

Fedor Ratnikov



Deep Learning Interpretability

2021



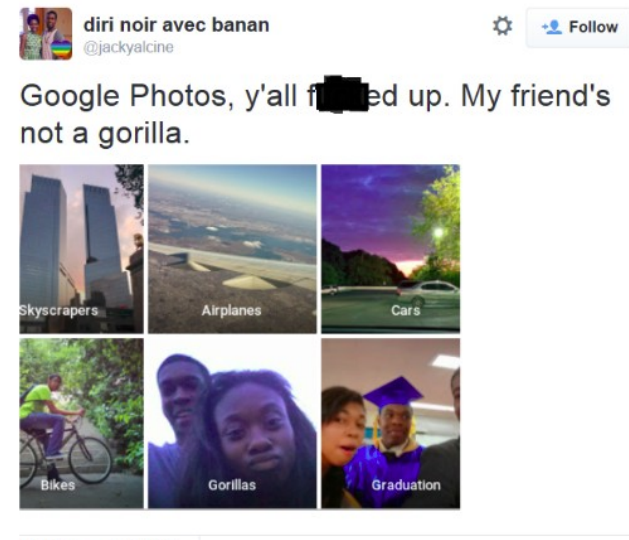
Yandex



EPFL



ML mistakes have a cost



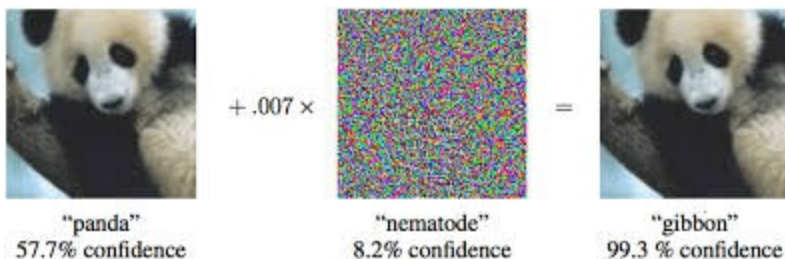
Uber self-driving car crashes during US tests

3. Robot Injured a child

A so-called "crime fighting robot," created by the platform, crashed into a child in a Silicon Valley mall in July, injuring the 16-year-old.

Chinese billionaire's face identified as jaywalker

Traffic police in major Chinese cities are using AI to address jaywalking. They deploy smart cameras using facial recognition techniques at intersections to detect and identify jaywalkers, whose partially obscured faces are often misidentified.



The question of trust

How can I explain my model's prediction?

Why did it make this decision/mistake?

What features does it rely on?

The question of trust

How can I explain my model's prediction?

Why did it make this decision/mistake?

What features does it rely on?

Is my model certain about what it says?

Is there something wrong with this input?

Can I rely on this prediction?

The question of trust

How can I explain my model's prediction?

Why did it make this decision/mistake?

What features does it rely on?

Is my model certain about what it says?

Is there something wrong with this input?

Can I rely on this prediction?

Can I trust this data?

Is something missing?

Is there any bias?

The question of trust

How can I explain my model's prediction?

Why did it make this decision/mistake?

What features does it rely on?

Is my model certain about what it says?

Is there something wrong with this input?

Can I rely on this prediction?

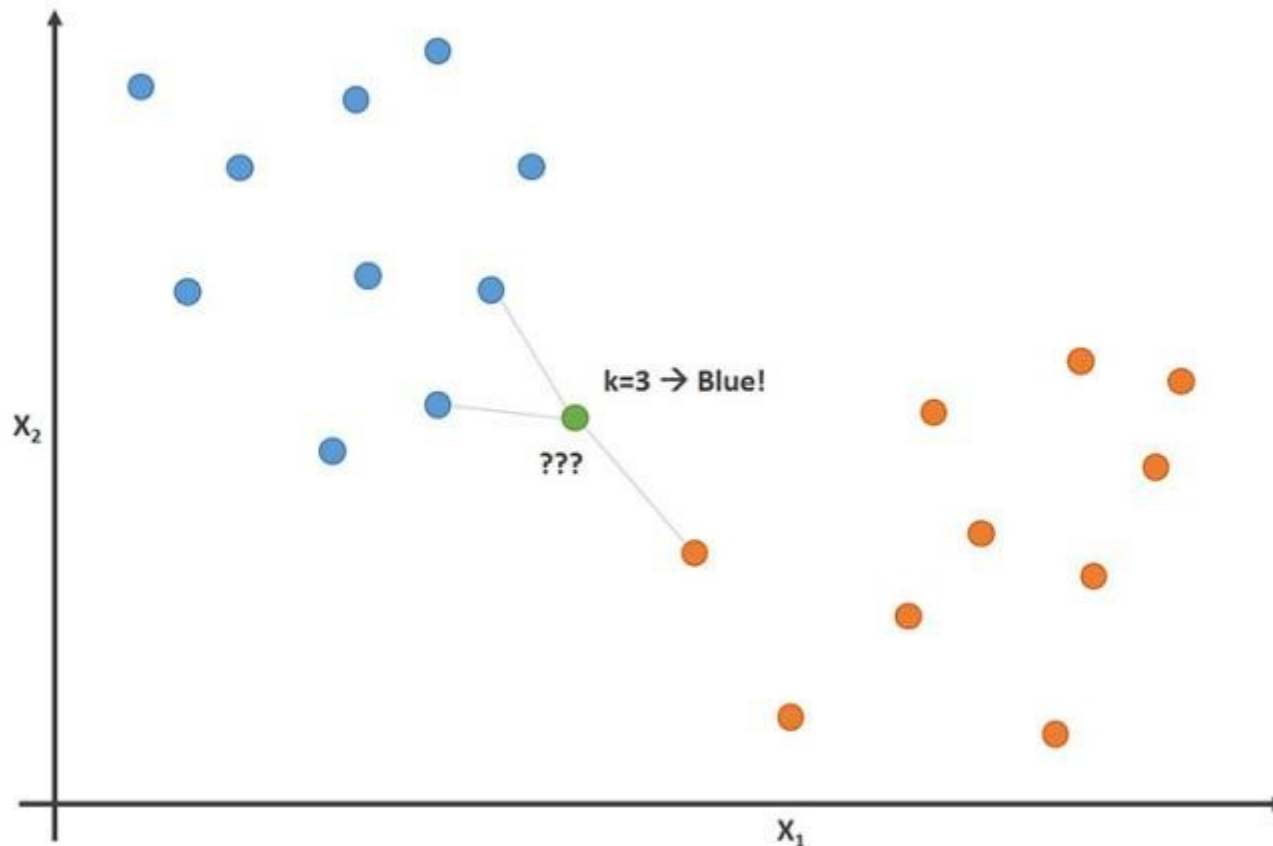
Can I trust this data?

Is something missing?

Is there any bias?

What is interpretable?

Simple stuff like **K Nearest Neighbors**



What is interpretable?

Simple stuff like Linear models

Example #3 of 6

True Class:  Atheism

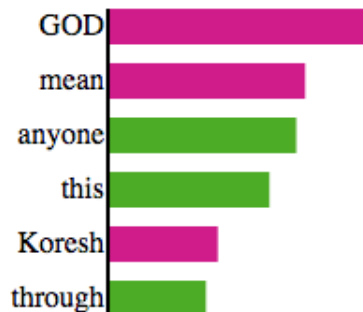
[Instructions](#)

[Previous](#)

[Next](#)

Algorithm 1

Words that A1 considers important:



Predicted:

 Atheism

Prediction correct:

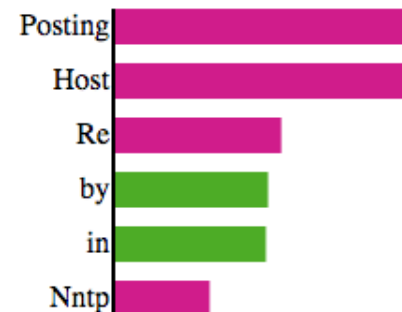


Document

From: pauld@verdix.com (Paul Durbin)
Subject: Re: DAVID CORESH IS! **GOD!**
Nntp-Posting-Host: sarge.hq.verdix.com
Organization: Verdix Corp
Lines: 8

Algorithm 2

Words that A2 considers important:



Predicted:

 Atheism

Prediction correct:

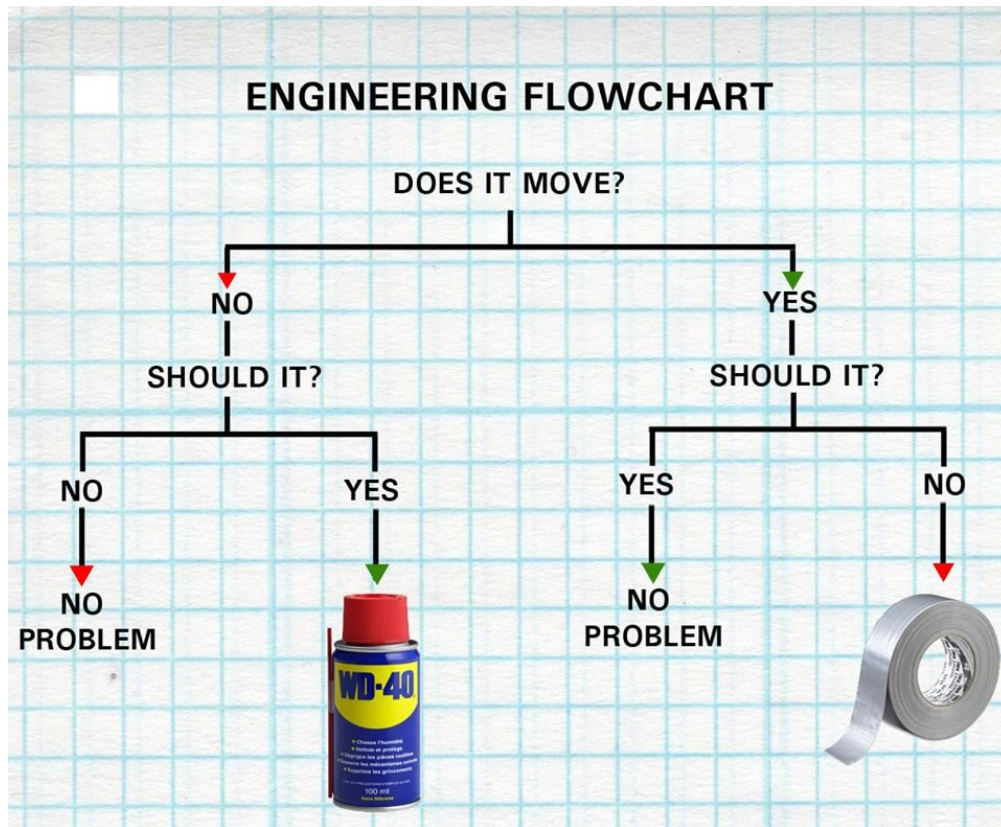


Document

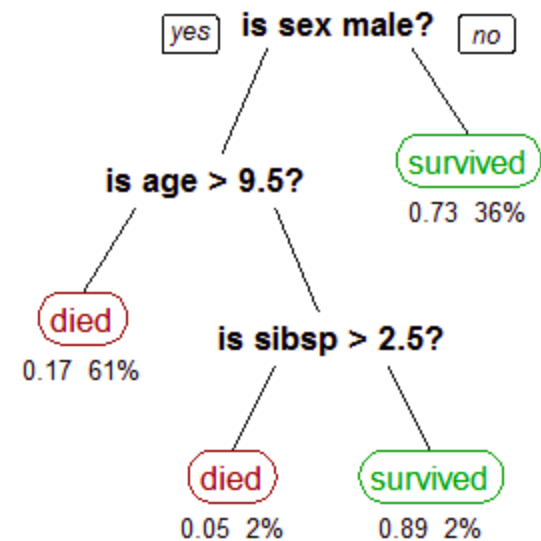
From: pauld@verdix.com (Paul Durbin)
Subject: **Re:** DAVID CORESH IS! GOD!
Nntp-Posting-Host: sarge.hq.verdix.com
Organization: Verdix Corp
Lines: 8

What is interpretable?

Simple stuff like **Decision Trees**



Survival on Titanic



What is interpretable?

Neural networks are not naturally interpretable

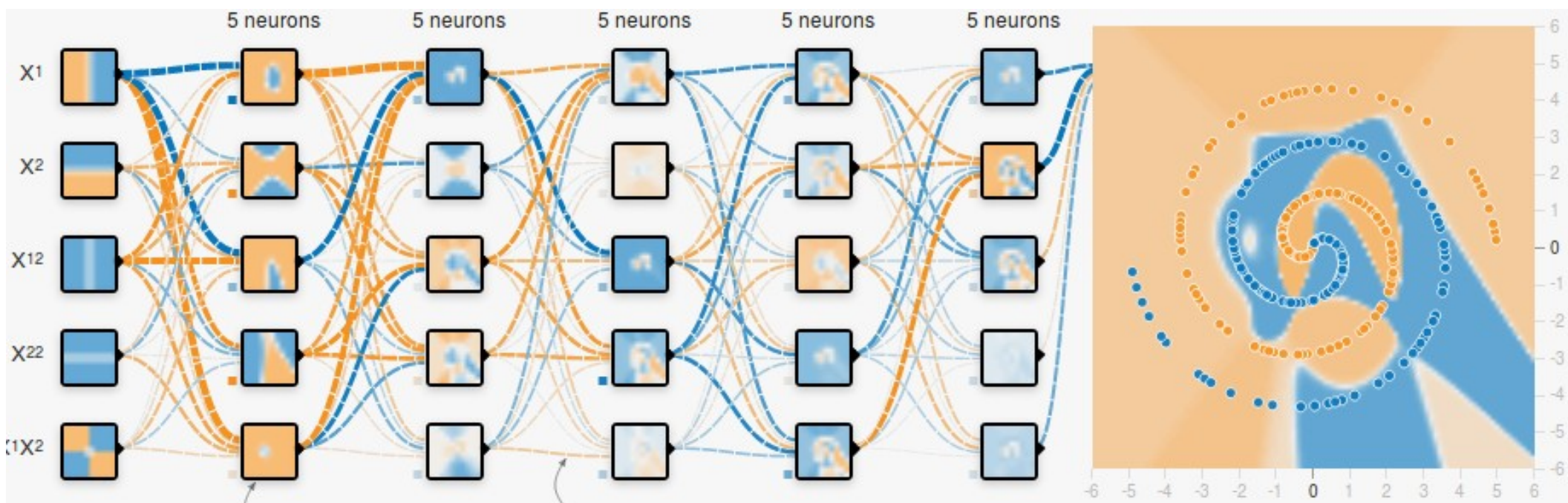
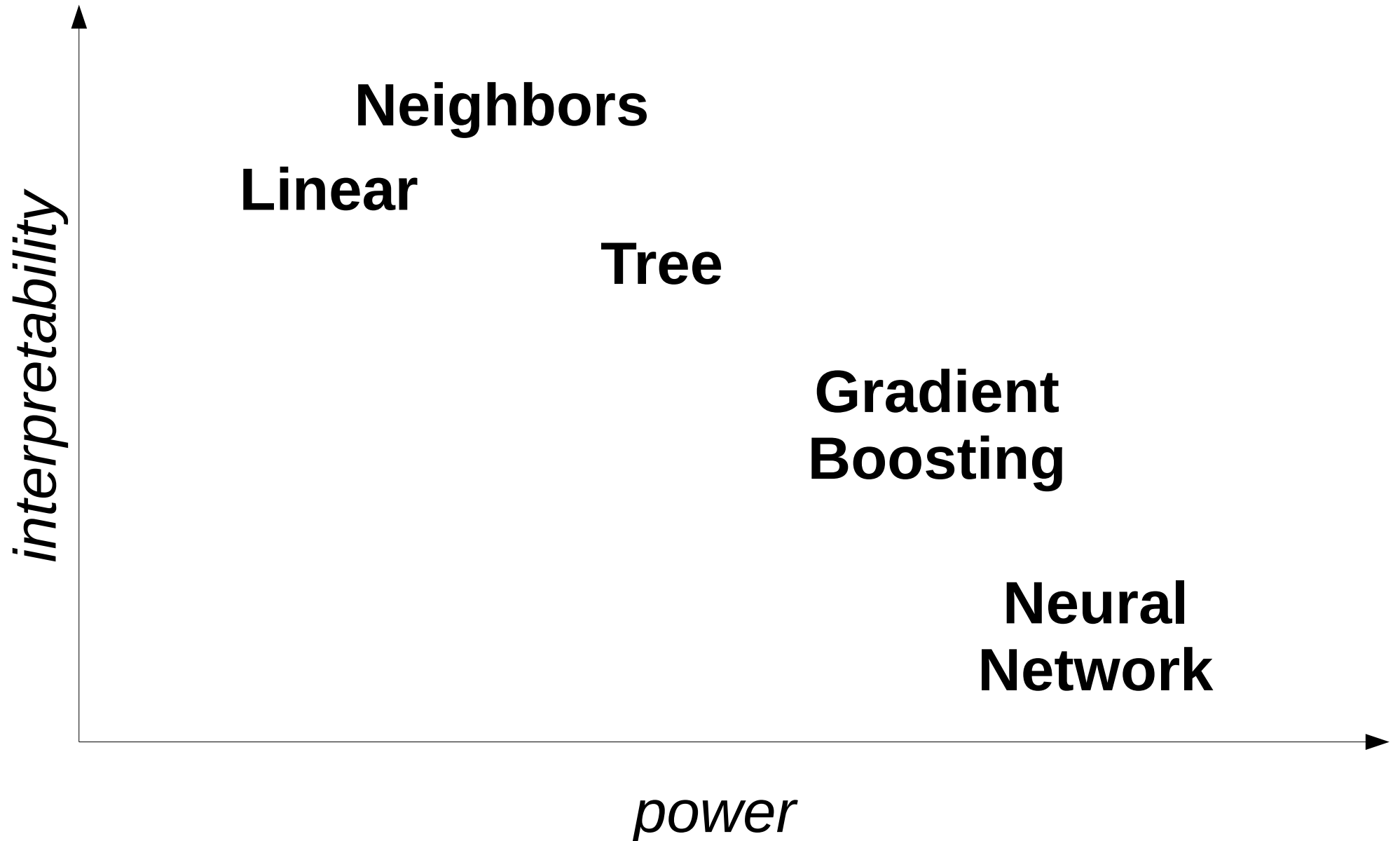
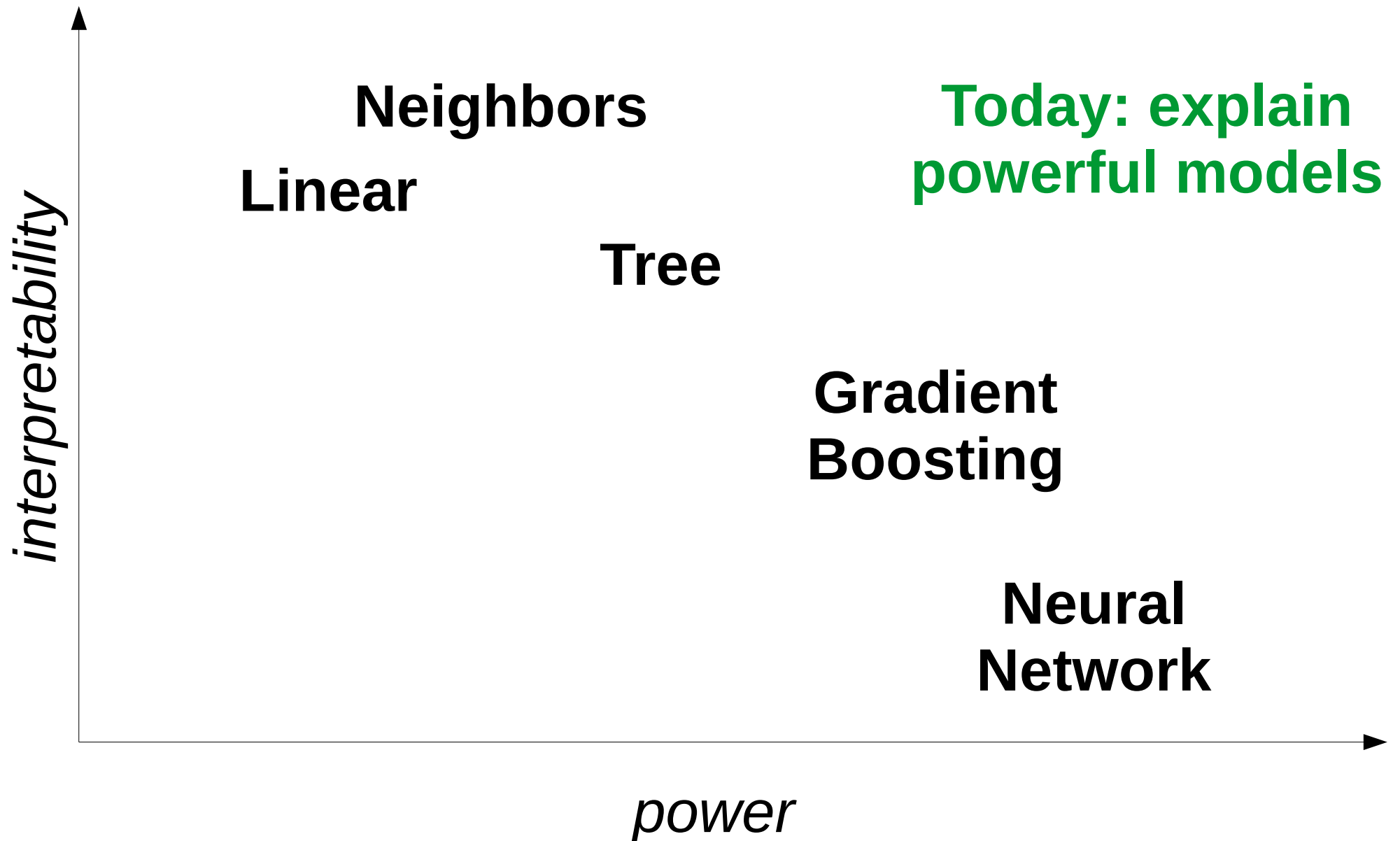


