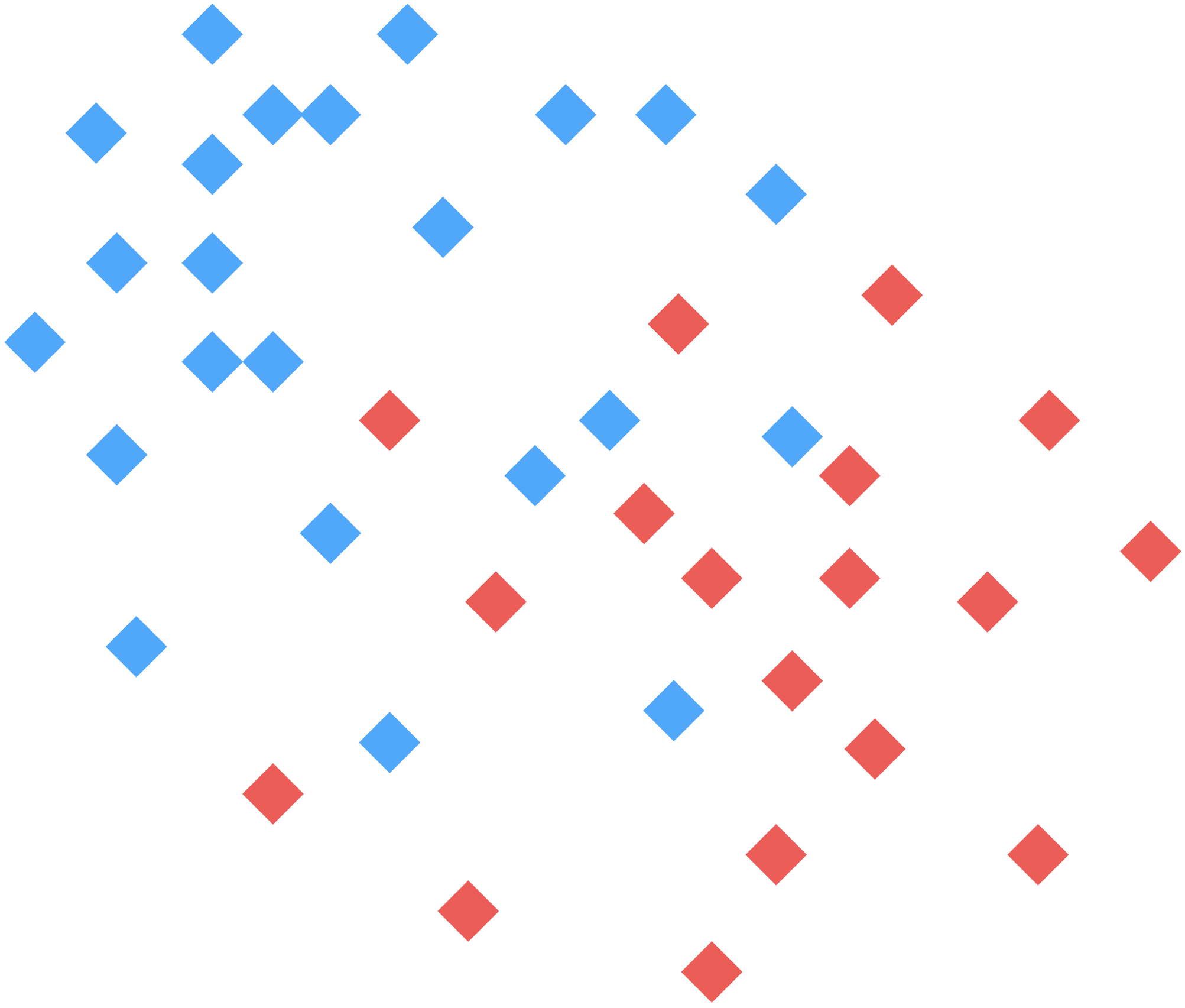
Test image

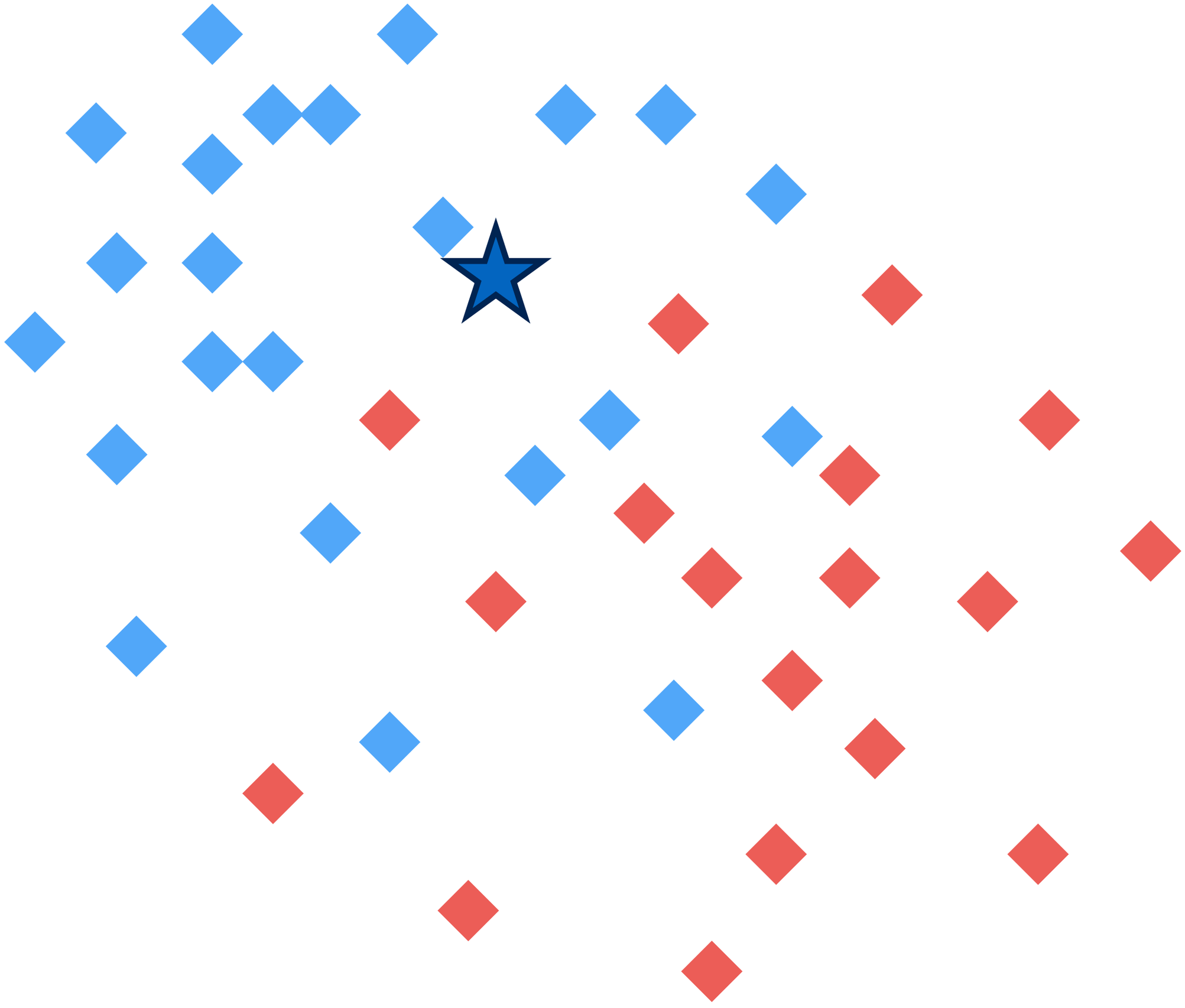
RBF SVM (raw pixels)