Image source: <https://playground.tensorflow.org>

Power vs interpretability



Power vs interpretability

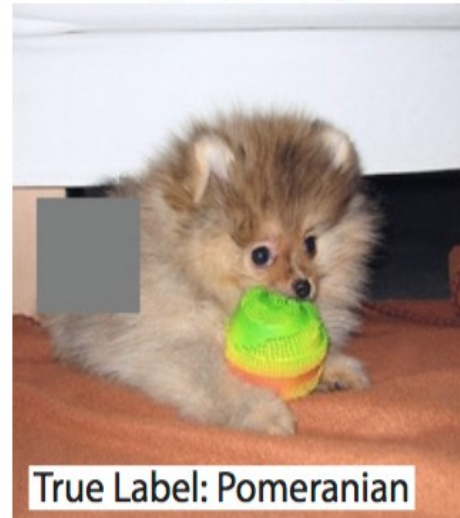


Explanation by occlusion

Idea:

- Let's add noise to inputs and see what happens!
- For images: slide a gray square over the image, measure how it affects predictions

(a) Input Image



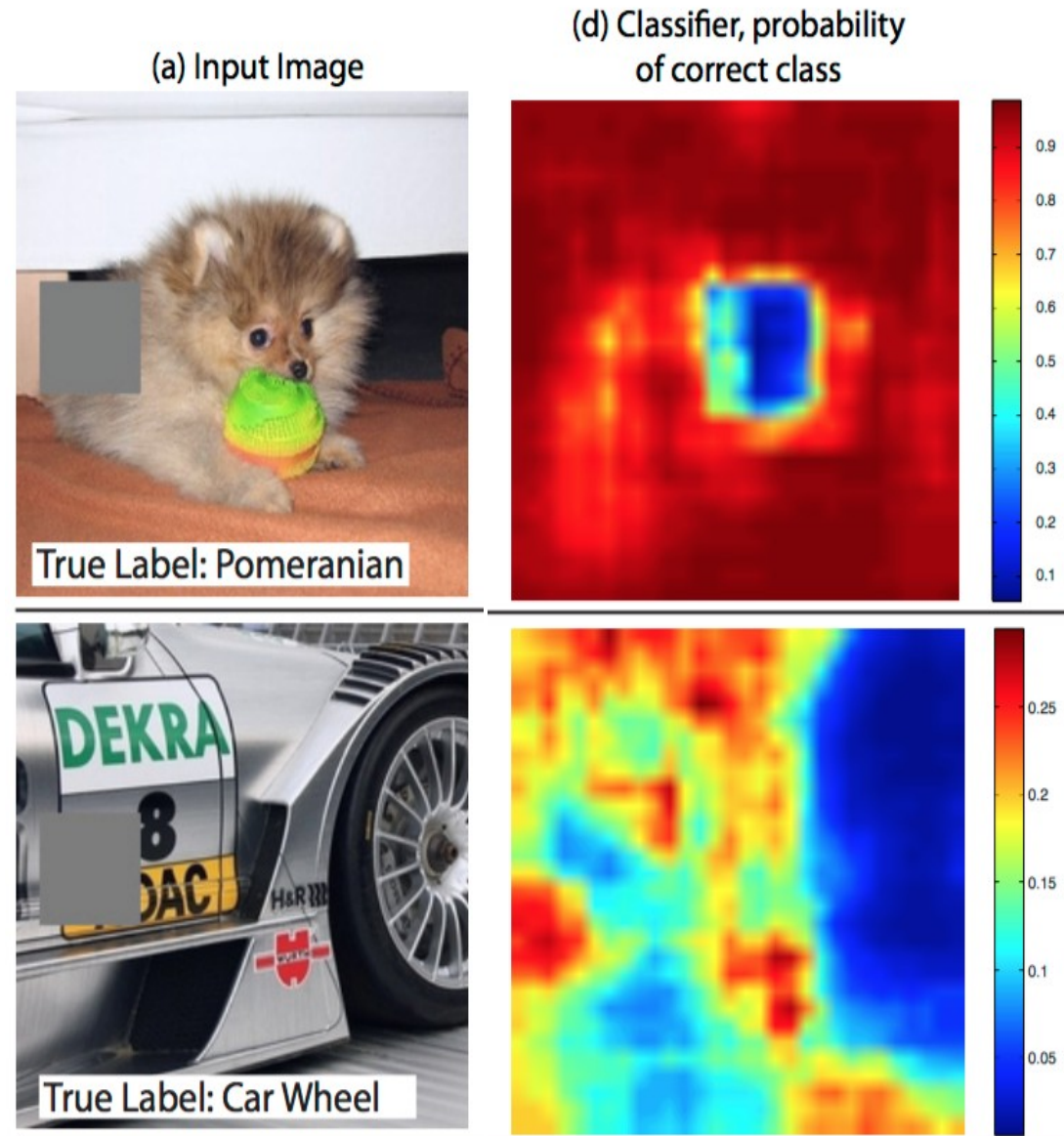
(d) Classifier, probability of correct class

Your guess?

Explanation by occlusion

Idea:

- Let's add noise to inputs and see what happens!
- For images: slide a gray square over the image, measure how it affects predictions



Explanation by occlusion

Idea:

- Let's add noise to inputs and see what happens!
- For texts: drop individual words and measure how it affects predictions

senior developer aspnet , c , sql

my client are looking for a senior . net developer to join their team designing and developing business solutions with a focus on buildir

sales specialist iv access and infusion

sales representative medical sales iv access and infusion an access and infusion solutions . formally recognised as the n

cleaning operative

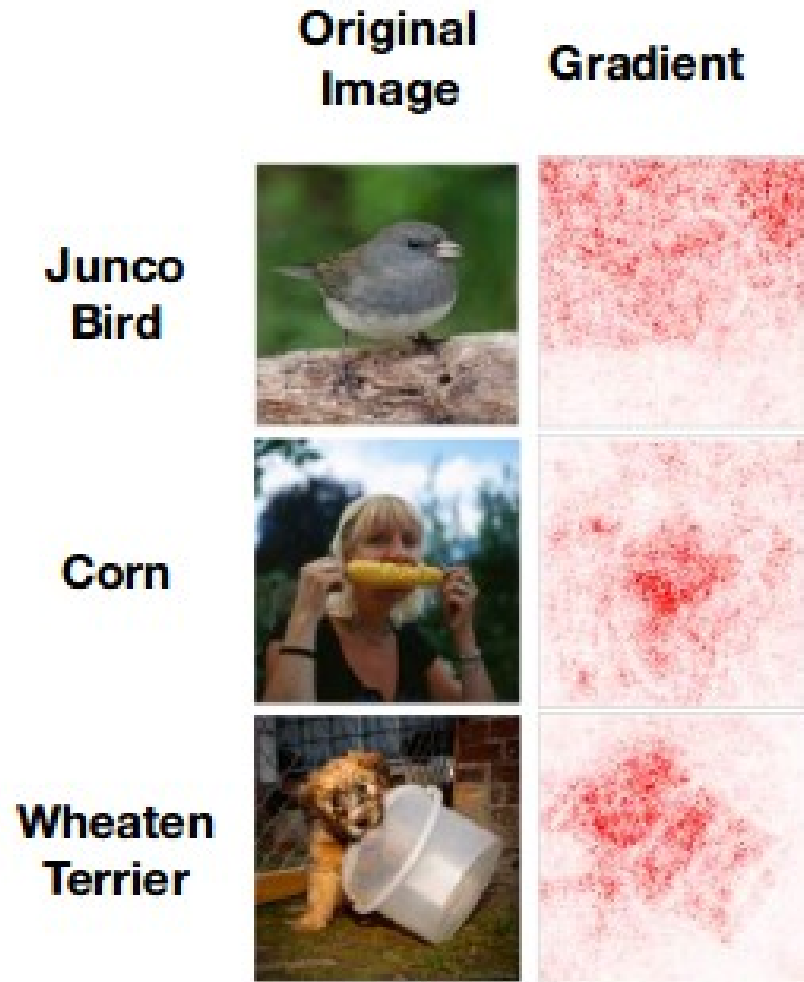
12 . 5 hours per week monday friday 9am 11 . 30am duties to include staff toilets and rest room . must be able to read as they will be using

Image: salary prediction

Explanation by gradients

Idea: use gradients!

$$\nabla_{x_i} model(x) = \frac{\partial model(x)}{\partial x_i}$$



Explanation by gradients

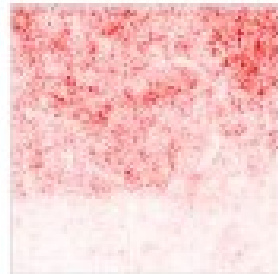
Idea: use gradients!

$$\nabla_{x_i} model(x) = \frac{\partial model(x)}{\partial x_i}$$

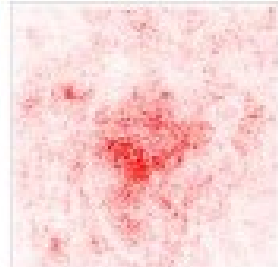
Original
Image

Gradient

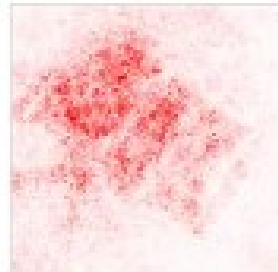
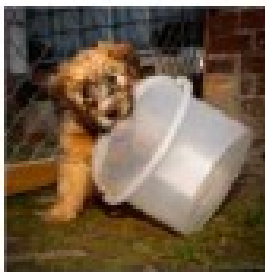
Junco
Bird



Corn



Wheaten
Terrier



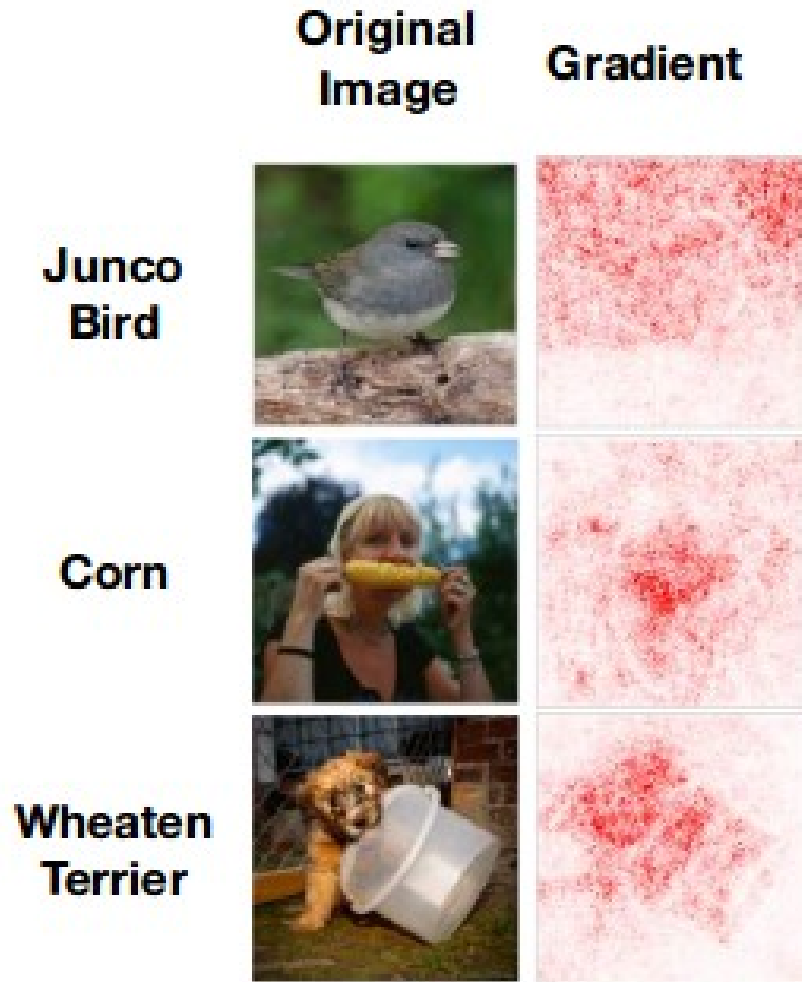
Gradients are too sensitive
to small changes in \mathbf{x}

Q: How would you fix that?

Explanation by gradients

Idea: use gradients!

$$\nabla_{x_i} model(x) = \frac{\partial model(x)}{\partial x_i}$$




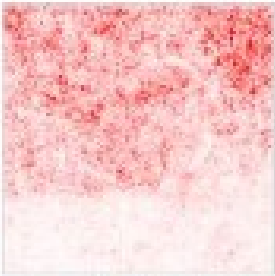
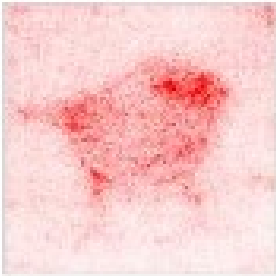

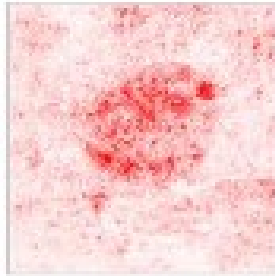

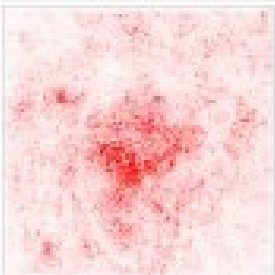
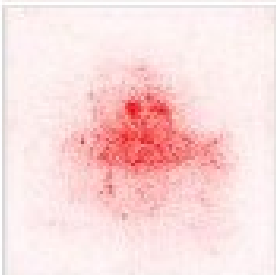

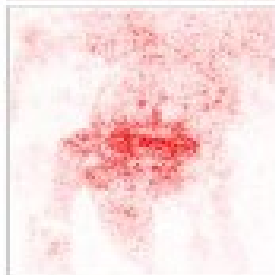

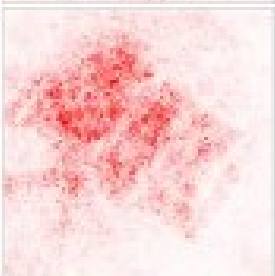
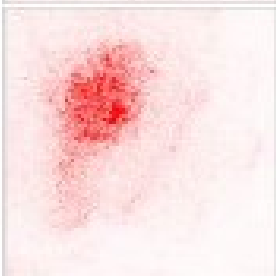

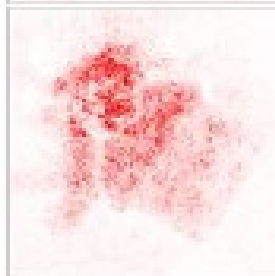
Gradients are too sensitive to small changes in \mathbf{x}

Smoothgrad: average gradients over several **noisy** copies of \mathbf{x}

(one of many heuristics)

Explanation by gradients

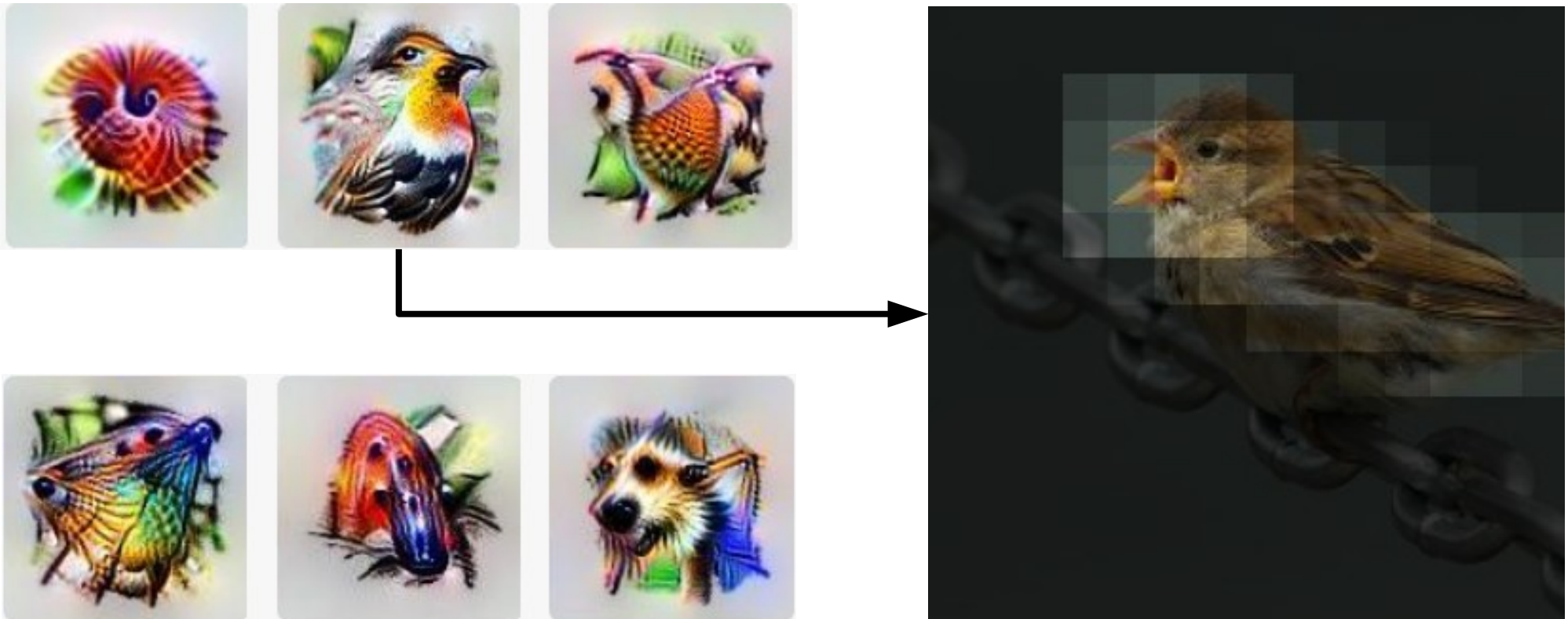
Idea: use gradients! $\nabla_{x_i} model(x) = \frac{\partial model(x)}{\partial x_i}$

	Original Image	Gradient	SmoothGrad	Guided BackProp	Integrated Gradients
Junco Bird					
Corn					
Wheaten Terrier					

Explanation by optimization

Idea: build an image that maximizes the activation of a particular neuron

Must read: distill.pub/2018/building-blocks



Explanation by optimization

Idea: build an image that maximizes the activation of a particular neuron

Must read: distill.pub/2018/building-blocks

More:

<https://distill.pub>

<https://poloclub.github.io>

<https://karpathy.github.io>

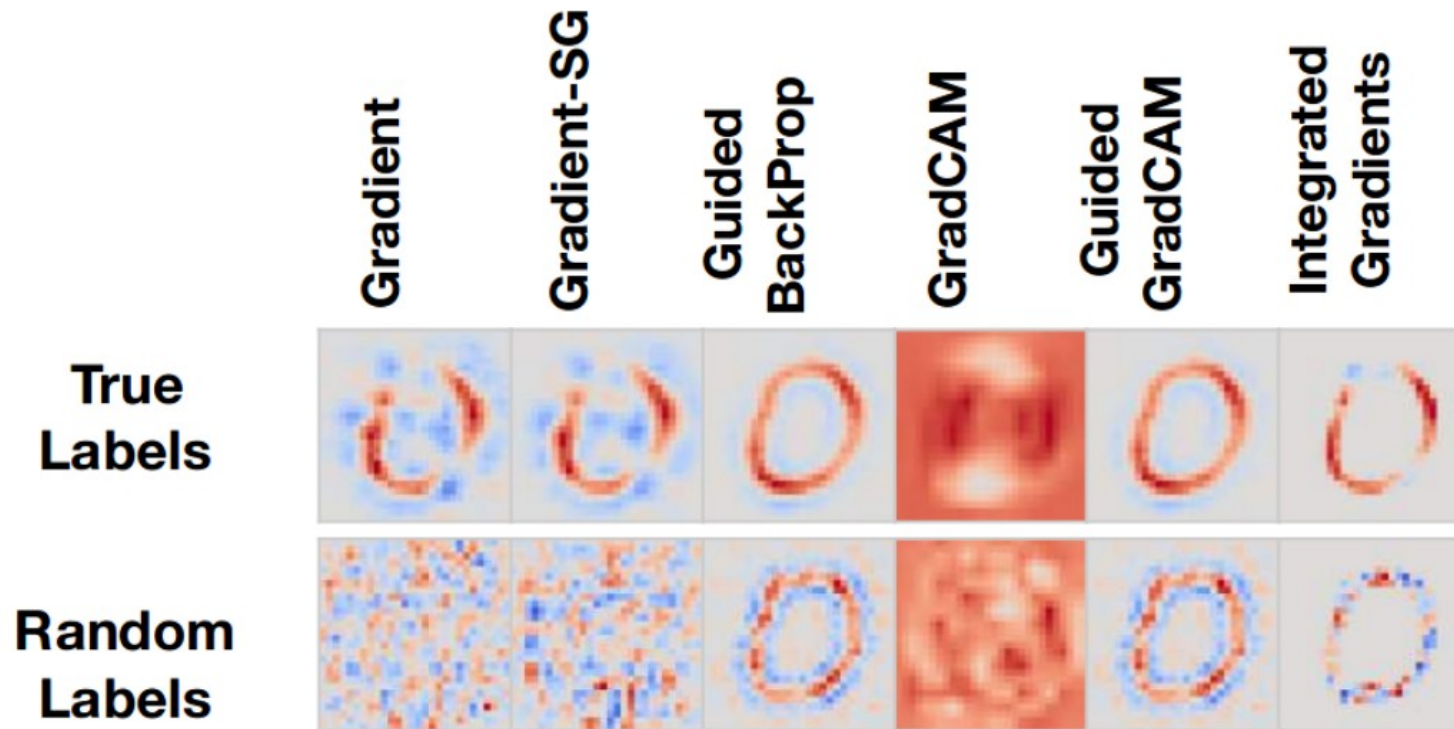
Don't trust yourself!

The method outputs a noisy image
you see something reasonable
should you be satisfied?

How can you **verify** the explanation?

Don't trust yourself!

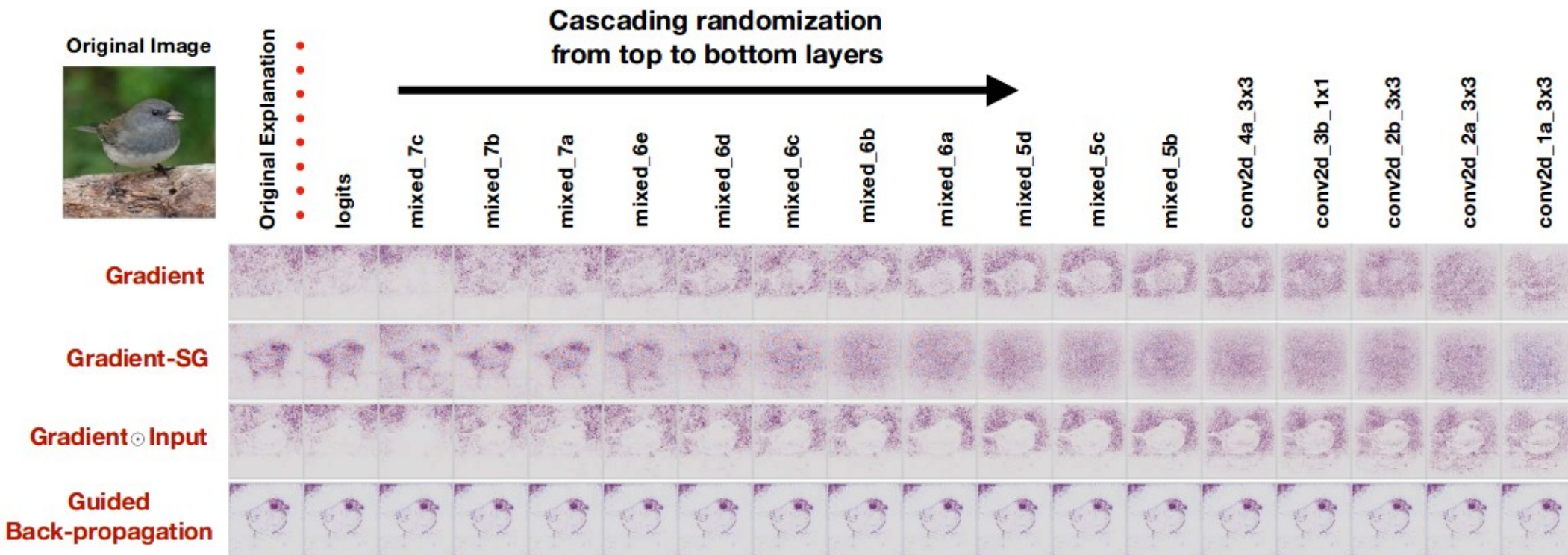
Idea: train a bogus model to see if the method can “explain” the fake model



Source: *Sanity Checks for Saliency Maps*

Don't trust yourself!

Idea: replace weights with random one layer at a time (top to bottom)

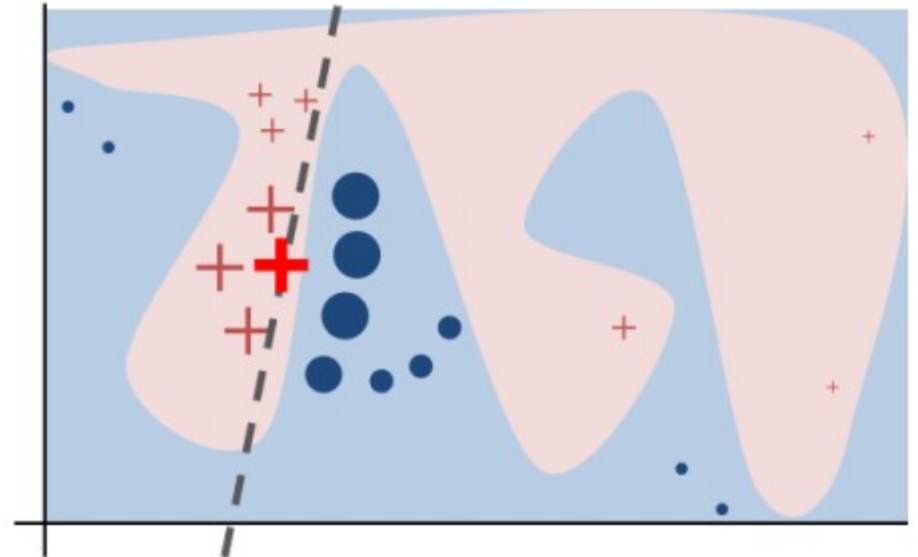


Source: *Sanity Checks for Saliency Maps*

Explanation by approximation

Idea:

- Approximate your model with something explainable
e.g. linear model
- The approximation only needs to hold **locally**
i.e. on similar inputs

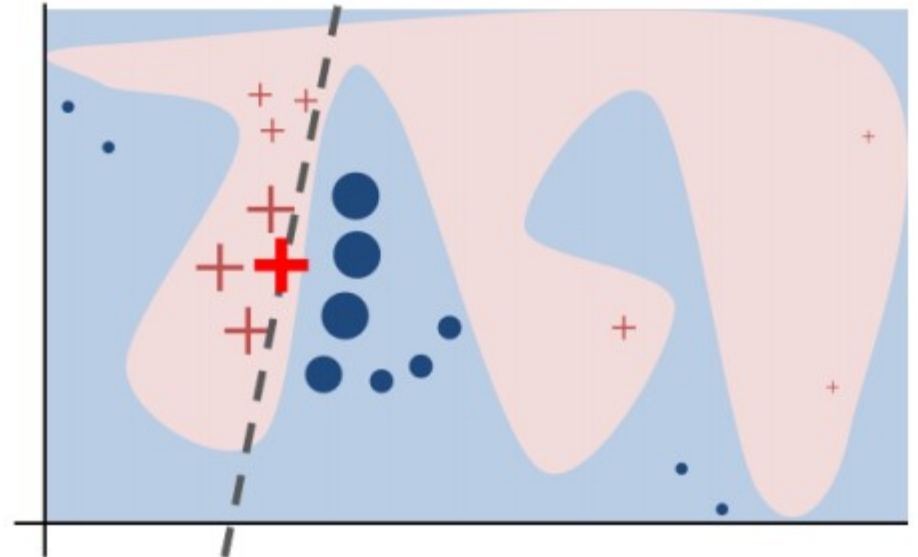


Read more in [the paper](#)

Explanation by approximation

Idea:

- Approximate your model with something explainable
e.g. linear model
- The approximation only needs to hold **locally**
i.e. on similar inputs



Read more in [the paper](#)



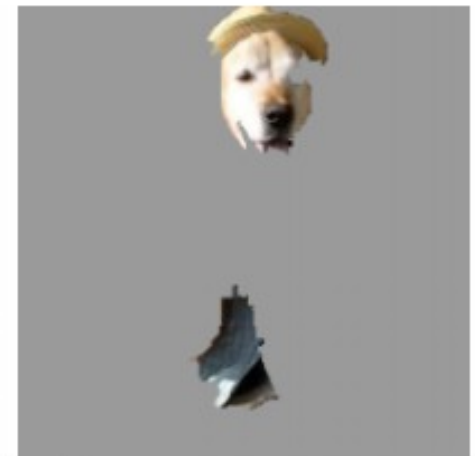
(a) Original Image



(b) Explaining *Electric guitar*



(c) Explaining *Acoustic guitar*

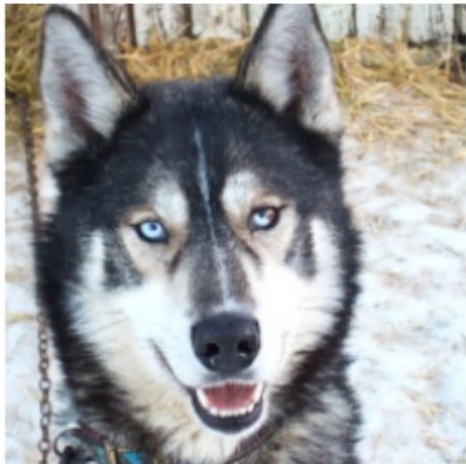
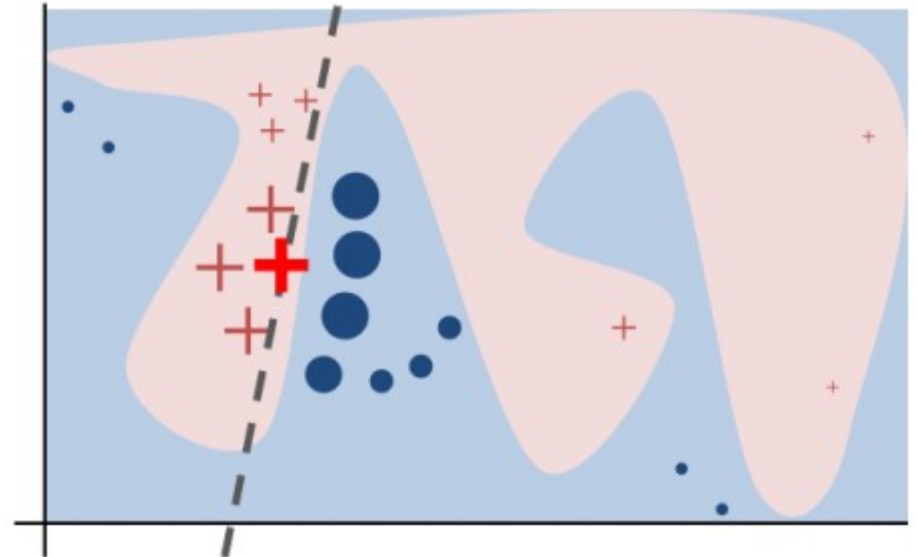


(d) Explaining *Labrador*

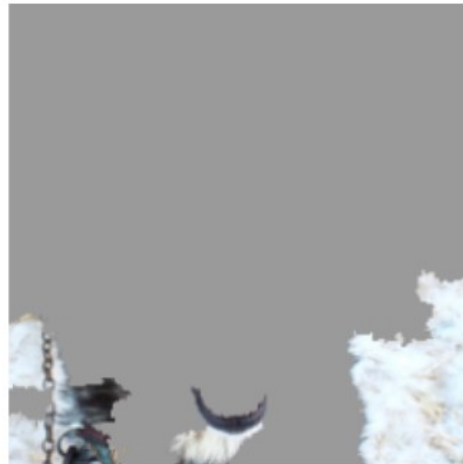
Explanation by approximation

Idea:

- Approximate your model with something explainable
e.g. linear model
- The approximation only needs to hold **locally**
i.e. on similar inputs



(a) Husky classified as wolf



(b) Explanation

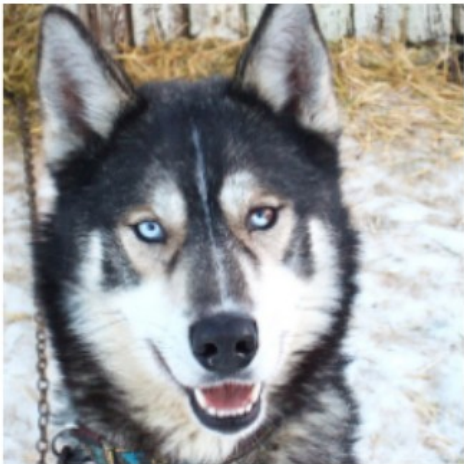
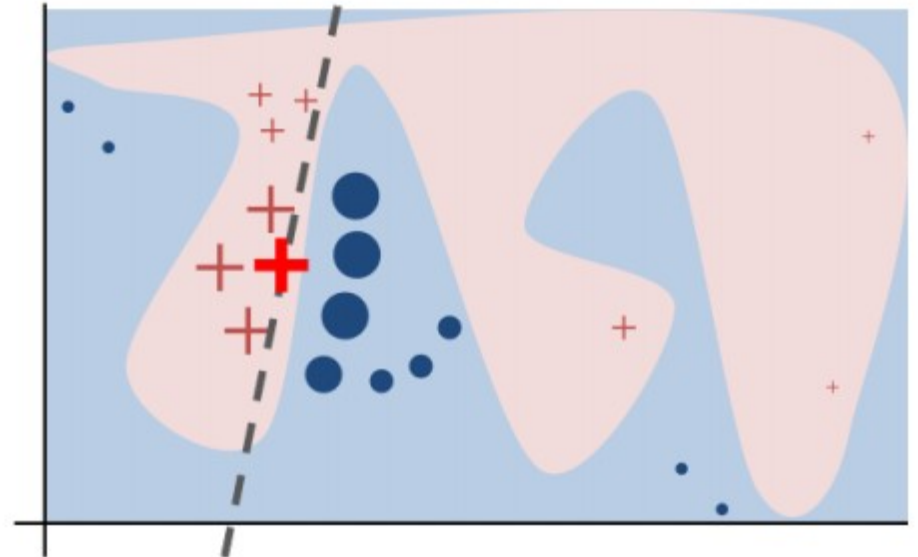
Left image: model mislabeled a husky dog as a wolf; explanation: snow :)

Figures taken from the paper
[Why Should I Trust You?](#)

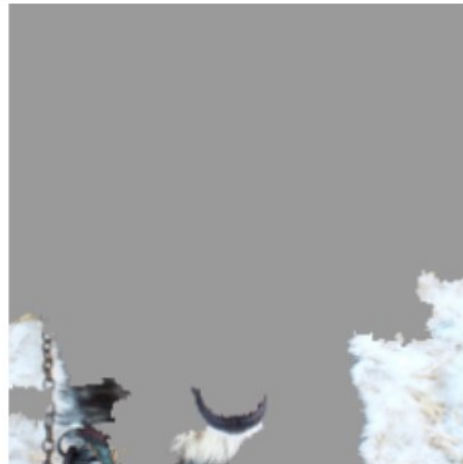
Explanation by approximation

Idea:

- Approximate your model with something explainable
e.g. linear model
- The approximation only needs to hold **locally**
i.e. on similar inputs



(a) Husky classified as wolf



(b) Explanation

Read more:

arxiv.org/abs/1602.04938

arxiv.org/abs/1705.07874

arxiv.org/abs/1904.12991

Explanation by game theory

Idea: features are a “players” that play a cooperative game of making a prediction

Explanation by game theory

Idea: features are a “players” that play a cooperative game of making a prediction

Equivalent “game”:

Alice, Bob and Carol ordered a \$1000 meal at a restaurant

Q: How should they split the bill?

Hint: here’s what it would cost for them individually & in pairs

Who goes	Alice	Bob	Carol	A & B	A & C	B & C	A, B & C
Total price	400	560	720	740	780	980	1000

Ideas?

Explanation by game theory

Idea: features are a “players” that play a cooperative game of making a prediction

Equivalent “game”:

Alice, Bob and Carol ordered a \$1000 meal at a restaurant

Q: How should they split the bill?

Hint: here’s what it would cost for them individually & in pairs

Who goes	Alice	Bob	Carol	A & B	A & C	B & C	A, B & C
Total price	400	560	720	740	780	980	1000

Game theorist’s answer: Shapley values!

$$\phi_i(v) = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!} (v(S \cup \{i\}) - v(S))$$

Explanation by game theory

Idea: features are a “players” that play a cooperative game of making a prediction

Equivalent “game”:

Alice, Bob and Carol ordered a \$1000 meal at a restaurant

Q: How should they split the bill?

Hint: here’s what it would cost for them individually & in pairs

Who goes	Alice	Bob	Carol	A & B	A & C	B & C	A, B & C
Total price	400	560	720	740	780	980	1000

Game theory: what is the best way to split the bill?

$\phi_i(v)$ =

WARNING

Do **NOT** take your game theorist friend to a restaurant! You’ll thank me later.

$v(S)$

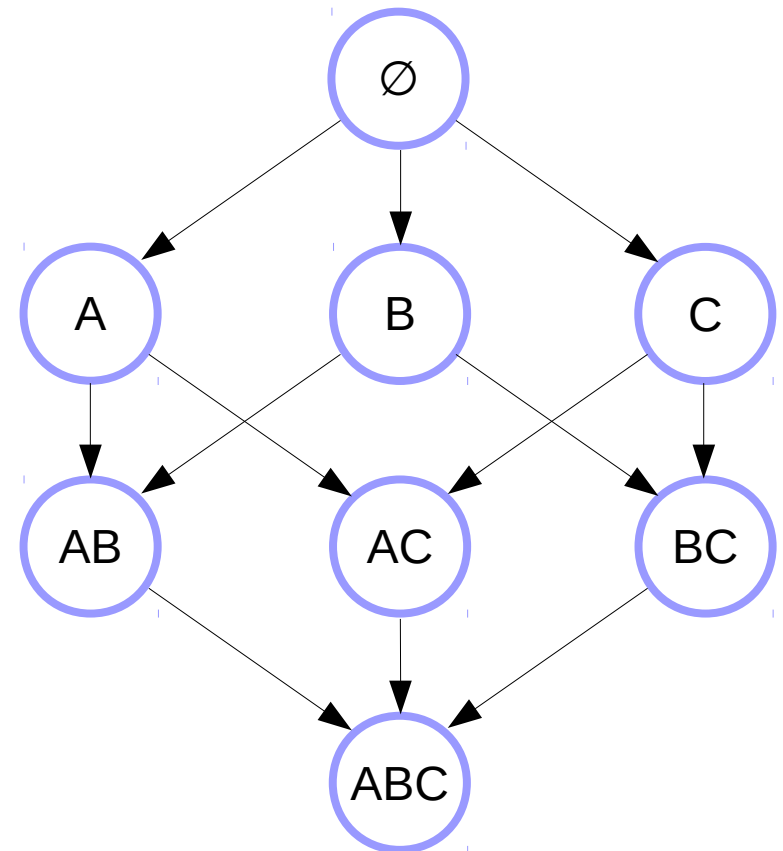
Shapley values explained

Same old table

Who goes	Alice	Bob	Carol	A & B	A & C	B & C	A, B & C
Total price	400	560	720	740	780	980	1000

$\text{Shapley}(X)$ = average increase
in cost from adding X to a group

Note: average over all *paths*.