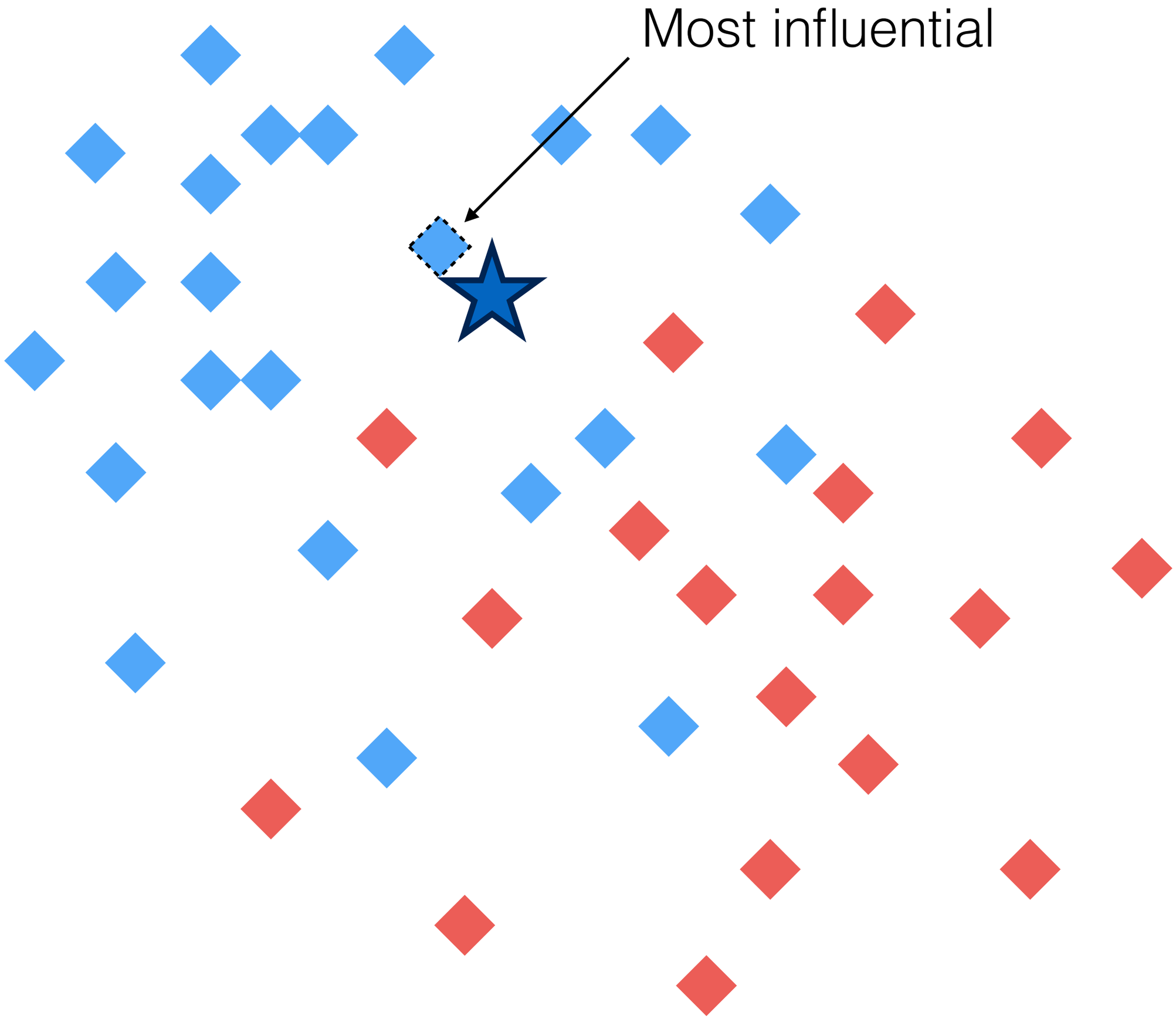


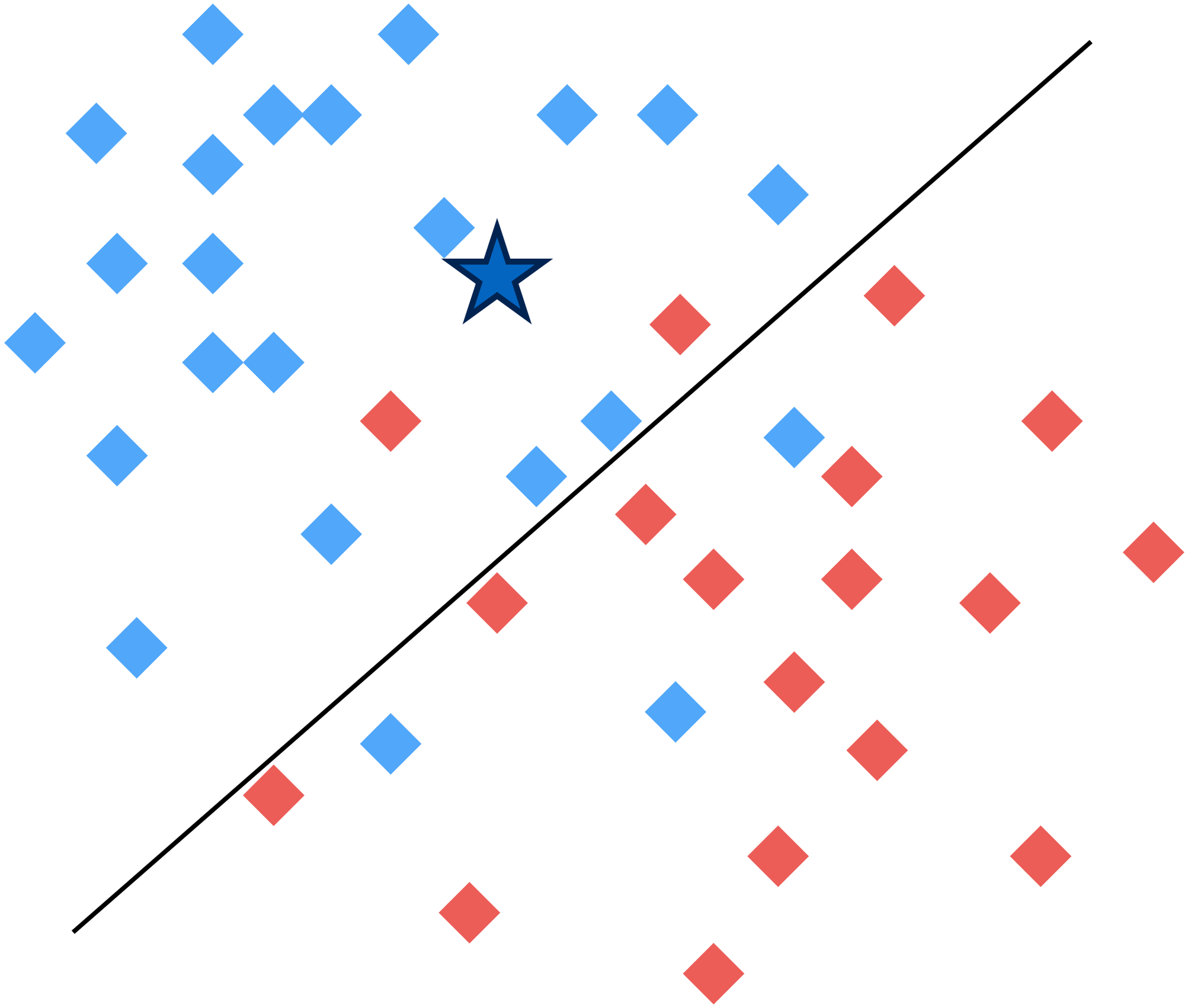
Logistic regression (Inception features)