Shapley values explained

Same old table

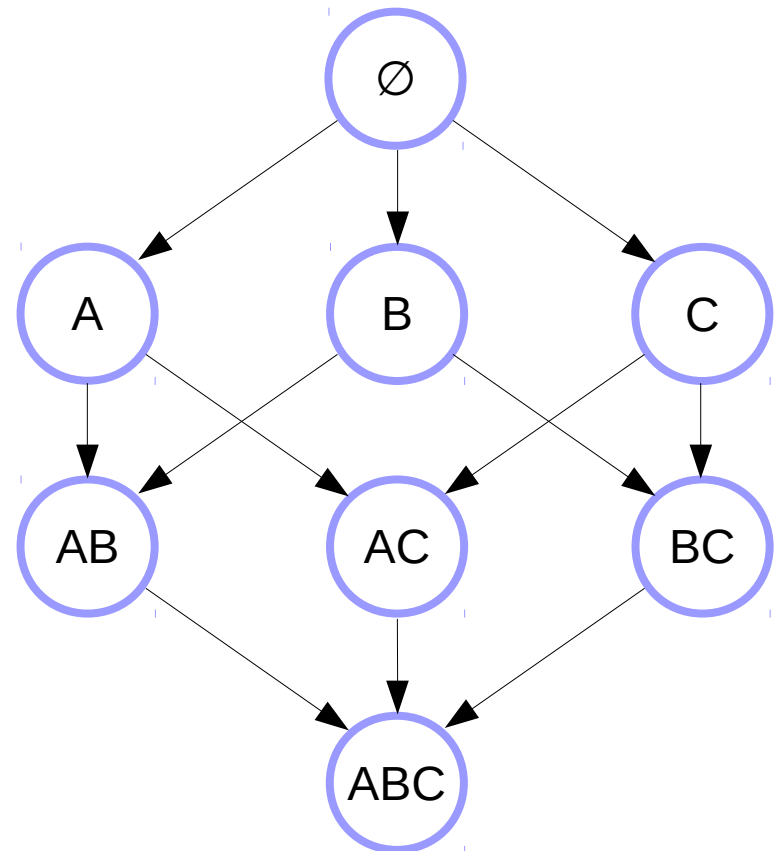
Who goes	Alice	Bob	Carol	A & B	A & C	B & C	A, B & C
Total price	400	560	720	740	780	980	1000

$\text{Shapley}(X)$ = average increase
in cost from adding X to a group

Note: average over all *paths*.

$$\text{Shapley}(A) = \frac{2}{6} \cdot 400 + \dots$$

Q: What else?



Shapley values explained

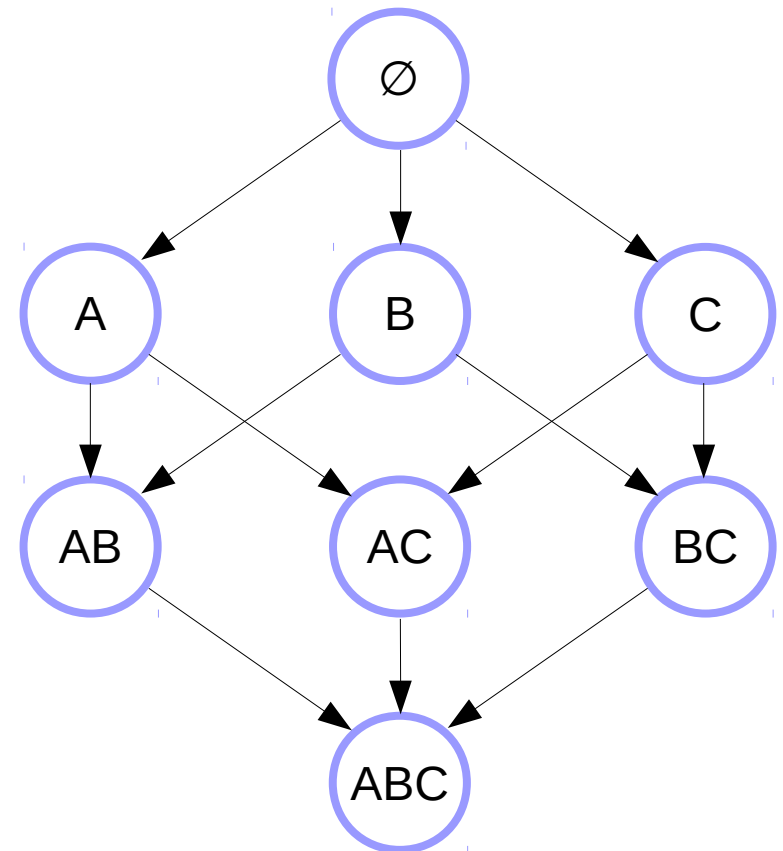
Same old table

Who goes	Alice	Bob	Carol	A & B	A & C	B & C	A, B & C
Total price	400	560	720	740	780	980	1000

$\text{Shapley}(X)$ = average increase
in cost from adding X to a group

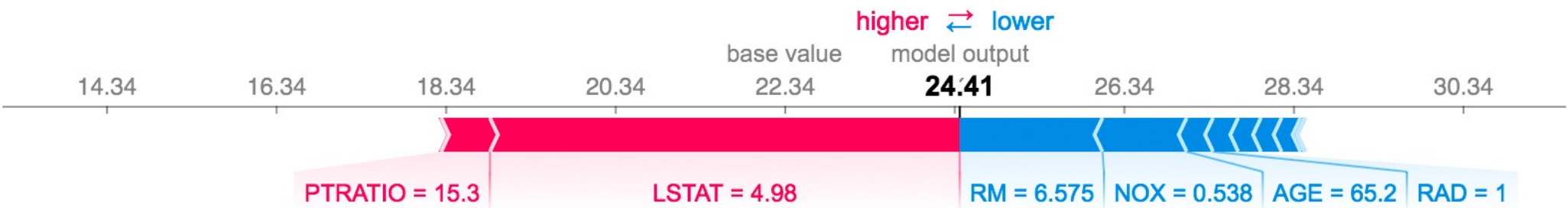
Note: average over all *paths*.

$$\begin{aligned}\text{Shapley}(A) &= \frac{2}{6} \cdot 400 + \dots \\ &+ \frac{1}{6} \cdot (740 - 560) + \frac{1}{6} \cdot (780 - 720) + \\ &+ \frac{2}{6} \cdot (1000 - 990) = 180\end{aligned}$$



Explanation by game theory

SHAP = *Shapley values for features + clever approximation*
State of the art in after-the-fact model explanation



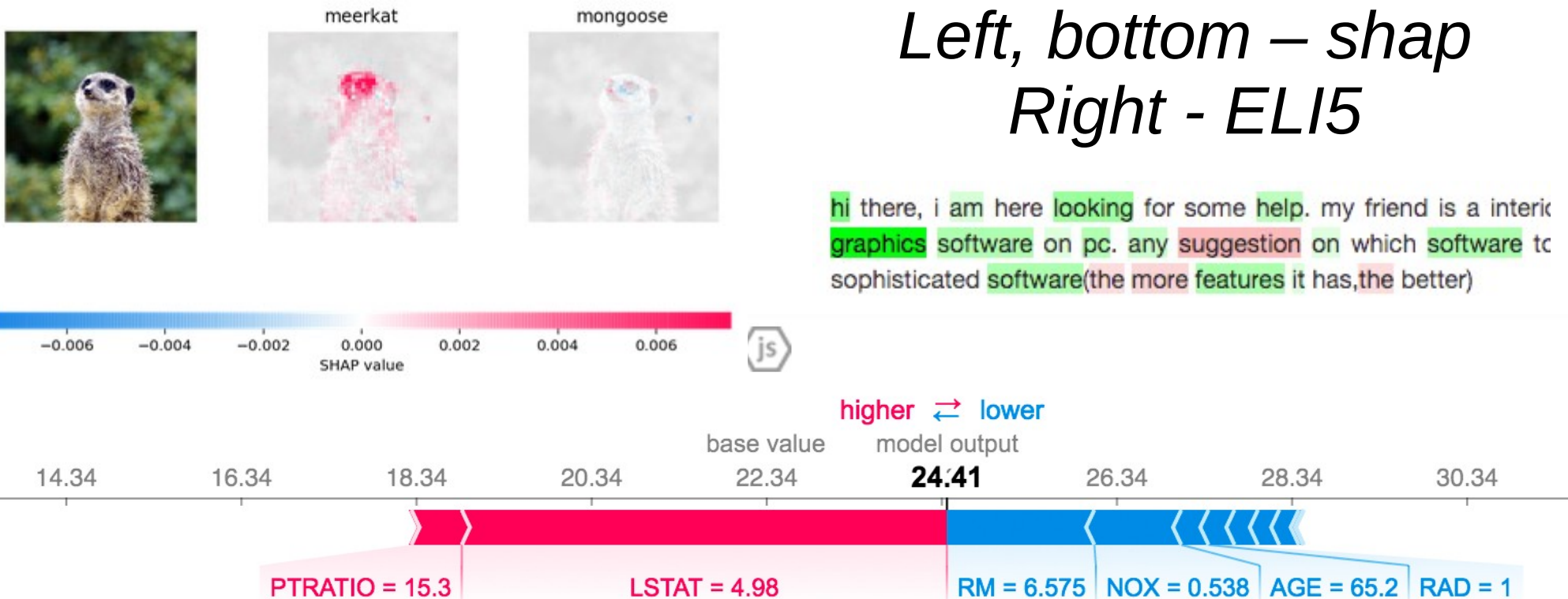
MOAR:

- SHAP original paper: tinyurl.com/shap-paper (NeurIPS'17)
- SHAP explained by paper author: youtu.be/ngOBhhINWb8
- Shapley values in game theory: youtu.be/w9O0fkfMkx0

Frameworks

SHAP - <https://github.com/slundberg/shap>
(tensorflow, keras, pytorch, sklearn-like)

ELI5 - <https://github.com/TeamHG-Memex/eli5>
(popular explainers for keras/tf, sklearn-like)

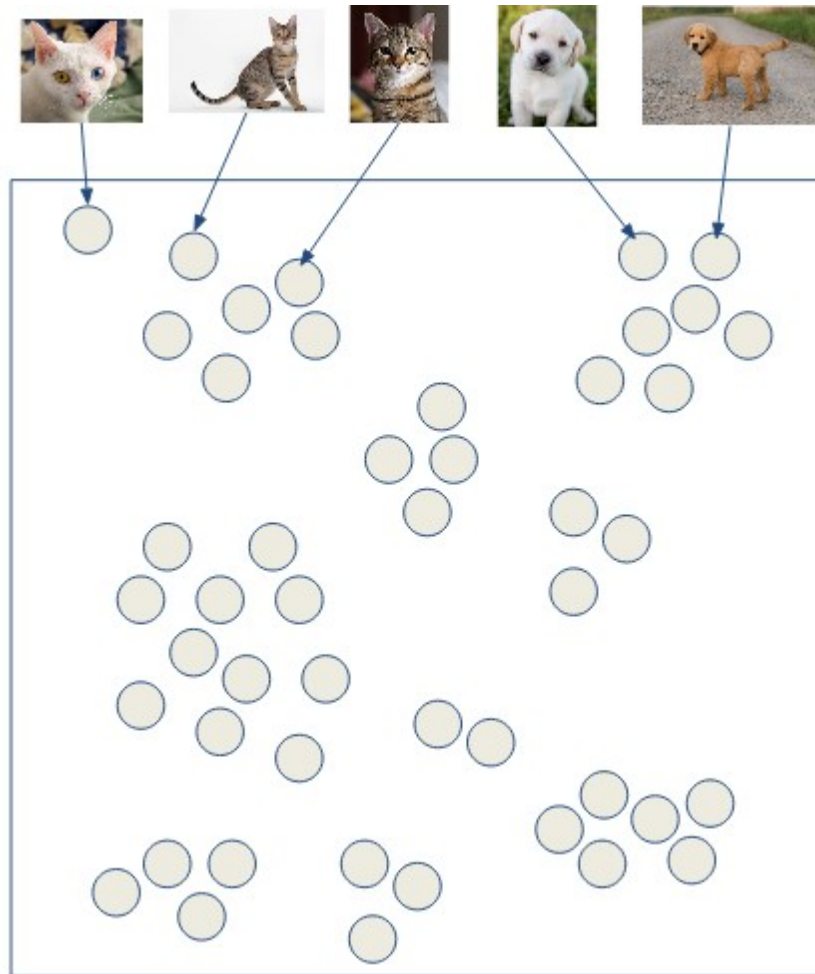


So far: explaining black-box models

Now: model-specific methods

Explanation by design

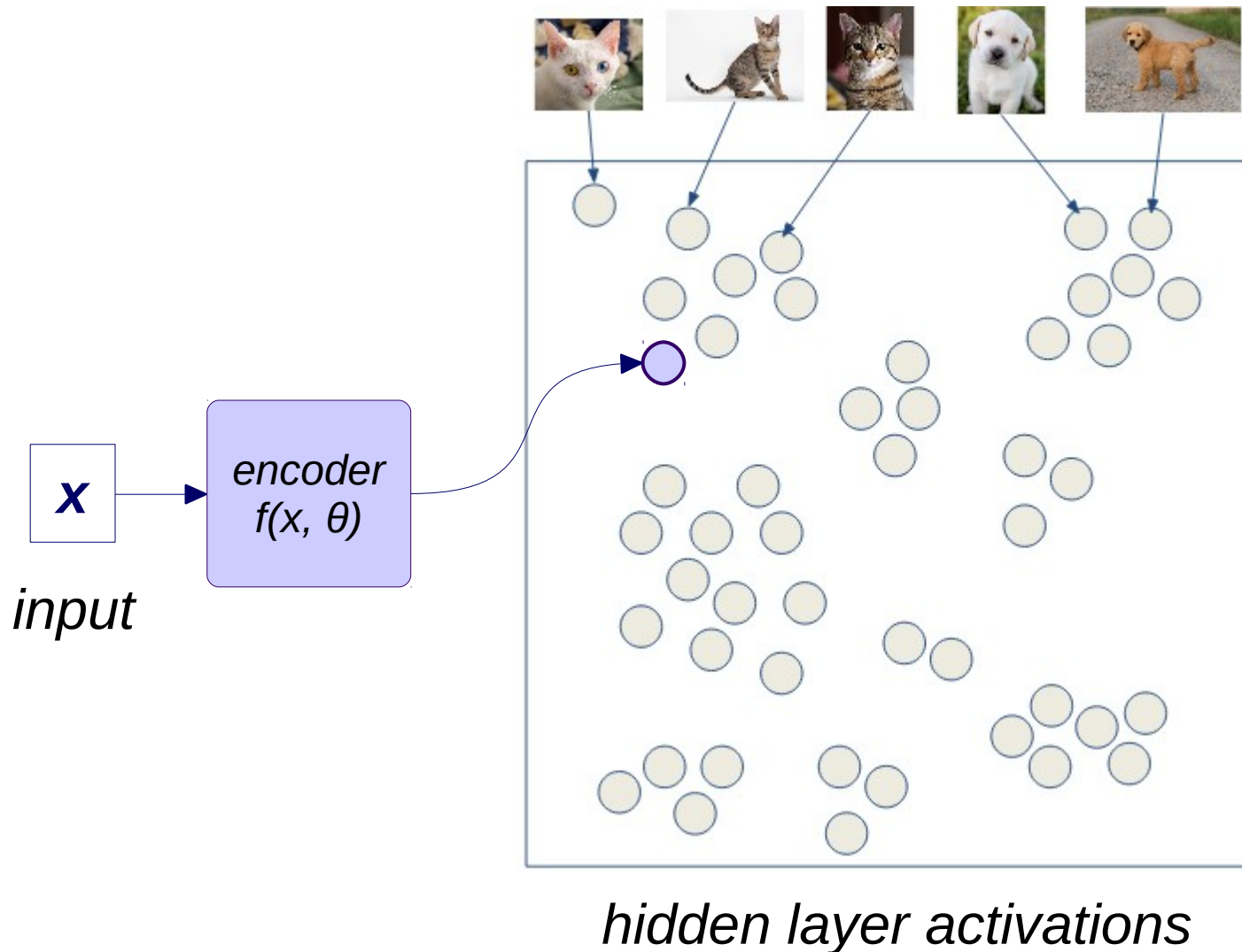
Idea: design architecture to be interpretable



hidden layer activations

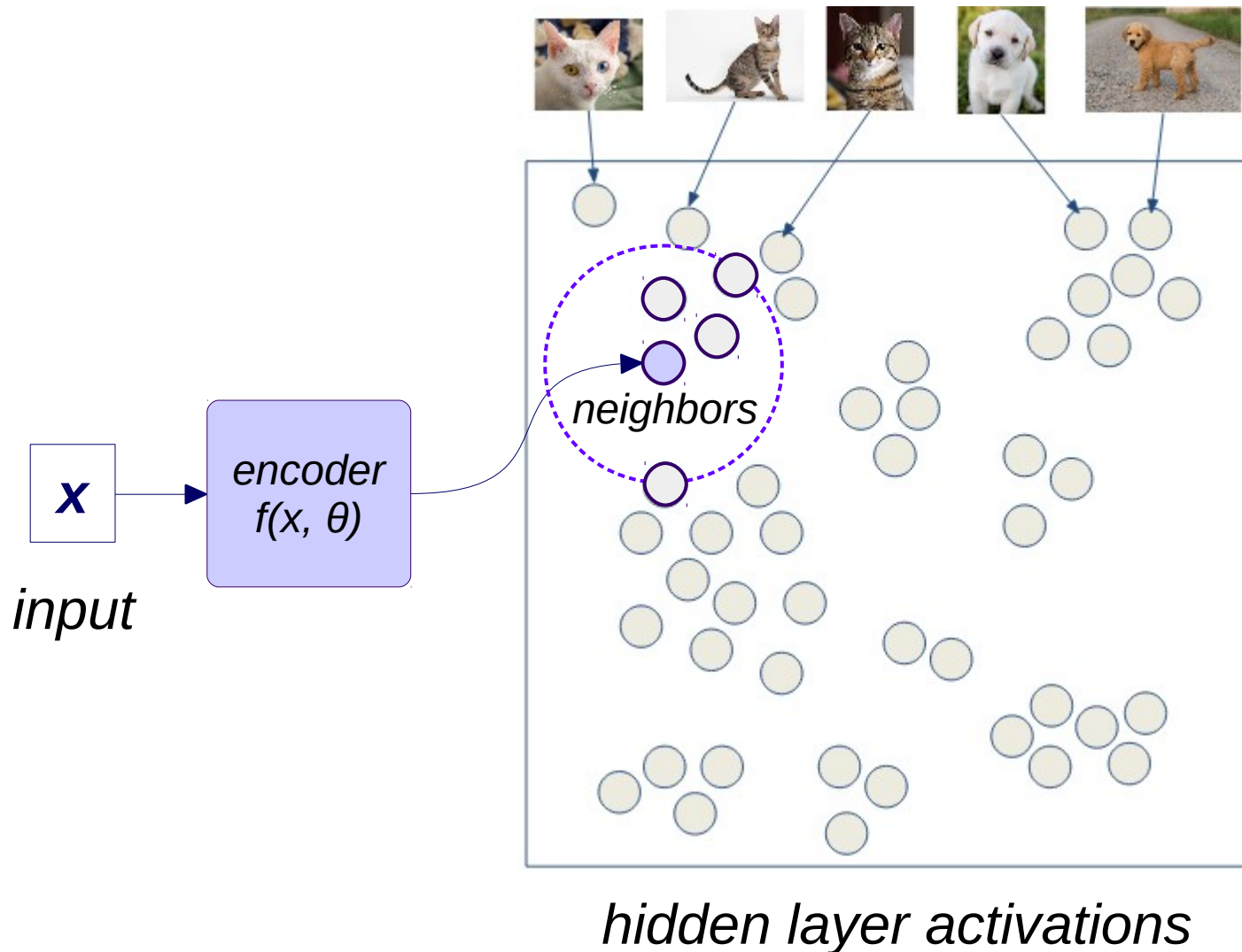
Explanation by design

Idea: design architecture to be interpretable



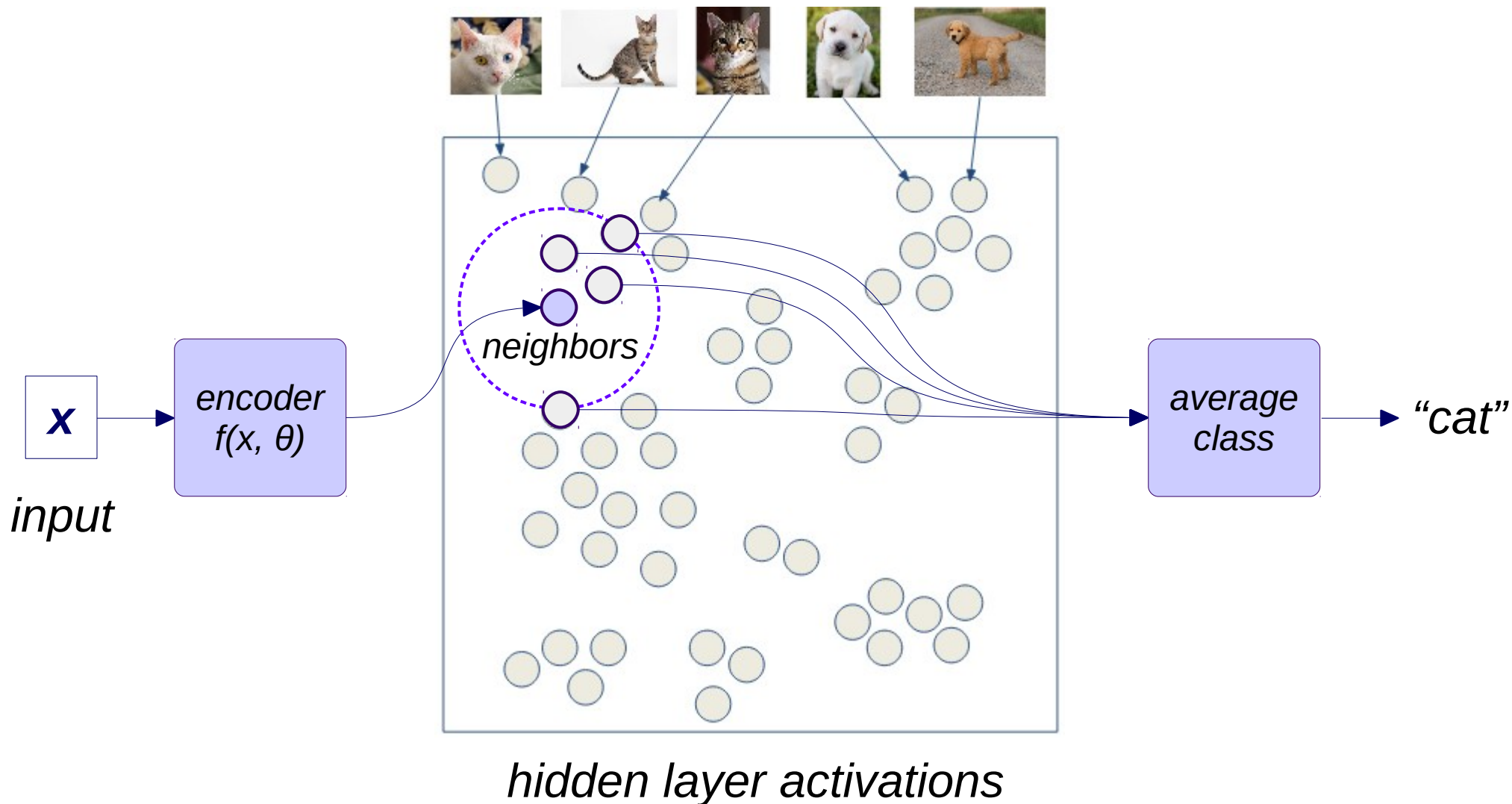
Explanation by design

Idea: design architecture to be interpretable



Explanation by design

Idea: design architecture to be interpretable



Explanation by design

Idea: design architecture to be interpretable

Prototype objects and answers: $(\hat{x}_0, \hat{y}_0), \dots, (\hat{x}_N, \hat{y}_N)$