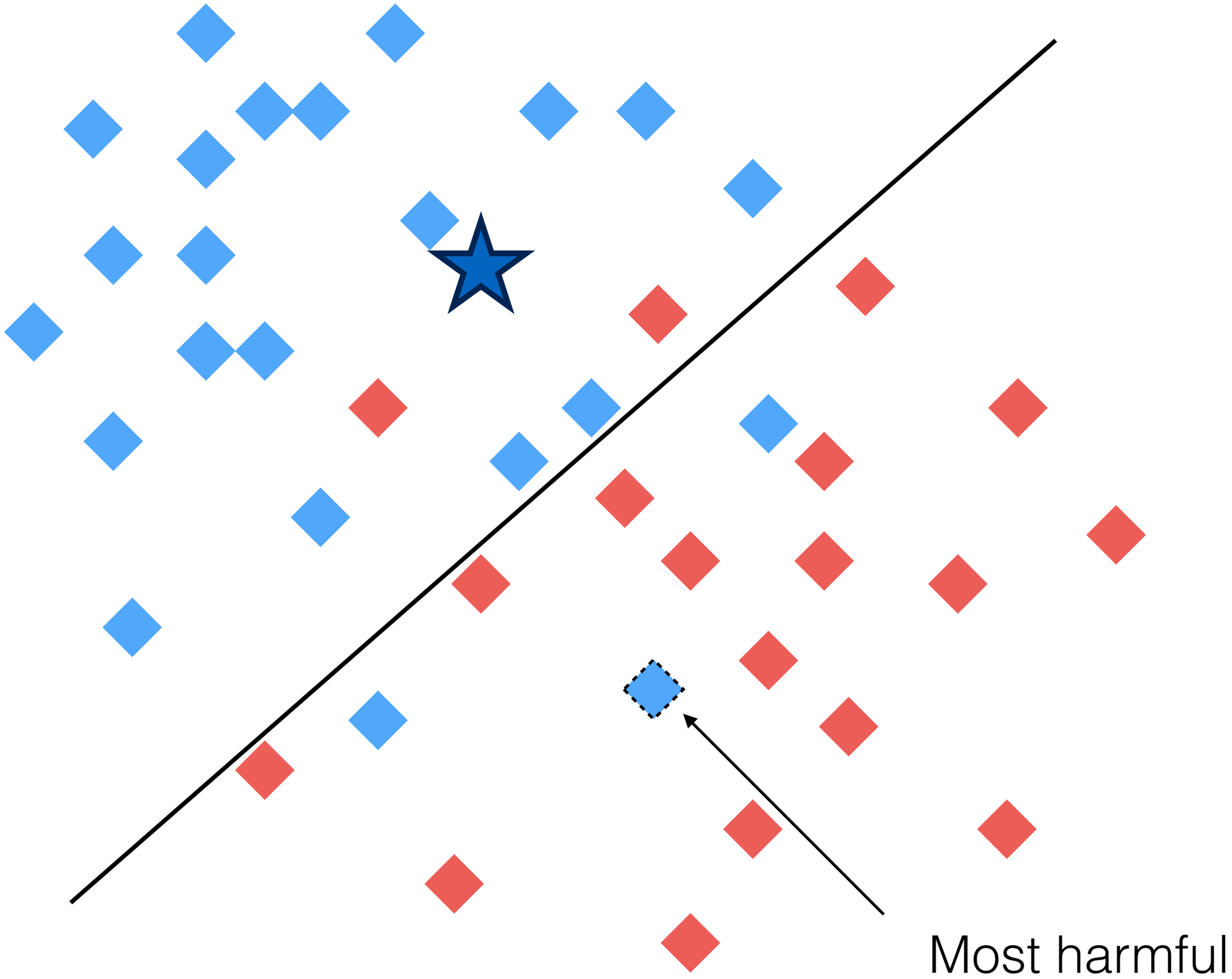


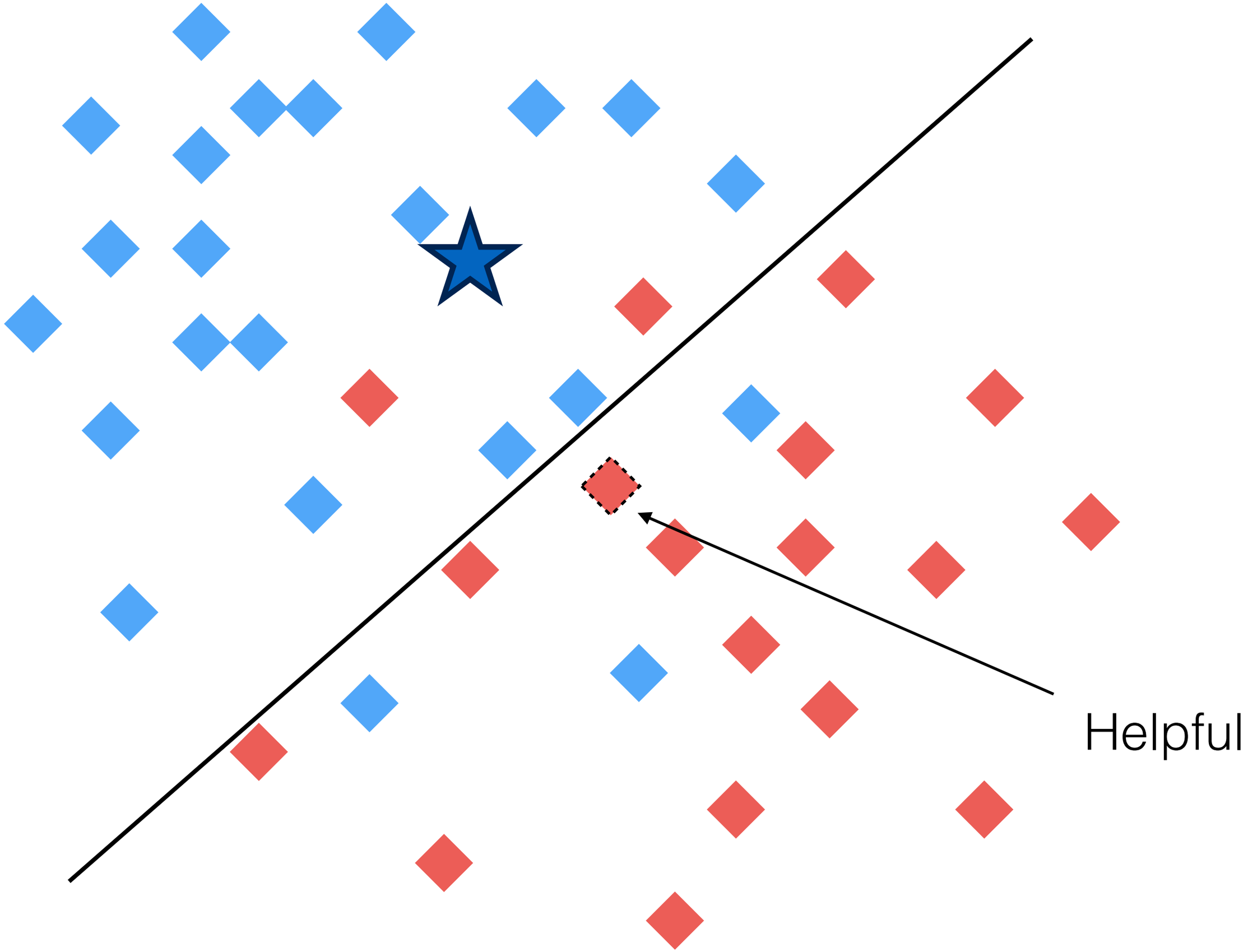





Most influential

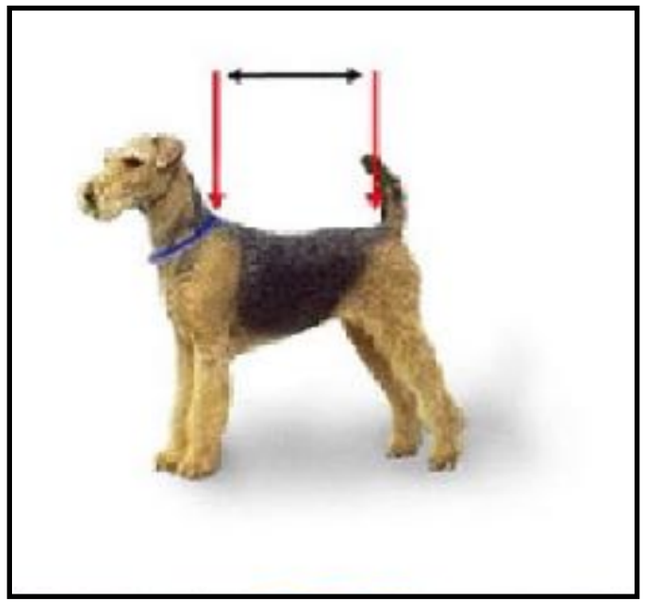



Most harmful


Helpful

Test image

Helpful dog



Motivation

Influence functions

> Applications

Conclusion

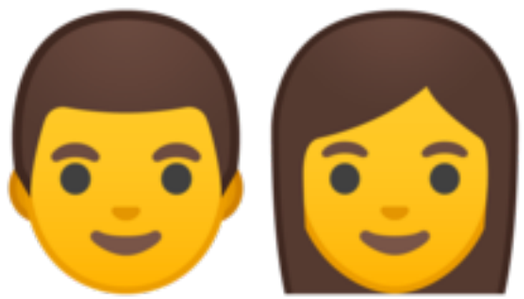
Application 1

Debugging model errors

Debugging model errors

- If a model makes a mistake, can we find out why?
- Case study: hospital re-admission (logistic regression, 20K patients, 127 features)

Debugging model errors

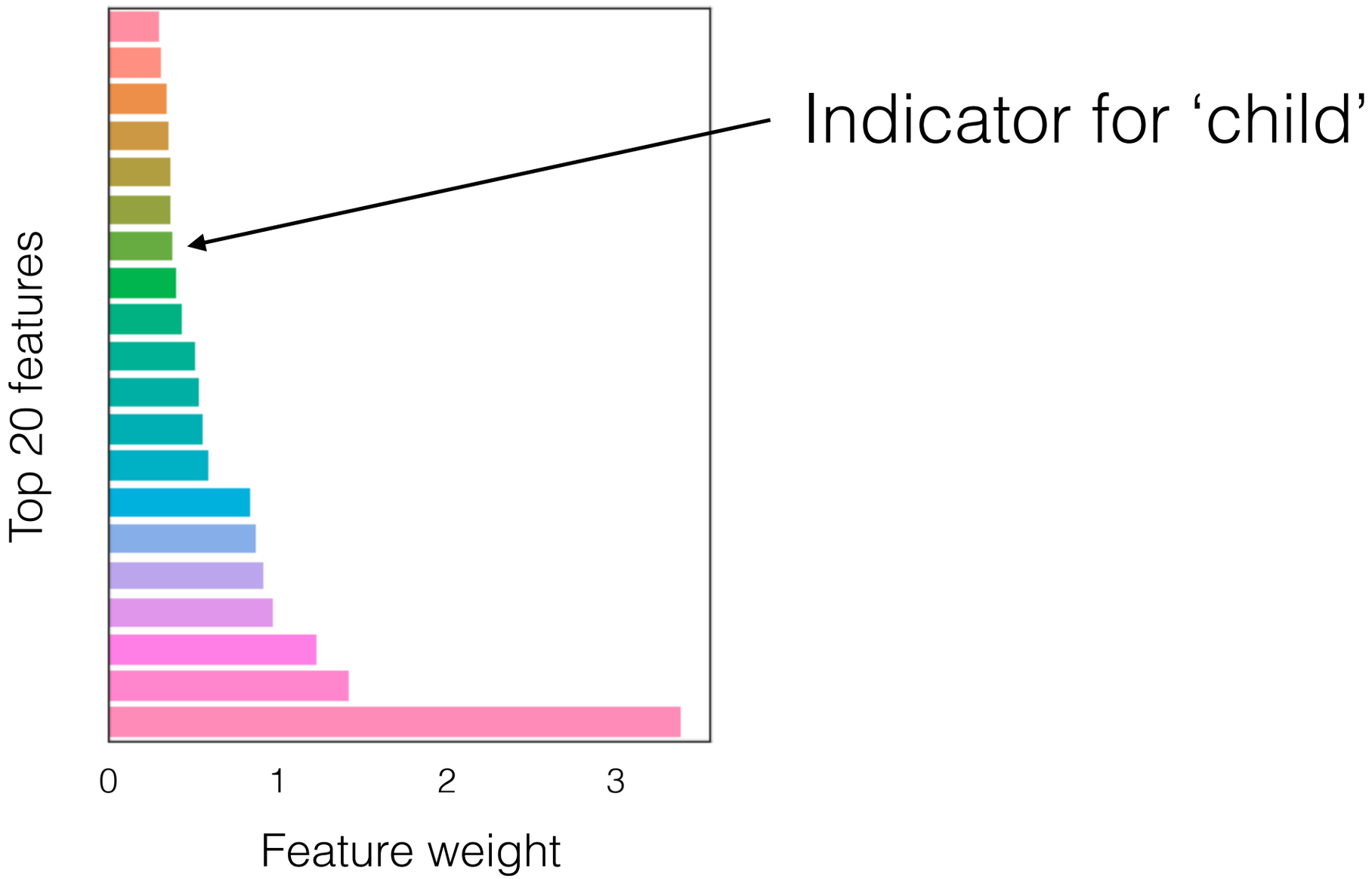
		Original		**Modified**
Healthy + re-admitted adults		~20k	**same** →	~20k
Healthy children		21	**-20** →	1
Re-admitted children		3	**same** →	3