“Attention” weights:
$$a(x, \hat{x}_i) = \frac{e^{\langle f(x, \theta), f(\hat{x}_i, \theta) \rangle}}{\sum_{j=0}^N e^{\langle f(x, \theta), f(\hat{x}_j, \theta) \rangle}}$$

Prediction by averaging:
$$y^{pred}(x) = \sum_i \hat{y}_i \cdot a_i(x, \hat{x}_i)$$

Explanation by design

Idea: design architecture to be interpretable

Prototype objects and answers: $(\hat{x}_0, \hat{y}_0), \dots, (\hat{x}_N, \hat{y}_N)$

“Attention” weights:
$$a(x, \hat{x}_i) = \frac{e^{\langle f(x, \theta), f(\hat{x}_i, \theta) \rangle}}{\sum_{j=0}^N e^{\langle f(x, \theta), f(\hat{x}_j, \theta) \rangle}}$$

$$y^{pred}(x) = \sum_i \hat{y}_i \cdot a_i(x, \hat{x}_i)$$

Read more: KNN

arxiv.org/abs/1703.05175

arxiv.org/abs/1803.04765

arxiv.org/abs/1809.02847

Read more: Linear

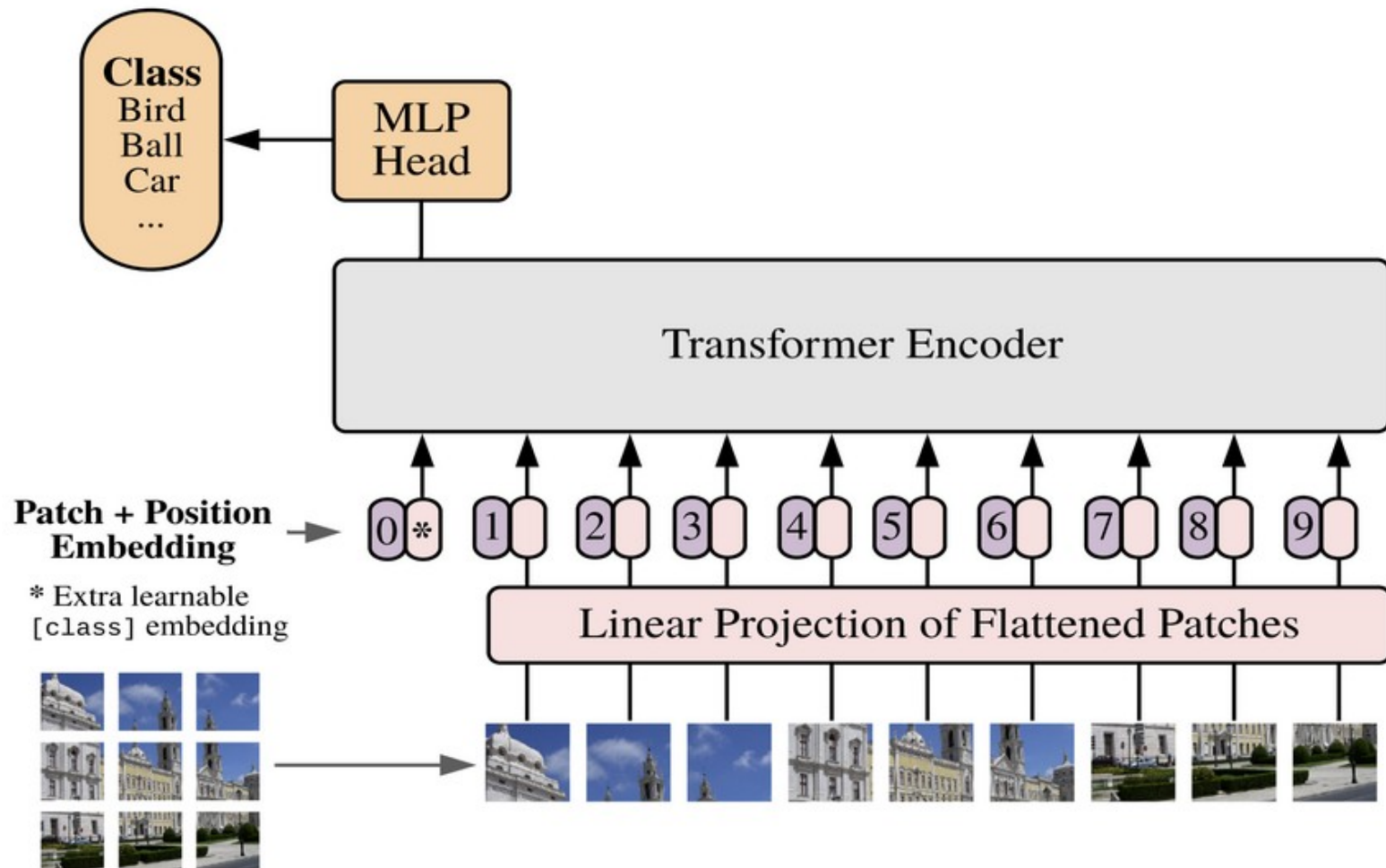
arxiv.org/abs/1705.08078

arxiv.org/abs/1806.07538

Taking it to the extreme

Paper: <https://arxiv.org/abs/2010.11929>

Vision Transformer (ViT)



Taking it to the extreme

Paper: <https://arxiv.org/abs/2104.14294>



View attention maps: <https://epfml.github.io/attention-cnn/>

The question of trust

How can I explain my model's prediction?

Why did it make this decision/mistake?

What features does it rely on?

Is my model certain about what it says?

Is there something wrong with this input?

Can I rely on this prediction?

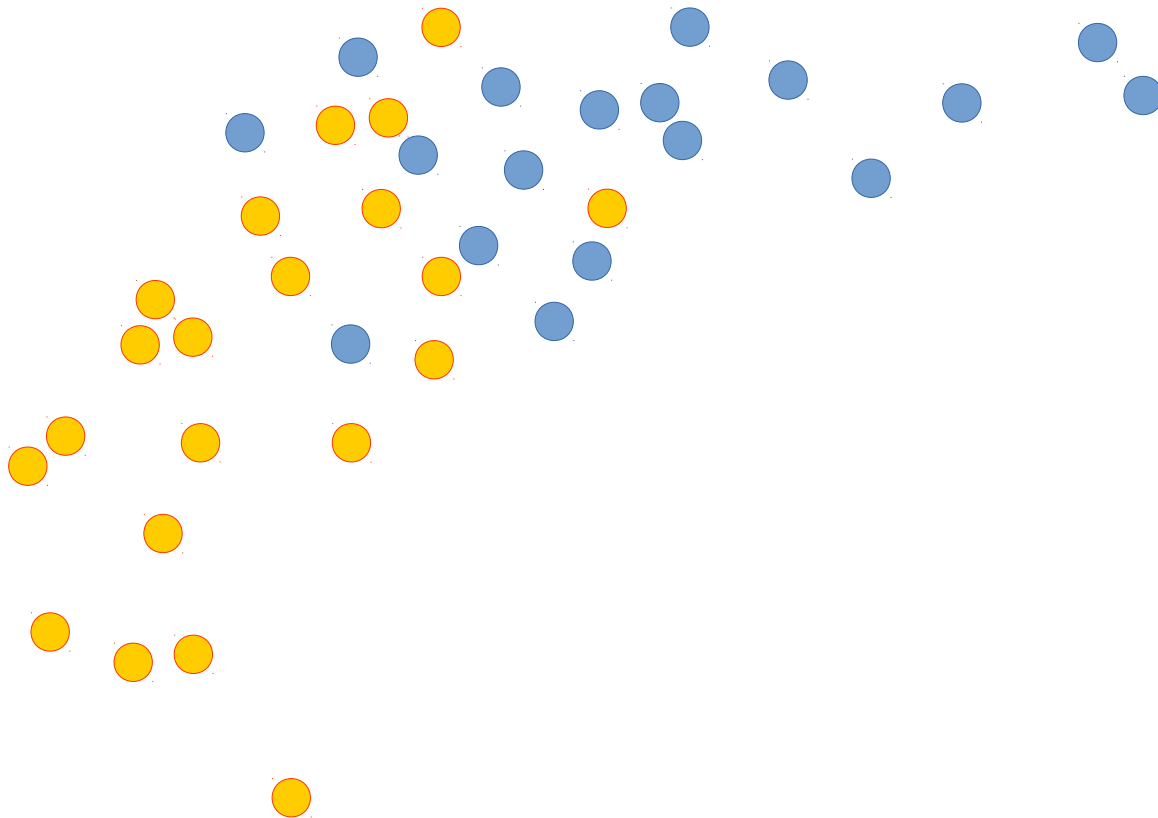
Can I trust this data?

Is something missing?

Is there any bias?

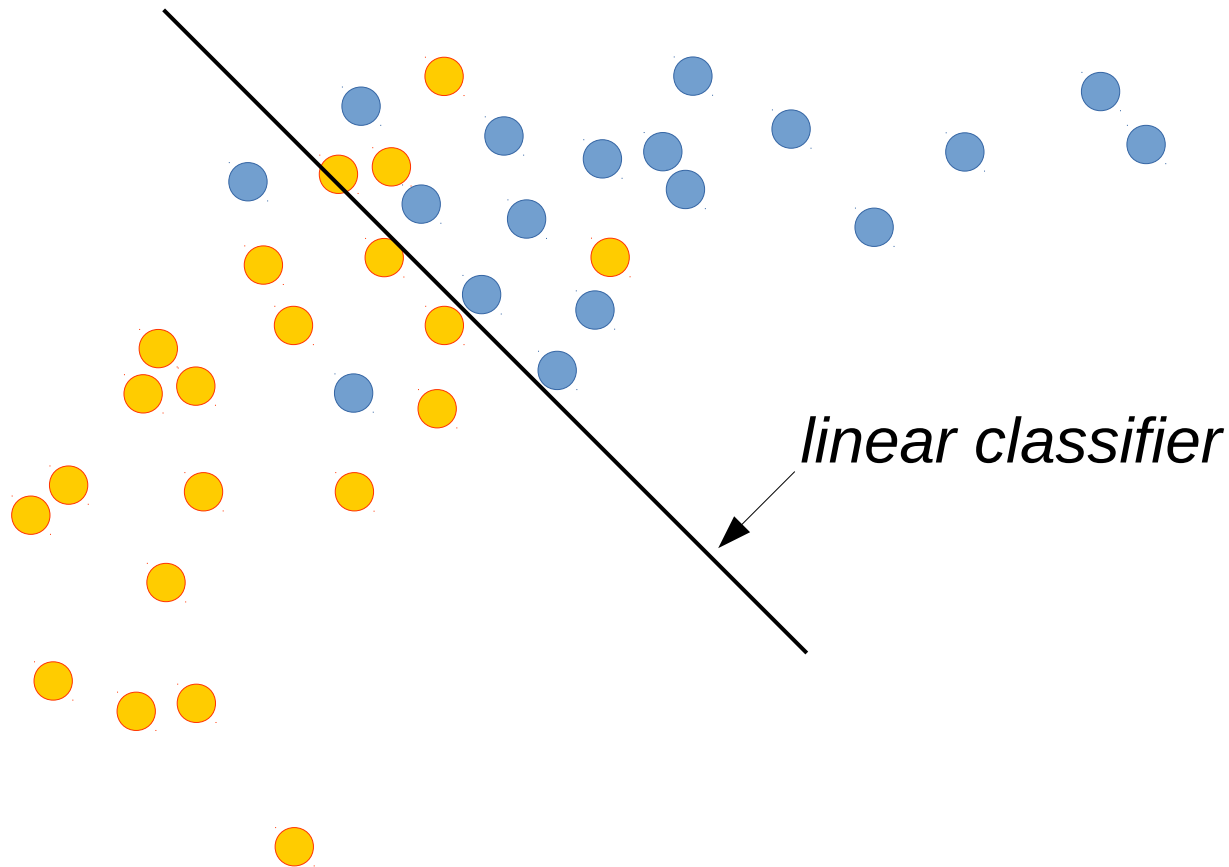
Types of uncertainty

example: binary classification



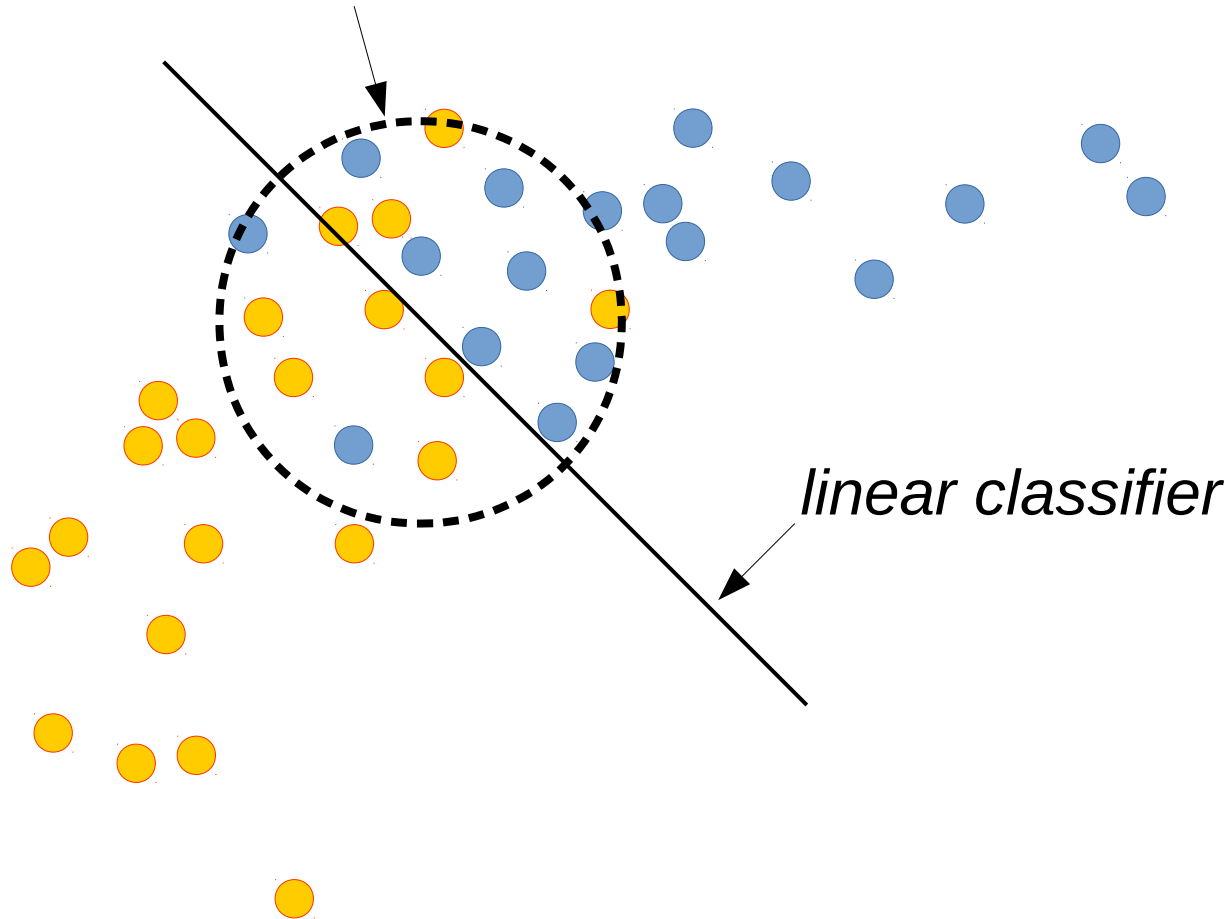
Types of uncertainty

example: binary classification



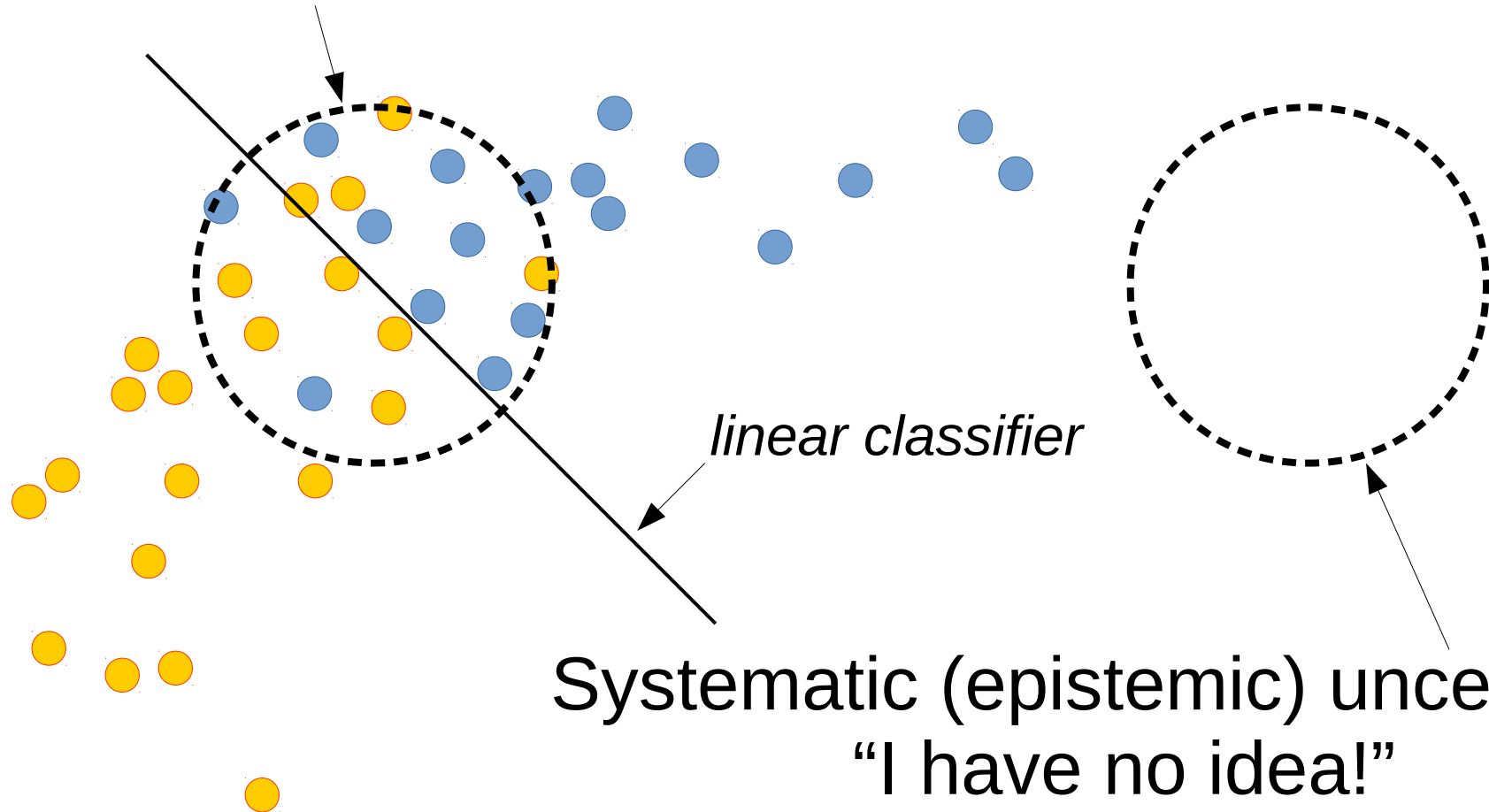
Types of uncertainty

Statistical (aleatoric) uncertainty
“I know there’s randomness”