Debugging model errors

True test label: Healthy
Model predicts: Re-admitted

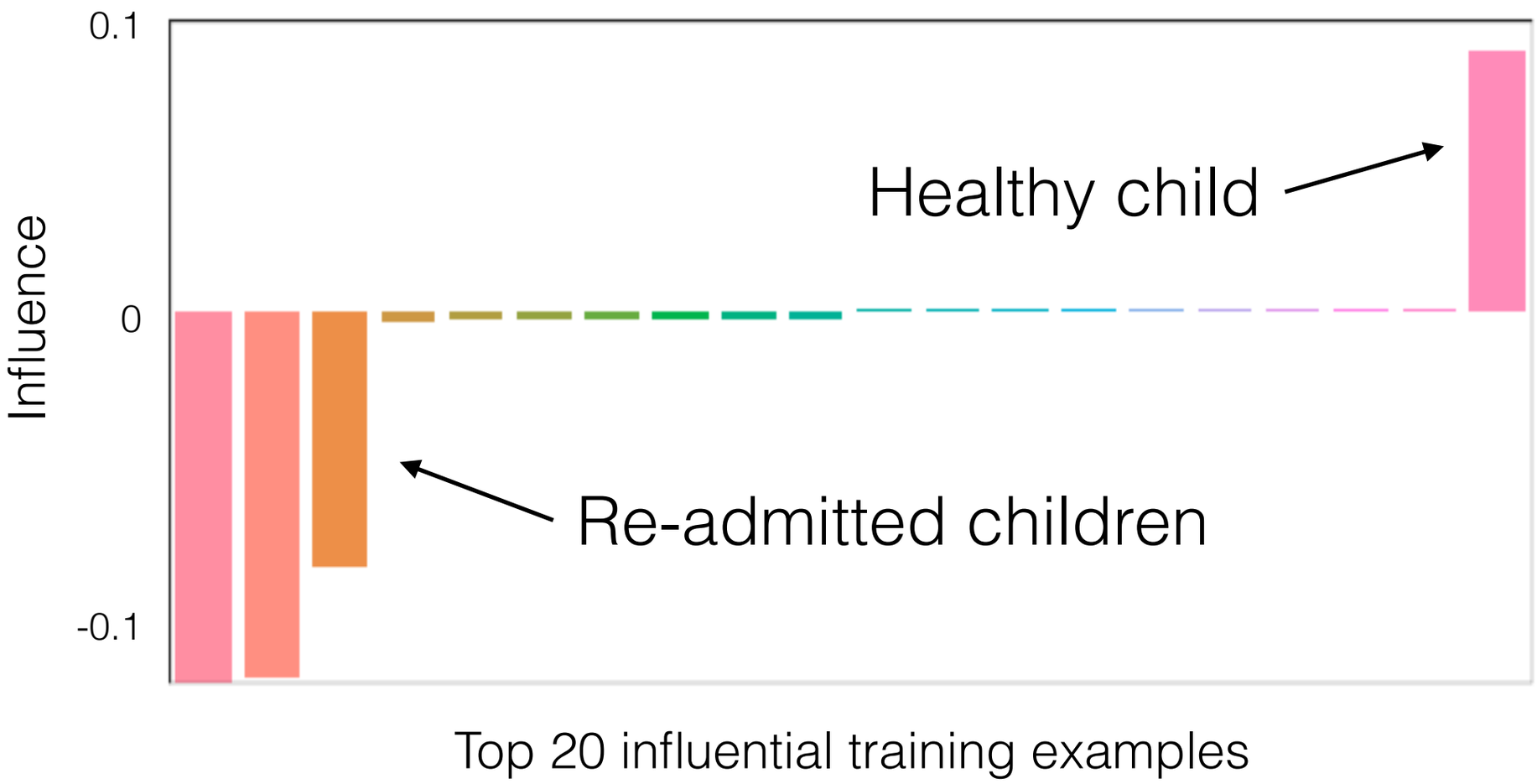
Debugging model errors

True test label: Healthy
Model predicts: Re-admitted



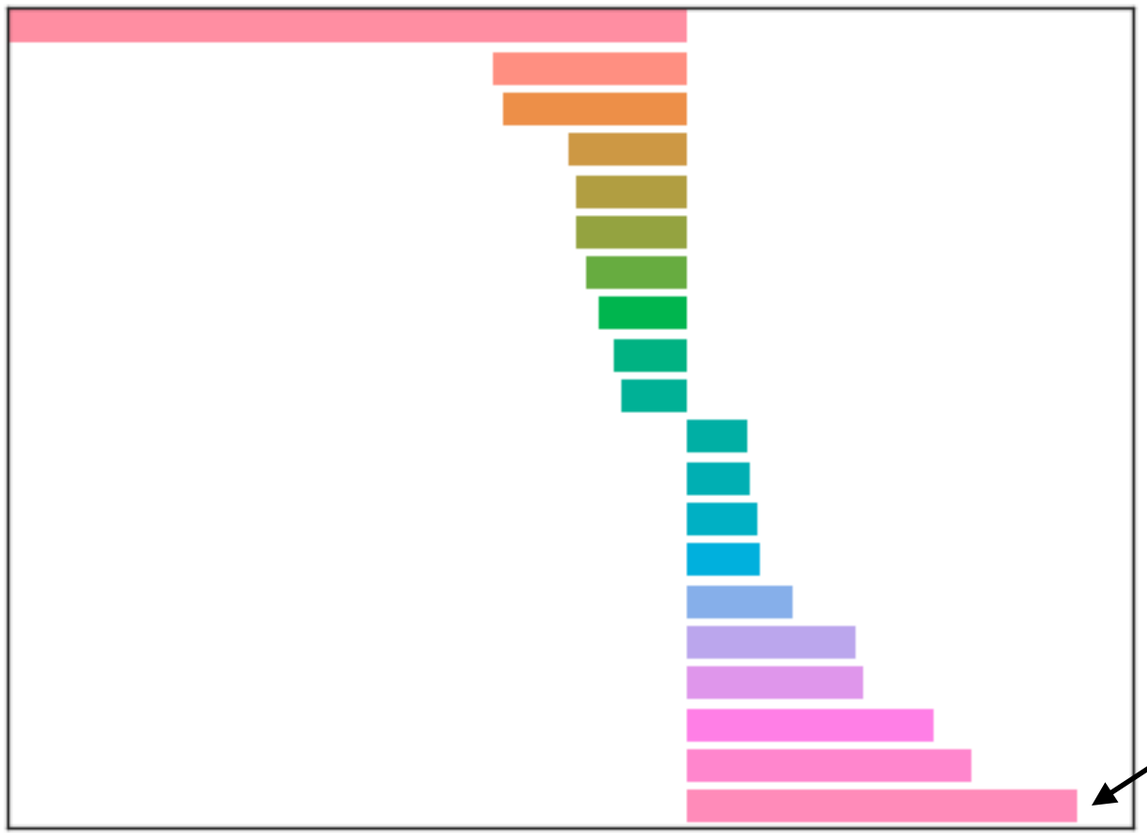
Debugging model errors

True test label: Healthy
Model predicts: Re-admitted



Debugging model errors

True test label: Healthy
Model predicts: Re-admitted



Application 2

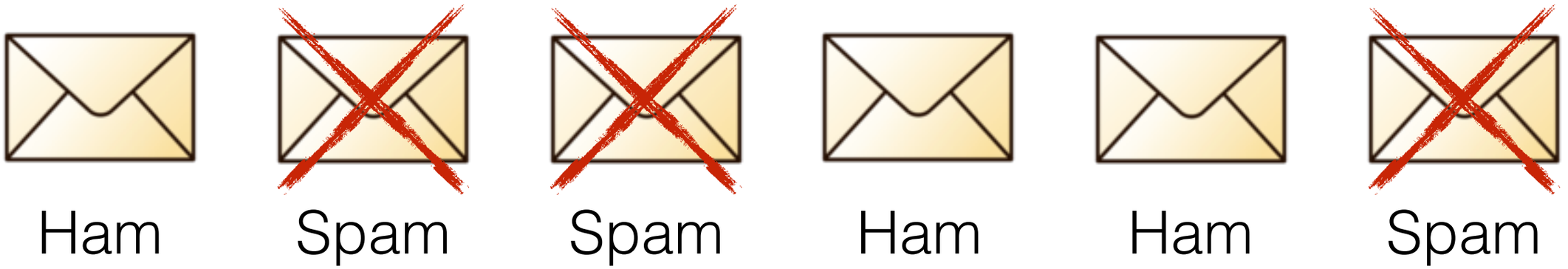
Fixing training data

Fixing training data

- Setup: training labels are noisy, and we have a small budget to manually inspect them
- Can we prioritize which labels to try to fix?

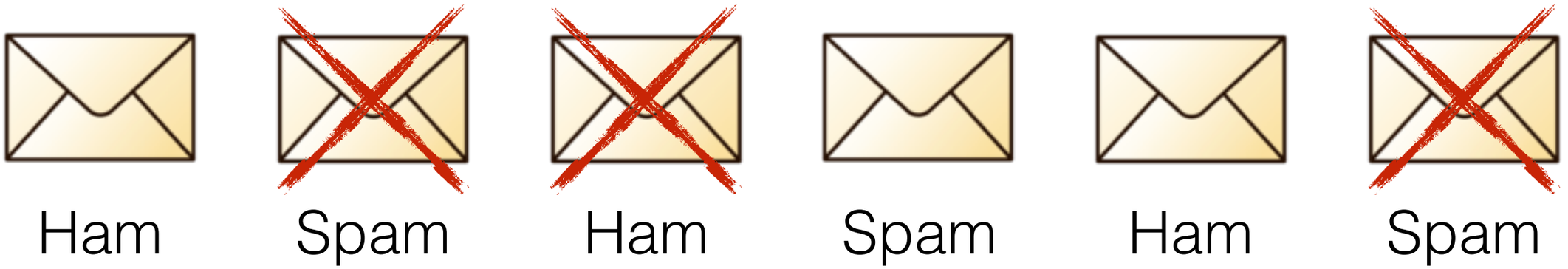
Fixing training data

- Setup: training labels are noisy, and we have a small budget to manually inspect them
- Can we prioritize which labels to try to fix?



Fixing training data

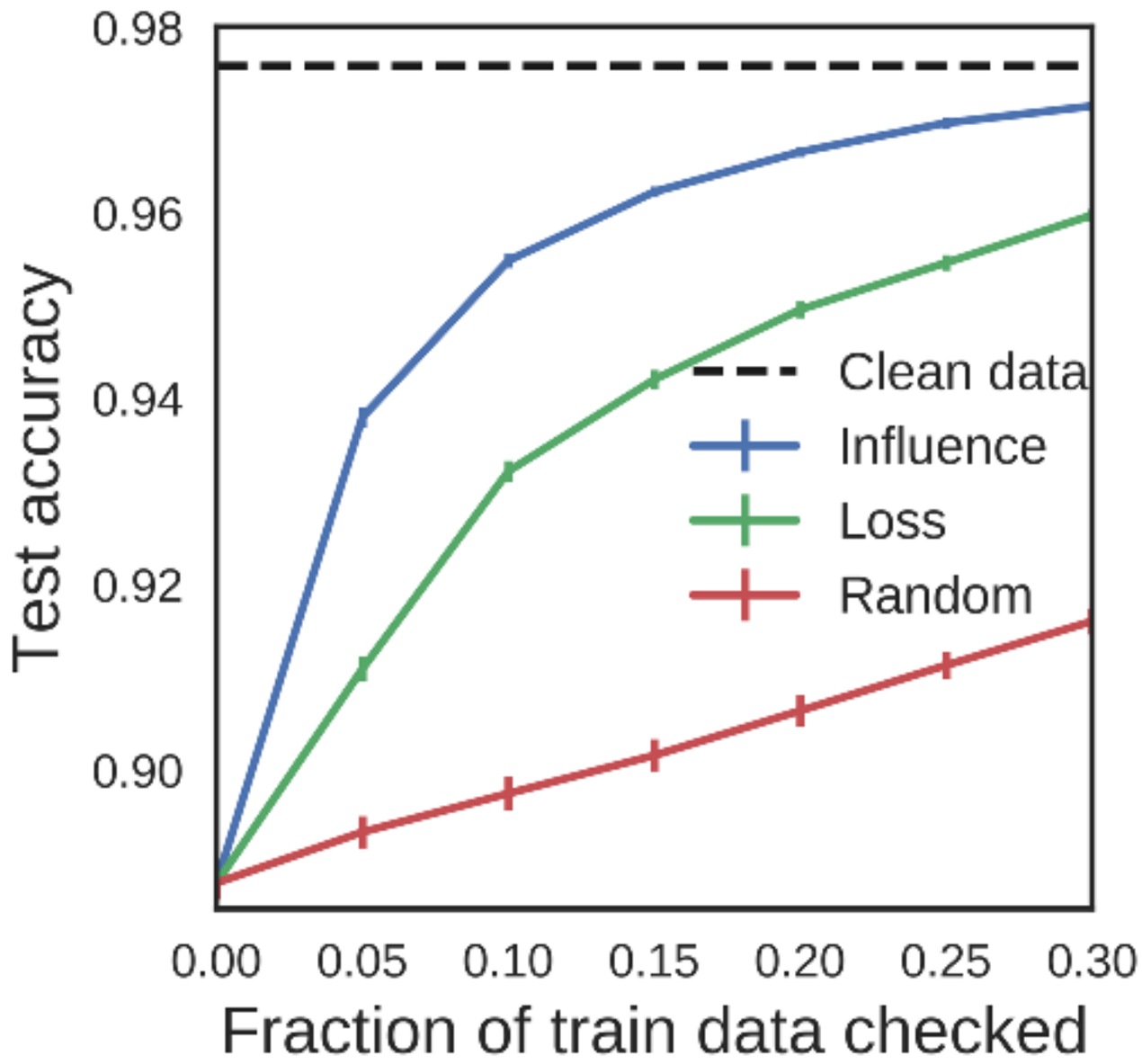
- Setup: training labels are noisy, and we have a small budget to manually inspect them
- Can we prioritize which labels to try to fix?