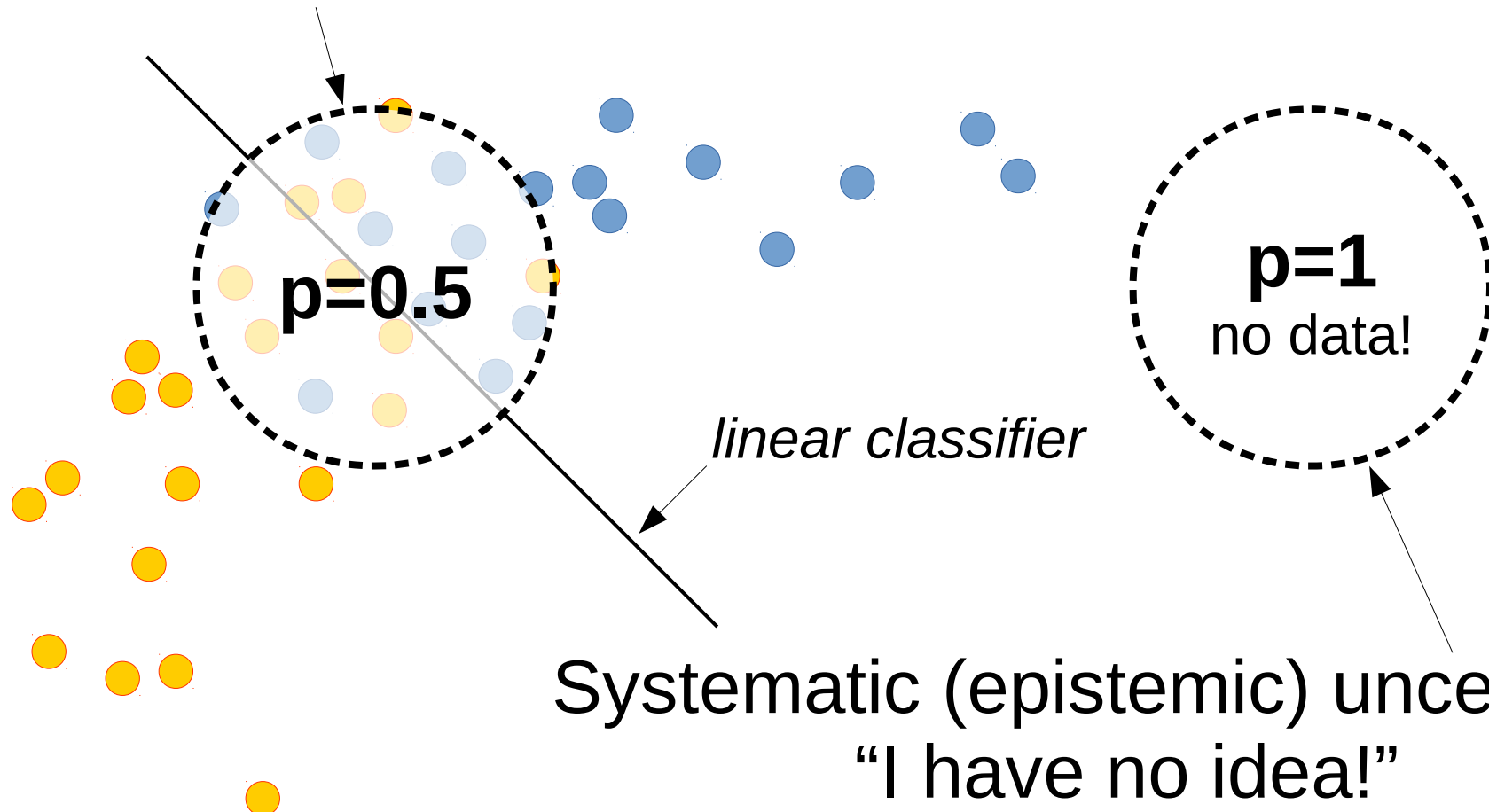
Types of uncertainty

Statistical (aleatoric) uncertainty
“I know there’s randomness”



Types of uncertainty

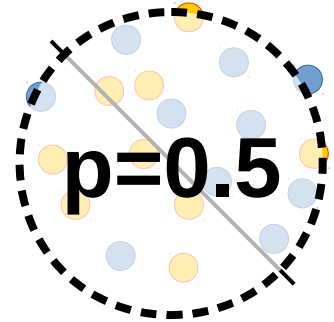
Statistical (aleatoric) uncertainty
“I know there’s randomness”



Systematic (epistemic) uncertainty
“I have no idea!”

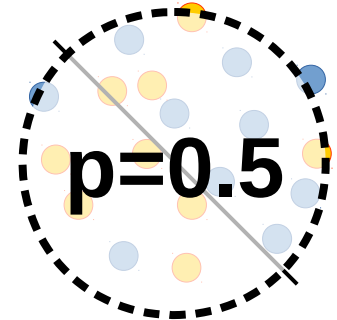
How to measure uncertainty

Aleatoric uncertainty: use predicted probability!
Exception: neural networks can be **overconfident**
Fix it by *calibrating* model predictions after the fact,
Read more: tinyurl.com/sklearn-calibration



How to measure uncertainty

Aleatoric uncertainty: use predicted probability!
Exception: neural networks can be **overconfident**
Fix it by *calibrating* model predictions after the fact,
Read more: tinyurl.com/sklearn-calibration



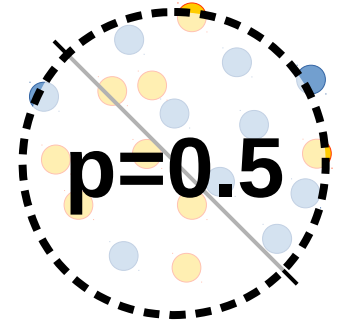
Epistemic (systematic) uncertainty: it gets tricky



Ideas?

How to measure uncertainty

Aleatoric uncertainty: use predicted probability!
Exception: neural networks can be **overconfident**
Fix it by *calibrating* model predictions after the fact,
Read more: tinyurl.com/sklearn-calibration



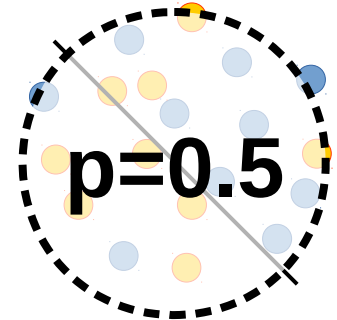
Epistemic (systematic) uncertainty: it gets tricky

Approach A: train *autoencoder* on input features
Low reconstruction error = **certain or not?**
High reconstruction error = **certain or not?**



How to measure uncertainty

Aleatoric uncertainty: use predicted probability!
Exception: neural networks can be **overconfident**
Fix it by *calibrating* model predictions after the fact,
Read more: tinyurl.com/sklearn-calibration



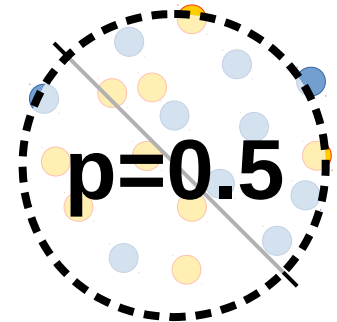
Epistemic (systematic) uncertainty: it gets tricky

Approach A: train *autoencoder* on input features
Low reconstruction error = familiar data
High reconstruction error = unfamiliar data
(For NLP: use *language models*)



How to measure uncertainty

Aleatoric uncertainty: use predicted probability!
Exception: neural networks can be **overconfident**
Fix it by *calibrating* model predictions after the fact,
Read more: tinyurl.com/sklearn-calibration



Epistemic (systematic) uncertainty: it gets tricky

Approach A: train *autoencoder* on input features
Low reconstruction error = familiar data
High reconstruction error = unfamiliar data



Approach B: train an *ensemble* of predictors
Predictors agree = familiar data
Predictors disagree = unfamiliar data

More: tinyurl.com/uncertainty-ensembles

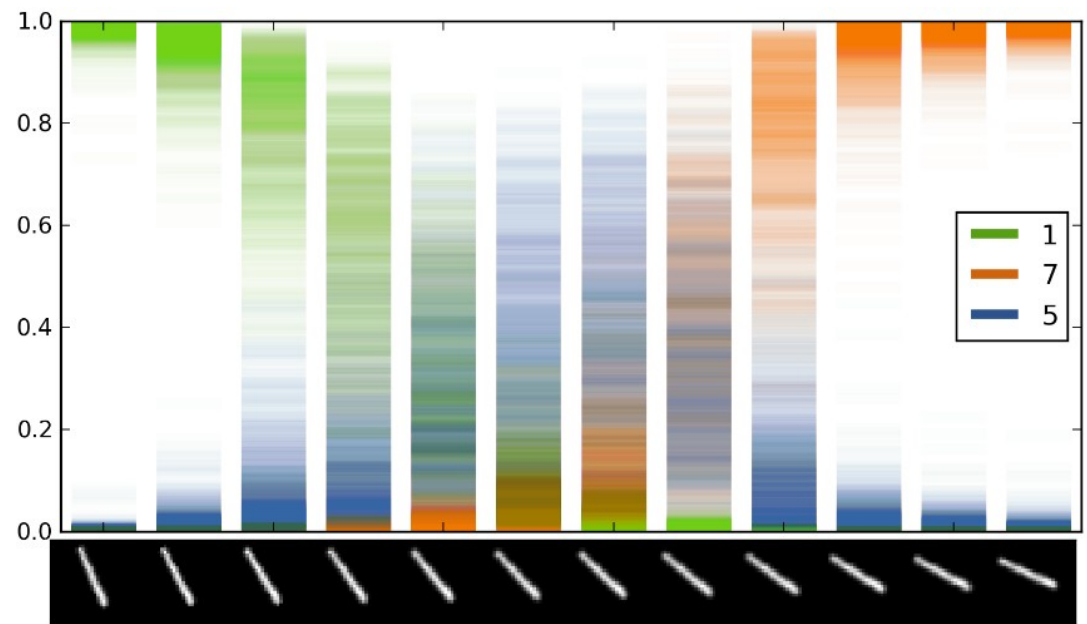
Uncertainty from dropout

Idea:

measure how robust
does your network
perform under noise

Example (left):

use dropout and
estimate variance



*Systematic uncertainty for different input
images, source: arXiv:1506.02142*

Read more in the [paper](#) or in a [blog post](#)

Bayesian Neural Networks

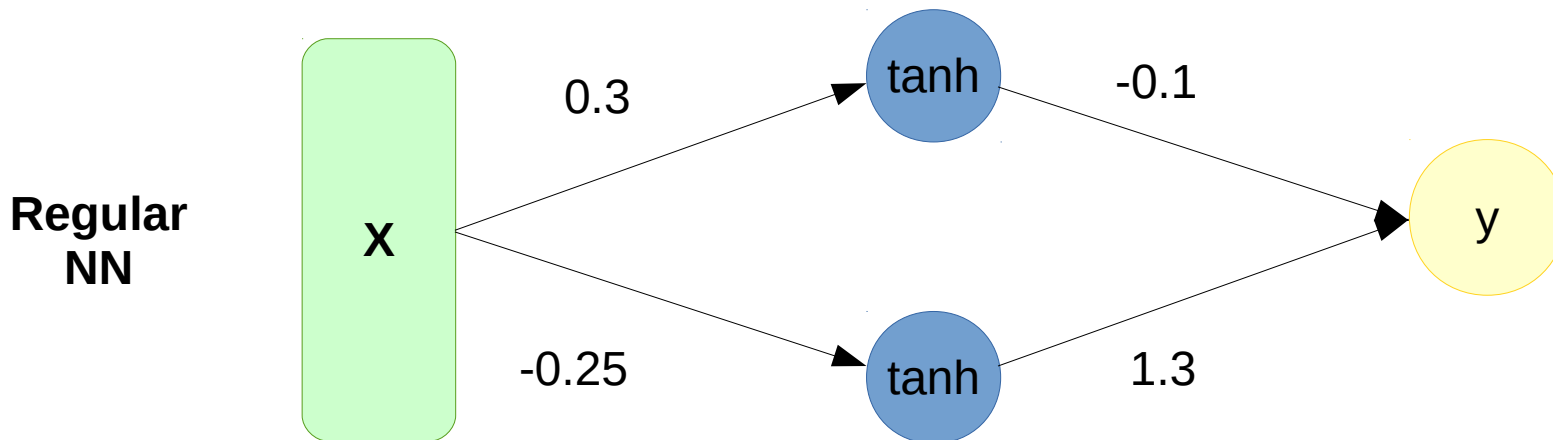
Disclaimer: this is a hacker's guide to BNNs!

It does not cover all the philosophy and general cases.

Bayesian Neural Networks

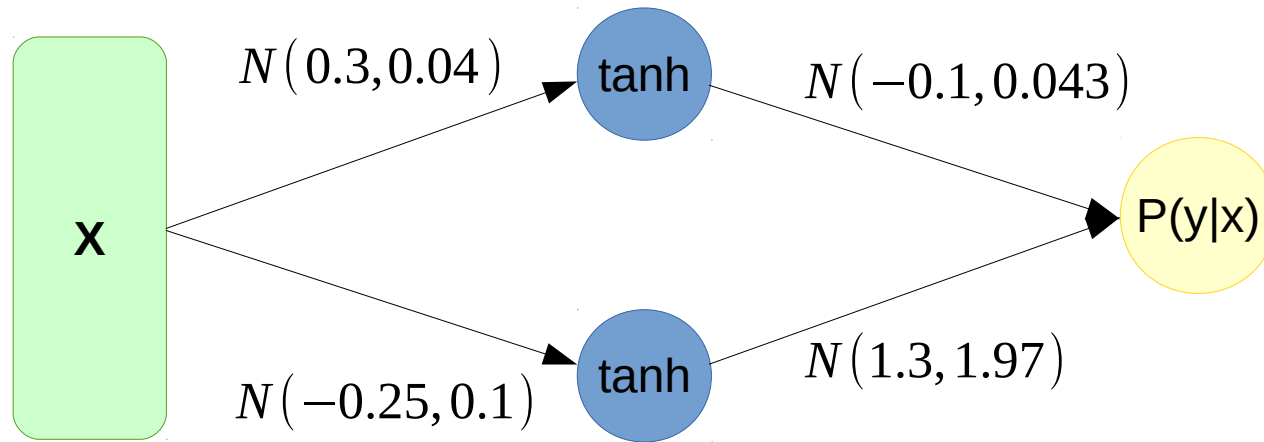
Disclaimer: this is a hacker's guide to BNNs!

It does not cover all the philosophy and general cases.

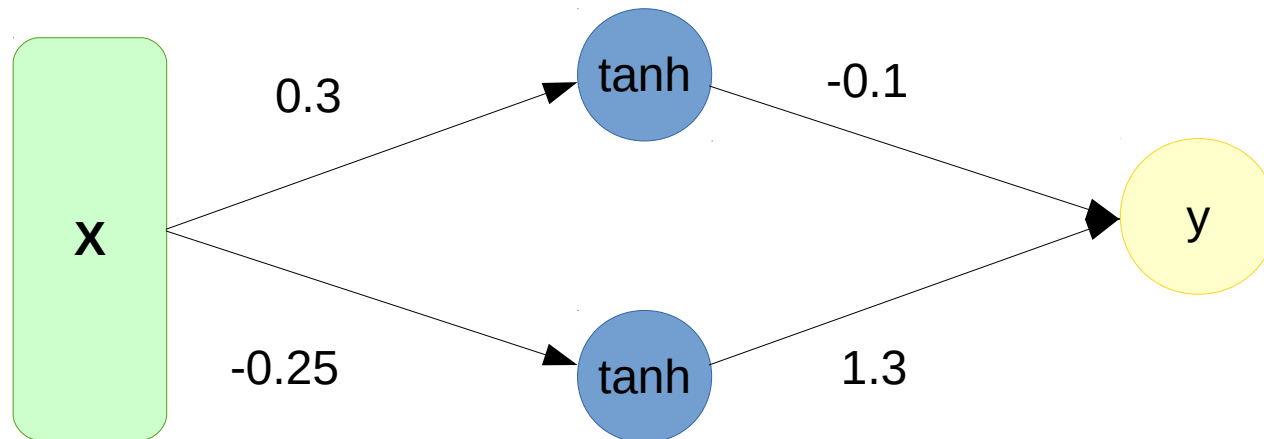


Bayesian Neural Networks

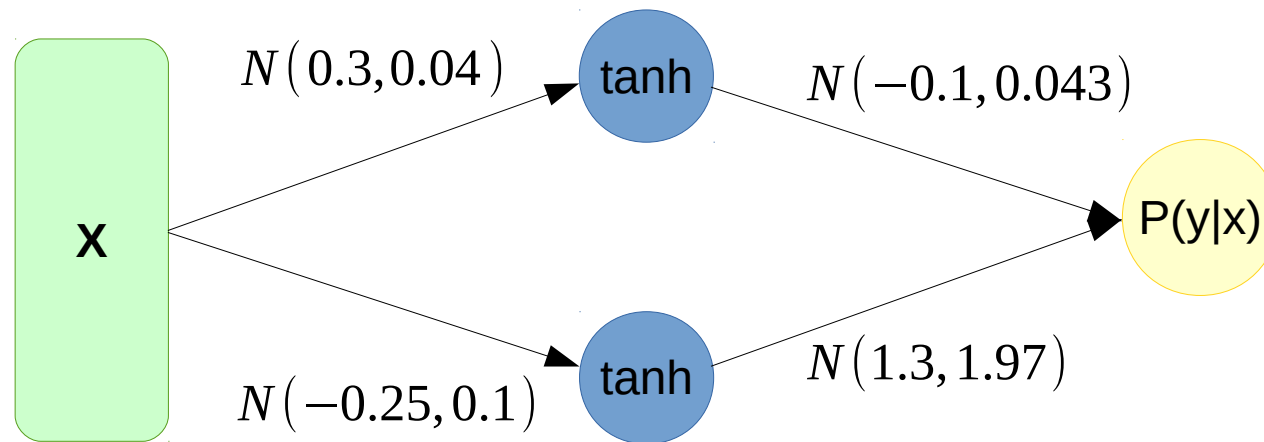
Bayesian
NN



Regular
NN



Bayesian Neural Networks



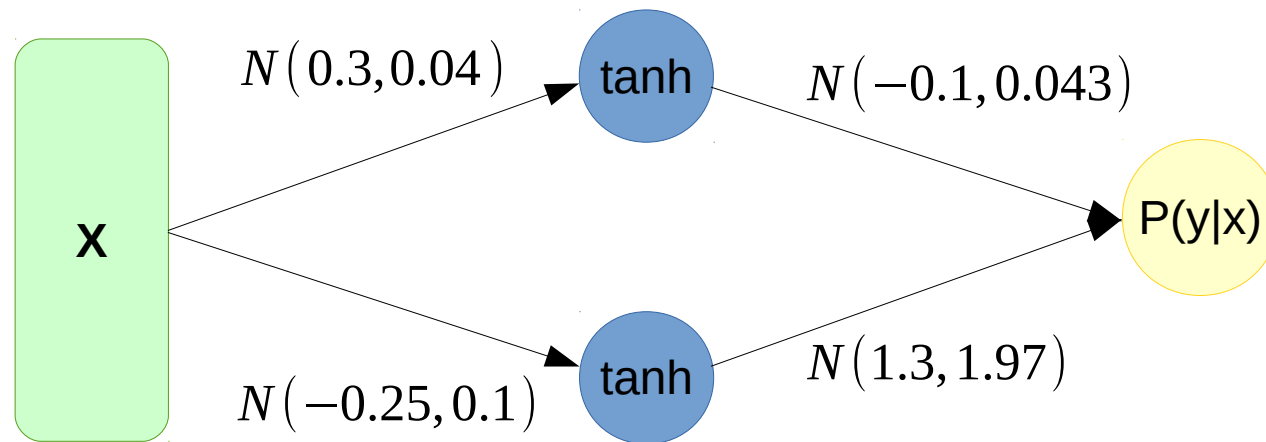
Idea:

- No explicit weights
 - Maintain parametric distribution on them instead!
 - Practical: fully-factorized normal or similar

$$q(\theta|\phi: [\mu, \sigma]) = \prod_i N(\theta_i | \mu_i, \sigma_i)$$

$$P(y|x) = E_{\theta \sim q(\theta|\phi)} P(y|x, \theta)$$

Bayesian Neural Networks



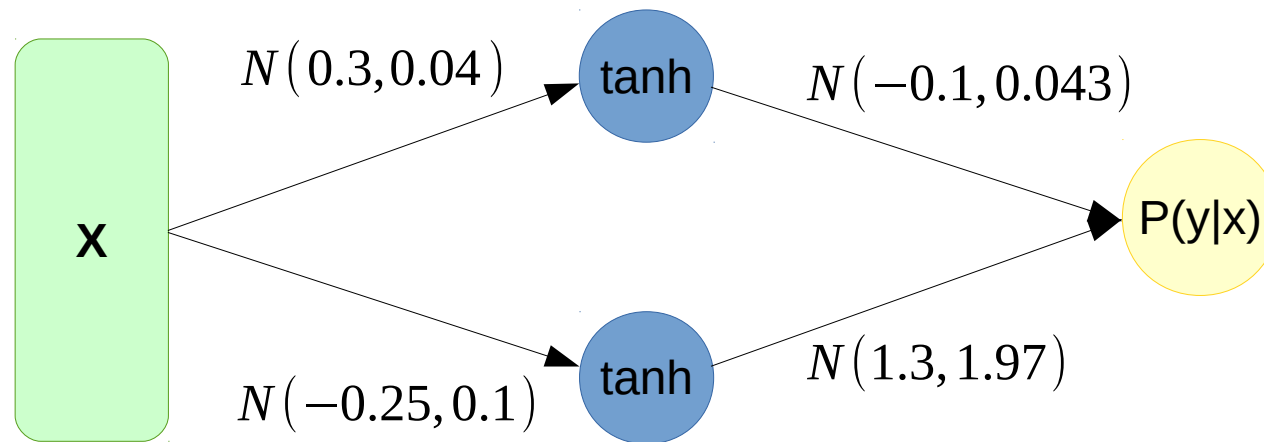
Idea:

- No explicit weights
 - Maintain parametric distribution on them instead!
 - Practical: fully-factorized normal or similar

$$q(\theta|\phi: [\mu, \sigma]) = \prod_i N(\theta_i | \mu_i, \sigma_i)$$

$$P(y|x) = E_{\theta \sim \underline{q(\theta|\phi)}} P(y|x, \theta)$$

Bayesian Neural Networks



Idea:

- No explicit weights
- Inference: sample from weight distributions, predict 1 “sample”
- To get distribution, aggregate K samples (e.g. with histogram)
 - Yes, it means running network **multiple times per one x**

$$P(y|x) = E_{\theta \sim \underline{q(\theta|\phi)}} P(y|x, \theta)$$

Bayesian Neural Networks

Idea:

- No explicit weights
 - Maintain parametric distribution on them instead!
 - Practical: fully-factorized normal or similar

$$q(\theta|\phi : [\mu, \sigma]) = \prod_i N(\theta_i | \mu_i, \sigma_i)$$

$$P(y|x) = E_{\theta \sim q(\theta|\phi)} P(y|x, \theta)$$

- Learn parameters of that distribution (reparameterization trick)
 - Less variance: local reparameterization trick.

$$\hat{\phi} = \operatorname{argmax}_{\phi} E_{x_i, y_i \sim d} E_{\theta \sim q(\theta|\phi)} P(y_i | x_i, \theta)$$

wanna explicit formulae? *d = dataset*

Evidence Lower bound

$d = \text{dataset}$

$$-KL(q(\theta|\phi) || p(\theta|d)) = -\int_{\theta} q(\theta|\phi) \cdot \log \frac{q(\theta|\phi)}{p(\theta|d)}$$

$$-\int_{\theta} q(\theta|\phi) \cdot \log \frac{q(\theta|\phi)}{\left[\frac{p(d|\theta) \cdot p(\theta)}{p(d)} \right]} = -\int_{\theta} q(\theta|\phi) \cdot \log \frac{q(\theta|\phi) \cdot p(d)}{p(d|\theta) \cdot p(\theta)}$$

$$-\int_{\theta} q(\theta|\phi) \cdot \left[\log \frac{q(\theta|\phi)}{p(\theta)} - \log p(d|\theta) + \log p(d) \right]$$

$$\left[E_{\theta \sim q(\theta|\phi)} \log p(d|\theta) \right] - KL(q(\theta|\phi) || p(\theta)) + \log p(d)$$

loglikelihood

-distance to prior

+const

Evidence Lower bound

$$\phi = \underset{\phi}{\operatorname{argmax}} (-KL(q(\theta|\phi) || p(\theta|d)))$$

$$\underset{\phi}{\operatorname{argmax}} \left(\underbrace{[E_{\theta \sim q(\theta|\phi)} \log p(d|\theta)]}_{\text{fit to the data}} - \underbrace{KL(q(\theta|\phi) || p(\theta))}_{\text{don't be too certain}} \right)$$

Evidence Lower bound

$$\phi = \underset{\phi}{\operatorname{argmax}} (-KL(q(\theta|\phi) \| p(\theta|d)))$$

$$\underset{\phi}{\operatorname{argmax}} ([E_{\theta \sim q(\theta|\phi)} \log p(d|\theta)] - KL(q(\theta|\phi) \| p(\theta)))$$

Can we perform gradient ascent directly?

Reparameterization trick

$$\phi = \underset{\phi}{\operatorname{argmax}} (-KL(q(\theta|\phi) \| p(\theta|d)))$$

$$\underset{\phi}{\operatorname{argmax}} \left(\underbrace{[E_{\theta \sim q(\theta|\phi)} \log p(d|\theta)]}_{\text{Use reparameterization trick}} - \underbrace{KL(q(\theta|\phi) \| p(\theta))}_{\text{simple formula (for normal q)}} \right)$$

BNN likelihood

What does this $\log P(d|...)$ mean?

$$E_{\theta \sim N(\theta|\mu_\phi, \sigma_\phi)} \log p(d|\theta) = E_{\psi \sim N(0,1)} \log p(d|(\mu_\phi + \sigma_\phi \cdot \psi))$$

Reparameterization trick

$$\phi = \underset{\phi}{\operatorname{argmax}} (-KL(q(\theta|\phi) \| p(\theta|d)))$$

$$\underset{\phi}{\operatorname{argmax}} ([E_{\theta \sim q(\theta|\phi)} \log p(d|\theta)] - KL(q(\theta|\phi) \| p(\theta)))$$

BNN likelihood

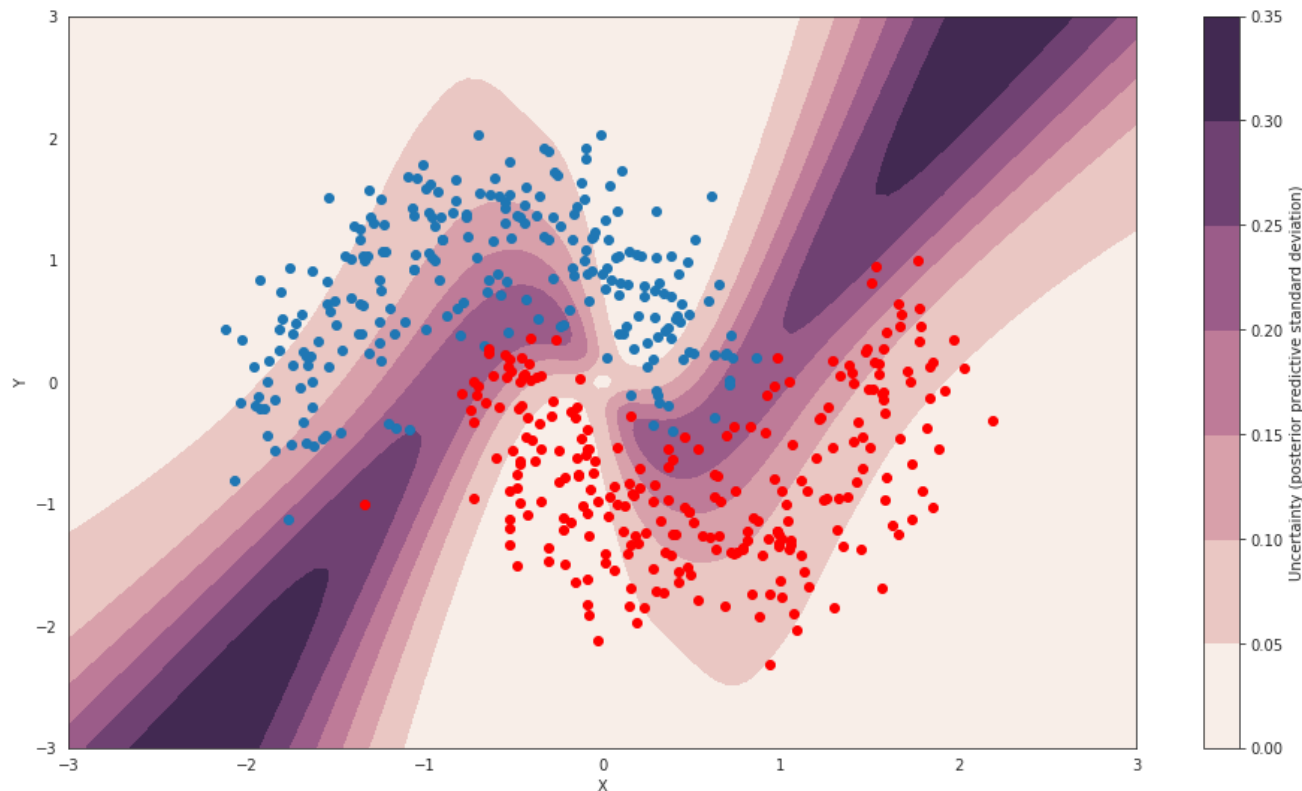
In other words,
 $\sum_{x,y \sim d} \log p(y|x, \mu + \sigma\psi)$

$$E_{\theta \sim N(\theta|\mu_\phi, \sigma_\phi)} \log p(d|\theta) = E_{\psi \sim N(0,1)} \log p(d|(\mu_\phi + \sigma_\phi \cdot \psi))$$

Bayesian Neural Networks

Estimating uncertainty:

1. sample weights several times
2. predict by averaging outputs
3. uncertainty = standard deviation



Read more...

Papers on uncertainty

bayesian neural networks: [blog post](#)
prior networks: arxiv.org/abs/1802.10501
batchnorm: arxiv.org/abs/1802.04893
dropout: arxiv.org/abs/1506.02142
video stuff: youtube.com/watch?v=HRfDiqgh6CE

The question of trust

How can I explain my model's prediction?

Why did it make this decision/mistake?

What features does it rely on?

Is my model certain about what it says?

Is there something wrong with this input?

Can I rely on this prediction?

Can I trust this data?

Is something missing?

Is there any bias?

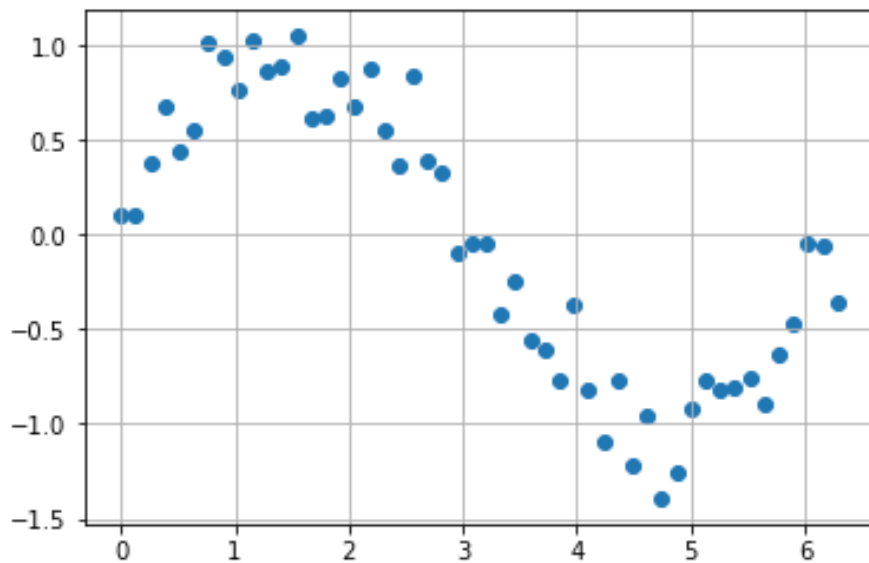
Exploratory data analysis

aka “seeing for yourself what’s in your data”

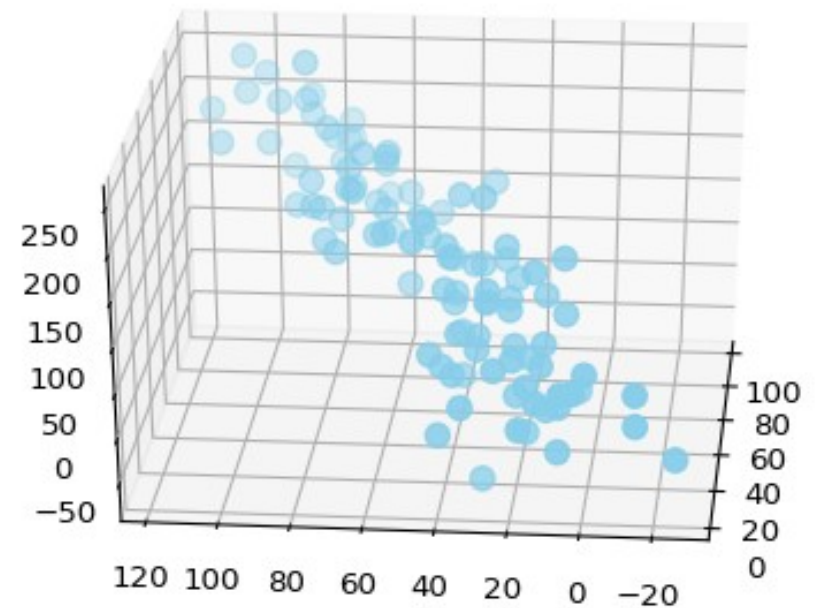
Q: How many dimensions can you show on a plot?

Exploratory data analysis

Q: How many dimensions can you show on a plot?



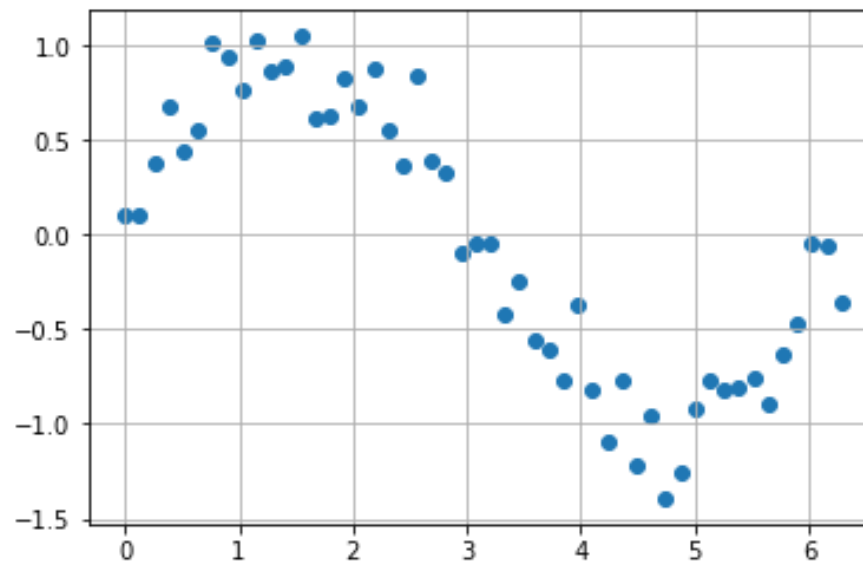
2d scatter-plot



3d scatter-plot

Exploratory data analysis

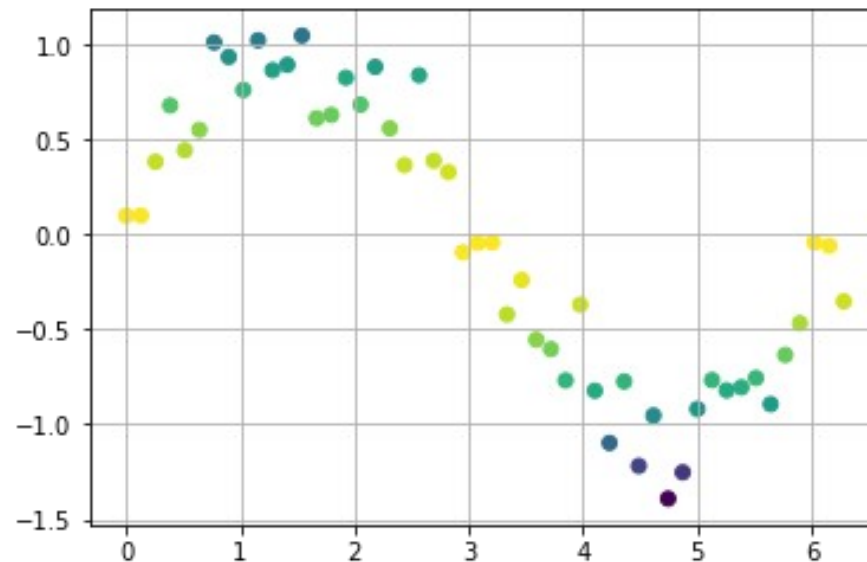
Q: How many dimensions can you show on a plot?



2 dimensions

Exploratory data analysis

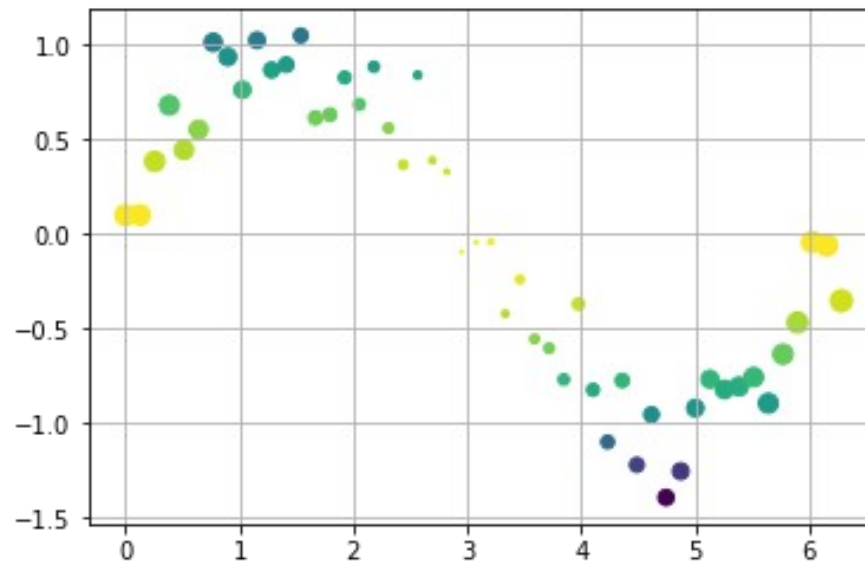
Q: How many dimensions can you show on a plot?



3 dimensions

Exploratory data analysis

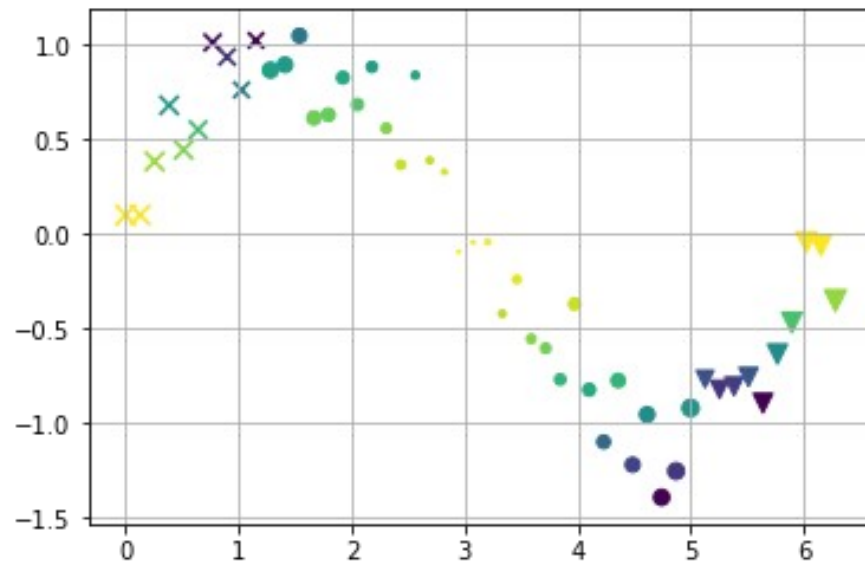
Q: How many dimensions can you show on a plot?



4 dimensions

Exploratory data analysis

Q: How many dimensions can you show on a plot?



5 dimensions

Exploratory data analysis

Q: How many dimensions can you show on a plot?