Fixing training data

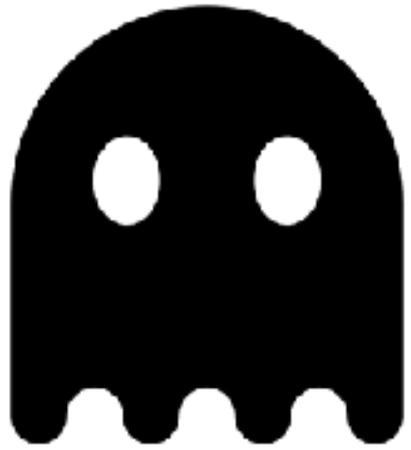
- Key idea: if a training point is not influential, don’t waste effort checking it

Fixing training data

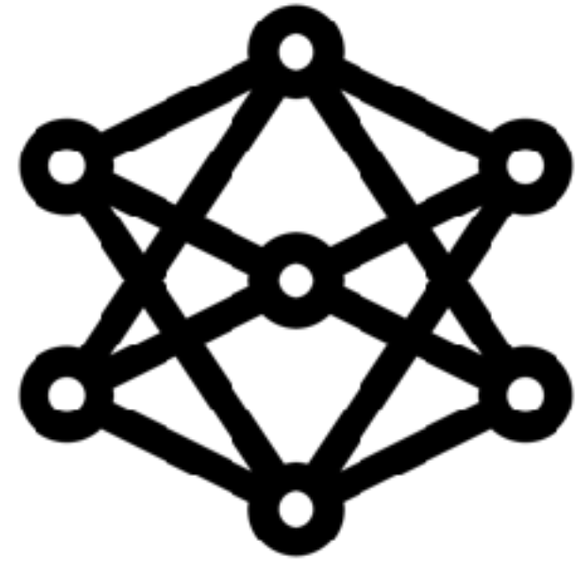


Application 3

Adversarial training examples



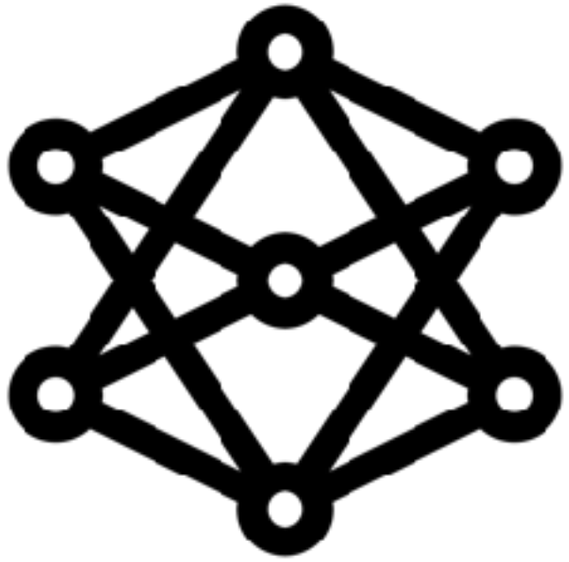
Test data



Correct prediction

Model

Test data

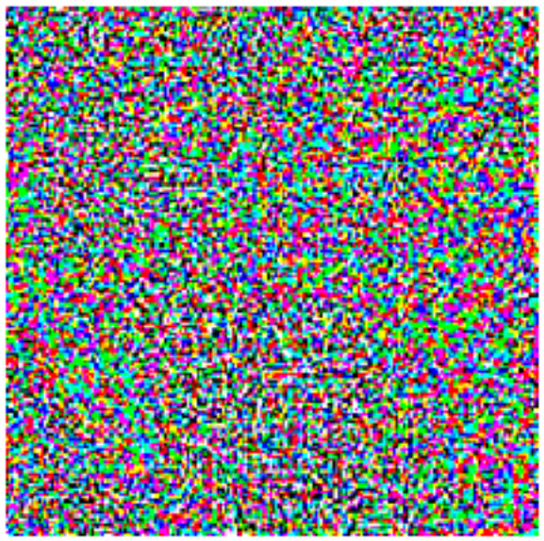


Model

Wrong prediction

“panda”
57.7% confidence

$+ .007 \times$



$=$

“gibbon”
99.3% confidence

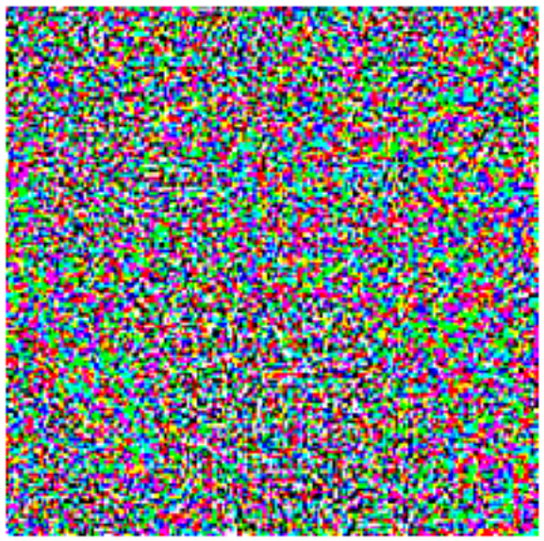
Image from Goodfellow, Shlens, Szegedy, 2015
Original demonstration from Szegedy et al., 2013

Adversarial test examples

“panda”

57.7% confidence

$+ .007 \times$



$=$

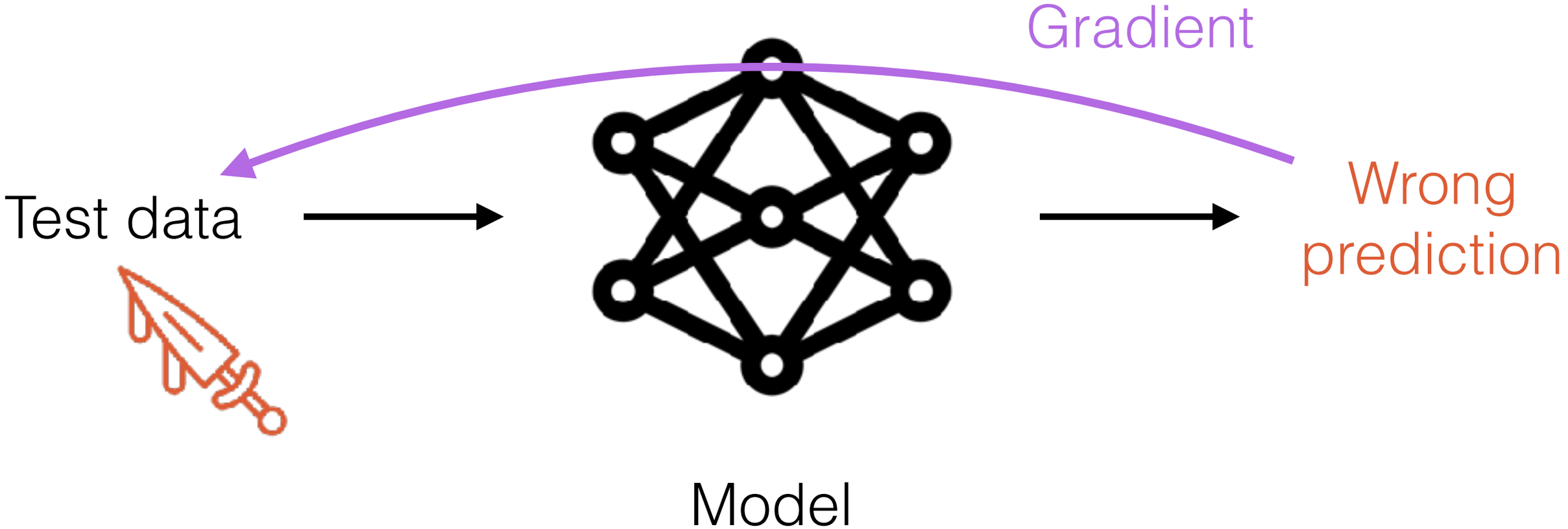
“gibbon”

99.3% confidence

Image from Goodfellow, Shlens, Szegedy, 2015
Original demonstration from Szegedy et al., 2013

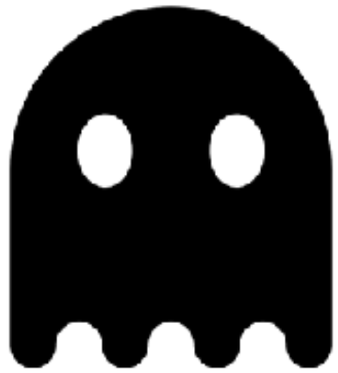
Adversarial test examples

Follow the gradient of the test loss w.r.t. test features (to increase loss) [1]



[1] Goodfellow, Shlens, Szegedy, 2015

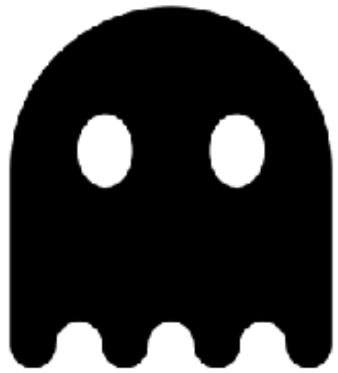
We have adversarial test examples.
Can we create adversarial **training** examples?



Adversarial training examples

Follow the gradient of the test loss w.r.t. train features

Influence functions help us calculate this gradient

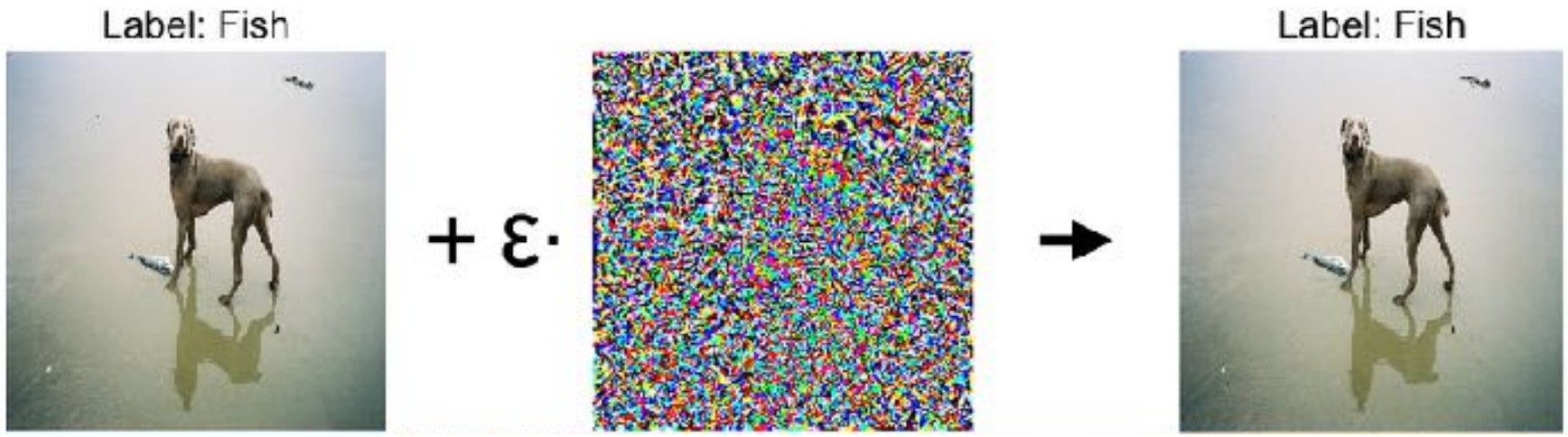


Adversarial training examples

- Setup: dog vs. fish classification, logistic regression on top of Inception features

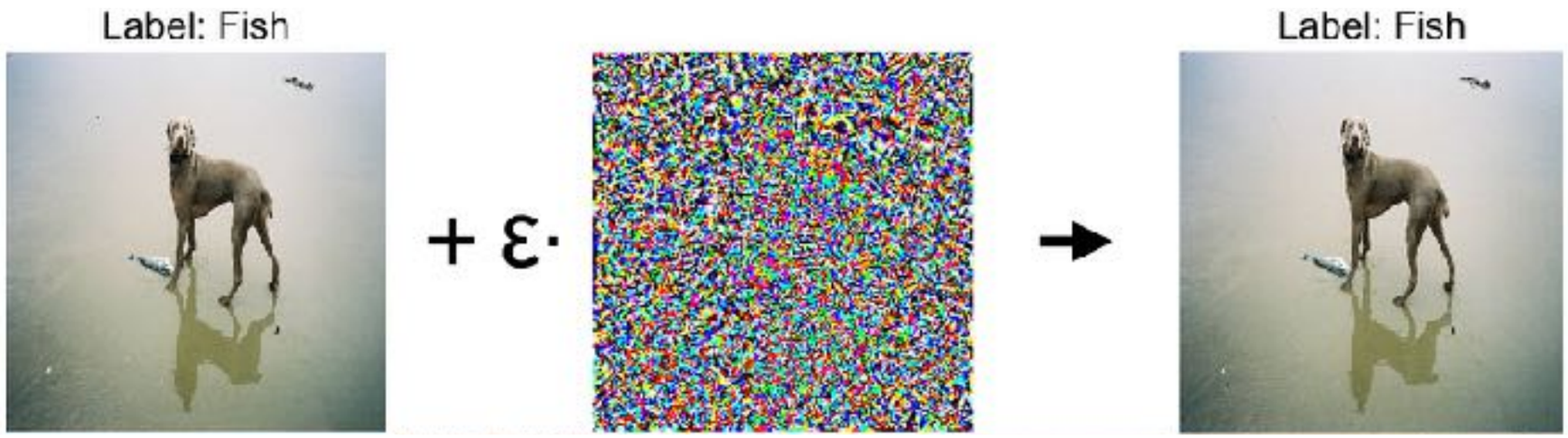
Adversarial training examples

A small perturbation to one **training** example:



Adversarial training examples

A small perturbation to one **training** example:



Can change multiple **test** predictions:

Orig (confidence):	Dog (97%)	Dog (98%)	Dog (98%)	Dog (99%)	Dog (98%)
New (confidence):	Fish (97%)	Fish (93%)	Fish (87%)	Fish (63%)	Fish (52%)

Three observations

1. Ambiguous examples are good attack vectors

Label: fish

Three observations

1. Ambiguous examples are good attack vectors

Label: fish

Three observations

1. Ambiguous examples are good attack vectors

2. Small change in pixels but large change in feature space

Three observations

1. Ambiguous examples are good attack vectors

2. Small change in pixels but large change in feature space

3. Attack makes model overfit to specific test examples (n ~ d)

Motivation

Influence functions

Applications

> Conclusion

Why did the model make this prediction?

↓

Which training points were most responsible for this prediction?

↓

How would the prediction change if we upweighted each training point?

Thank you

Github: https://bit.ly/gt-influence
CodaLab: https://bit.ly/cl-influence
Paper: https://arxiv.org/abs/1703.04730

pangwei@cs.stanford.edu

Percy Liang

Koda

This presentation uses images from the Noun Project:

Question by Valeriy | Box by Rockicon | Magnifying Glass by il Capitano | services by IconsGhost | Ghost by Bakunetsu Kaito
Neural Network by Knut M. Synstad | Poisoned Dagger by Ben Davis | world by Aleksandr Vector | Shield by Nikita Kozin