Your data has 200 dimensions...
any ideas?

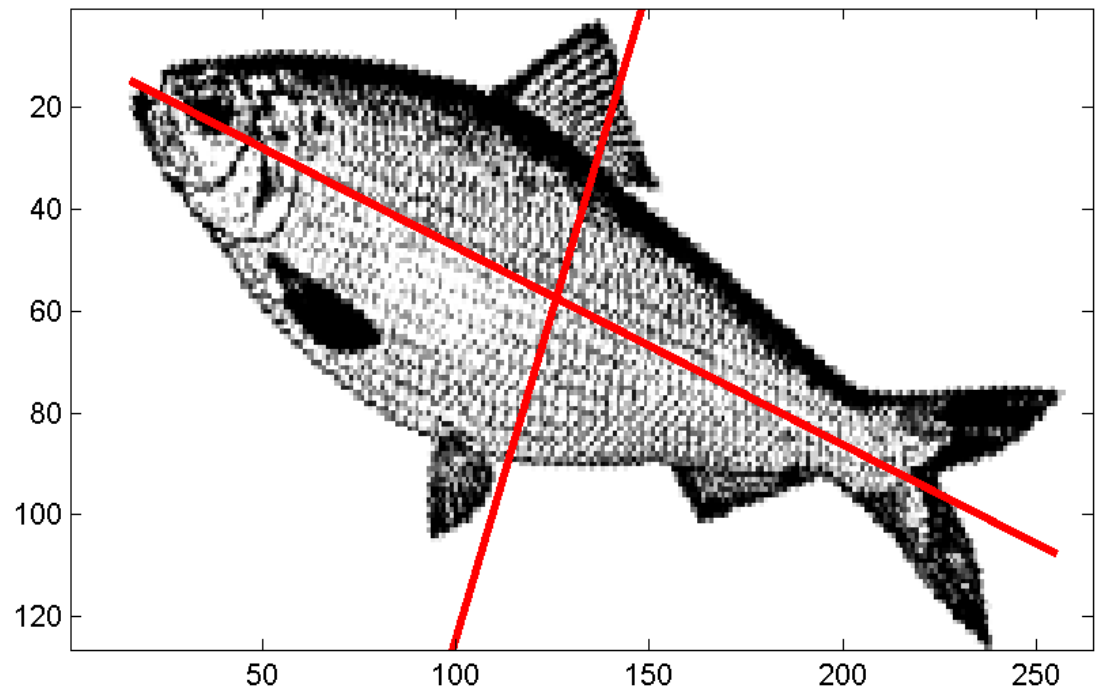
Recap: Principal Component Analysis

Idea:

- Linearly project data to lower-dim space

$$X \approx (X \times W_1) \times W_2$$

Minimize MSE

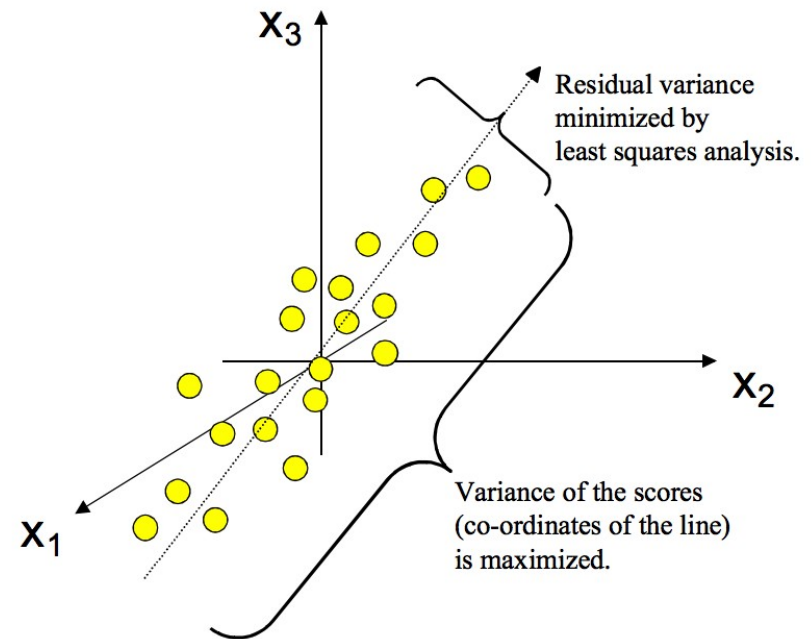


$$\operatorname{argmin}_{W_1, W_2} \|X - (X \times W_1) \times W_2\|$$

Recap: Principal Component Analysis

Idea:

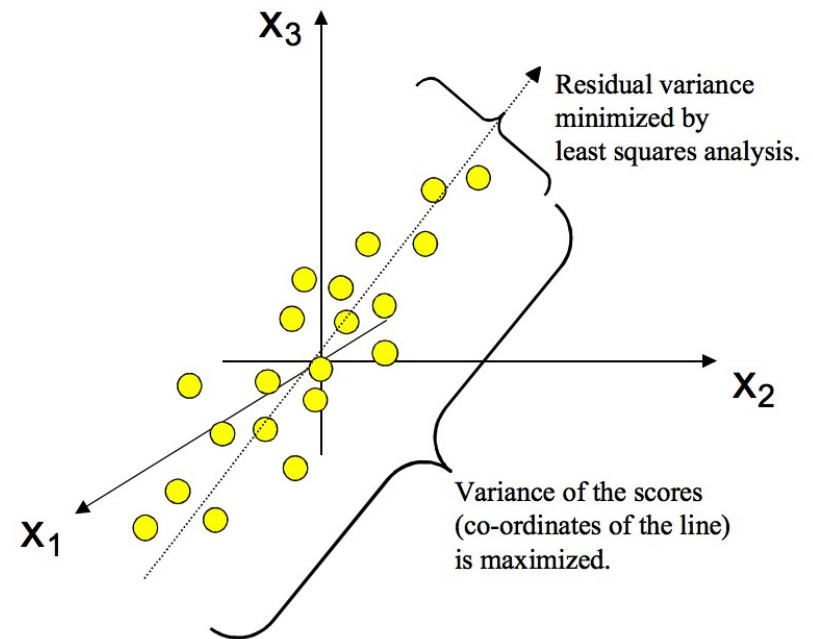
- Linearly project data to lower-dim space
- Attempt to preserve as much variance as possible



Recap: Principal Component Analysis

Idea:

- Linearly project data to lower-dim space
- Attempt to preserve as much variance as possible

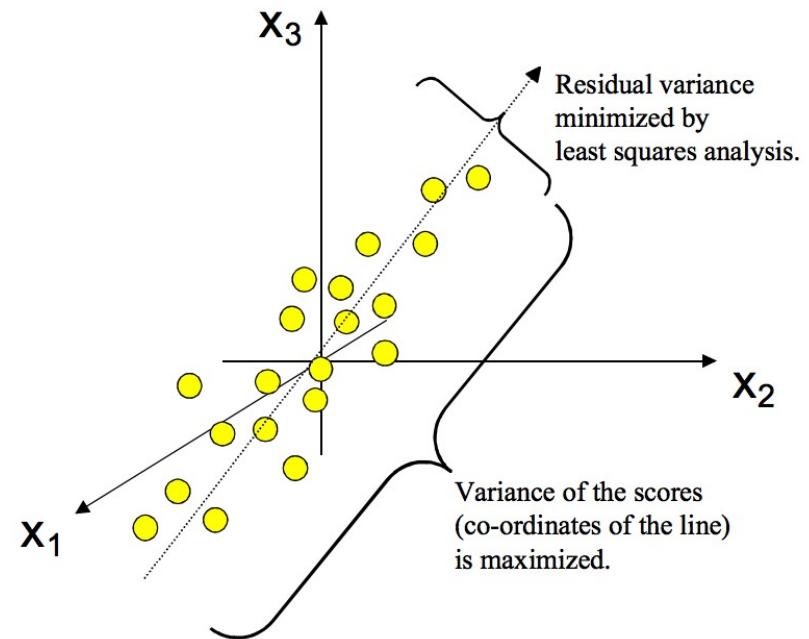


Q: What if linear projection is not enough?

Recap: Principal Component Analysis

Idea:

- Linearly project data to lower-dim space
- Attempt to preserve as much variance as possible



Q: What if linear projection is not enough?
deep autoencoders... or better

Manifold learning

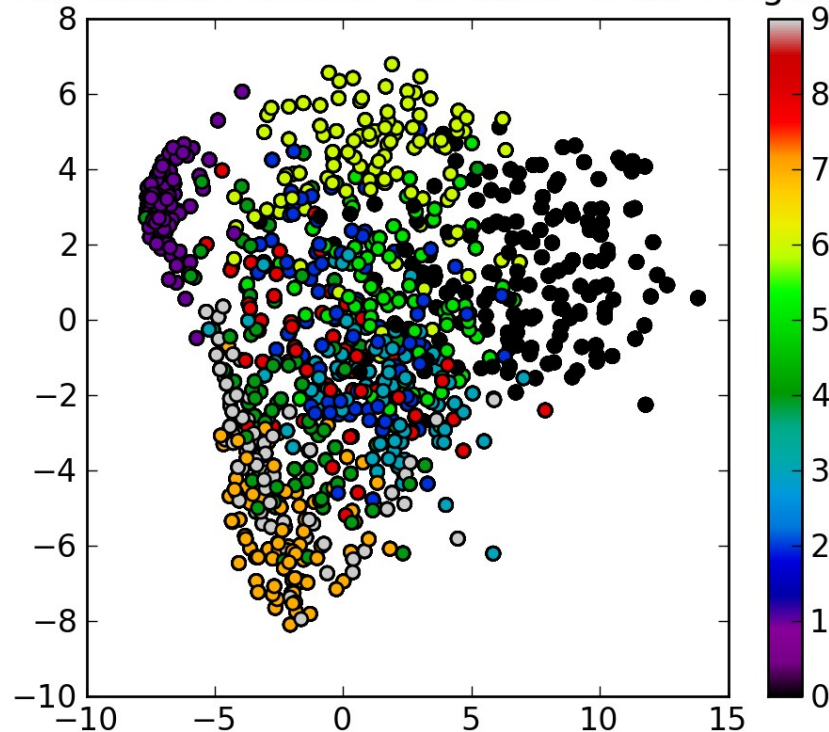
Idea: let's directly “learn” 2d point coordinates

Multidimensional Scaling

preserve pairwise distances

$$\hat{x} = \underset{\hat{x}}{\operatorname{argmin}} \frac{2}{N^2 - N} \sum_{i \neq j} (\|x_i - x_j\| - \|\hat{x}_i - \hat{x}_j\|)^2$$

MDS classical solution for 1000 random digits



Stochastic Neighborhood Embedding

preserve neighbor “probabilities”

$$P_{j|i} = \frac{e^{-\|x_i - x_j\|_2^2}}{\sum_k e^{-\|x_k - x_j\|_2^2}}$$

- large for nearest neighbors
- small for distant points
- adds up to 1

$$\hat{P}_{j|i} = \frac{e^{-\|\hat{x}_i - \hat{x}_j\|_2^2}}{\sum_k e^{-\|\hat{x}_k - \hat{x}_j\|_2^2}}$$

- same as P
- but in learned space

optimize crossentropy w.r.t. \hat{x}

$$\hat{x} = \operatorname{argmin}_{\hat{x}} -\frac{1}{N} \sum_i \sum_j P_{j|i} \cdot \log \hat{P}_{j|i}$$

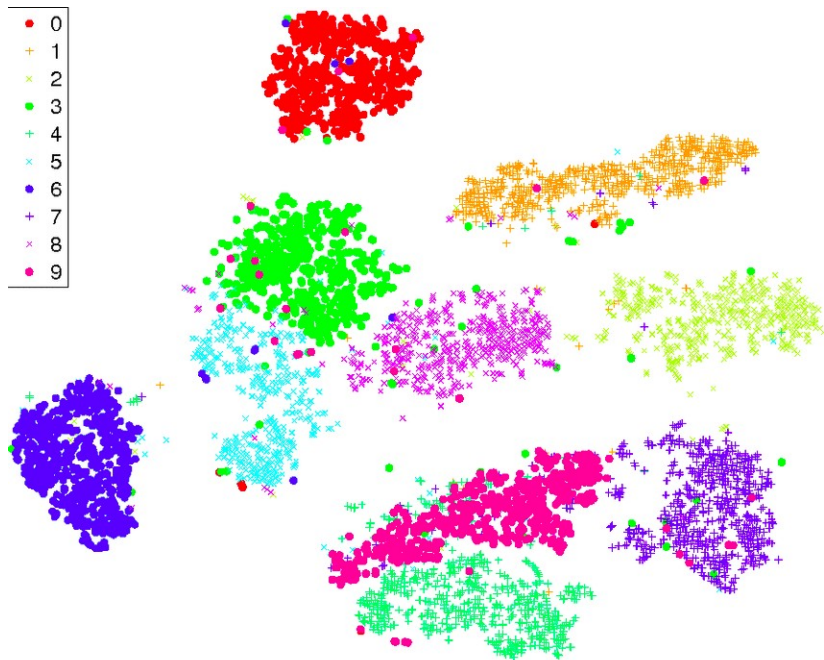
T-SNE

Like SNE from prev slide, but

- P is now *Student's t-distribution*

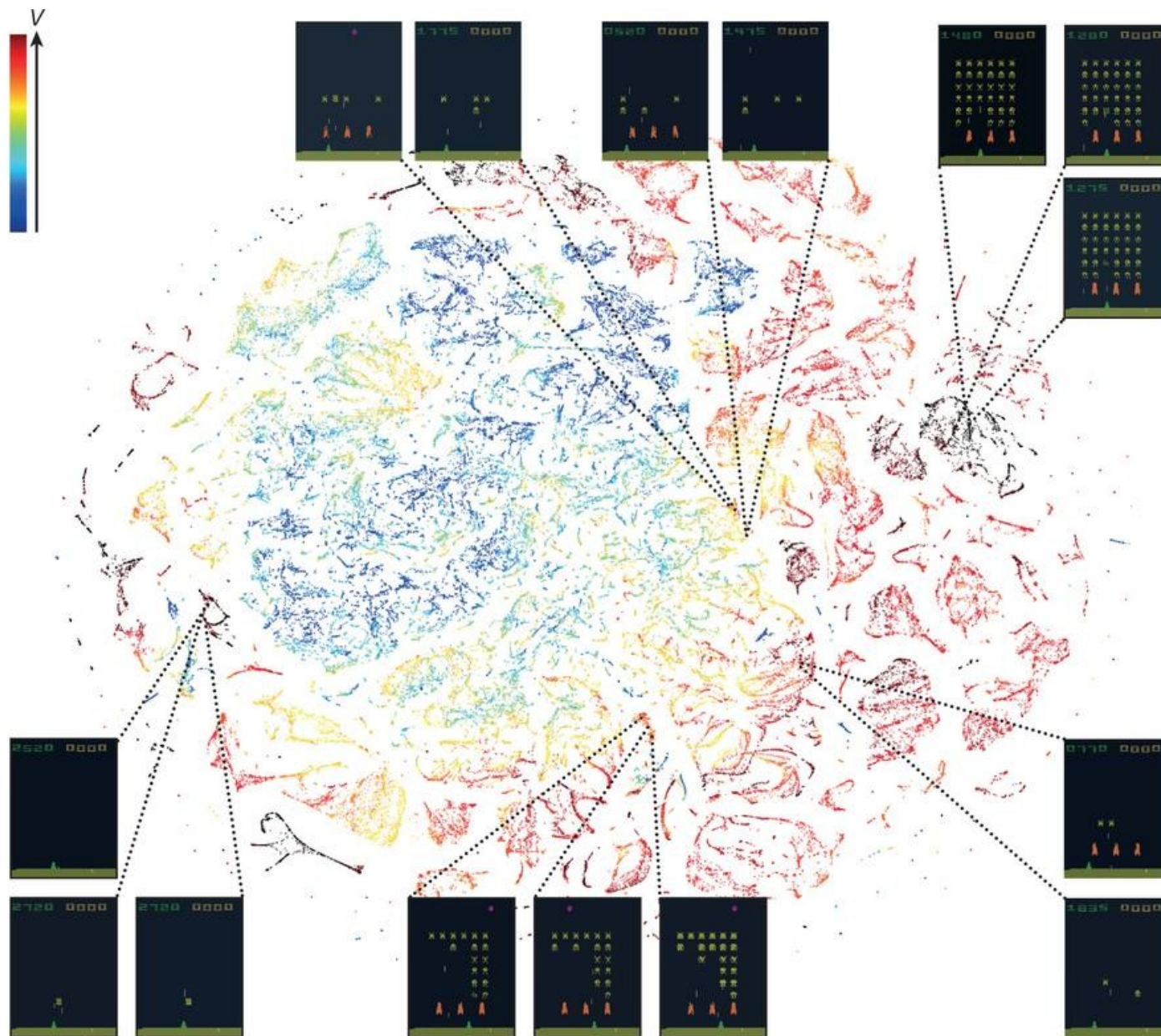
$$\hat{P}_{j|i} = \frac{(1 + \|\hat{x}_i - \hat{x}_j\|_2^2)^{-1}}{\sum_{k \neq l} (1 + \|\hat{x}_k - \hat{x}_l\|_2^2)^{-1}}$$

- A lot of optimization hacks
- By far the most popular method

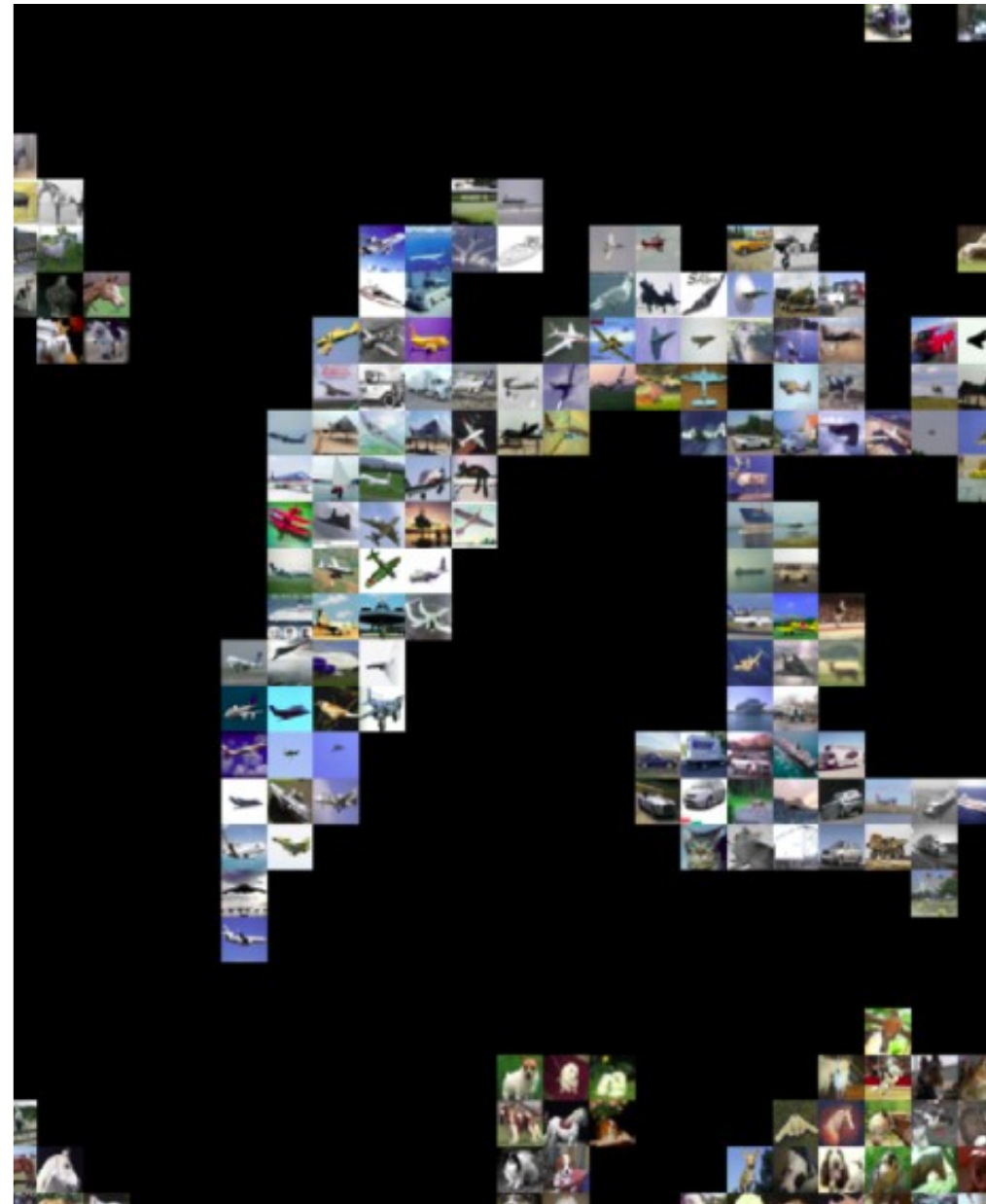
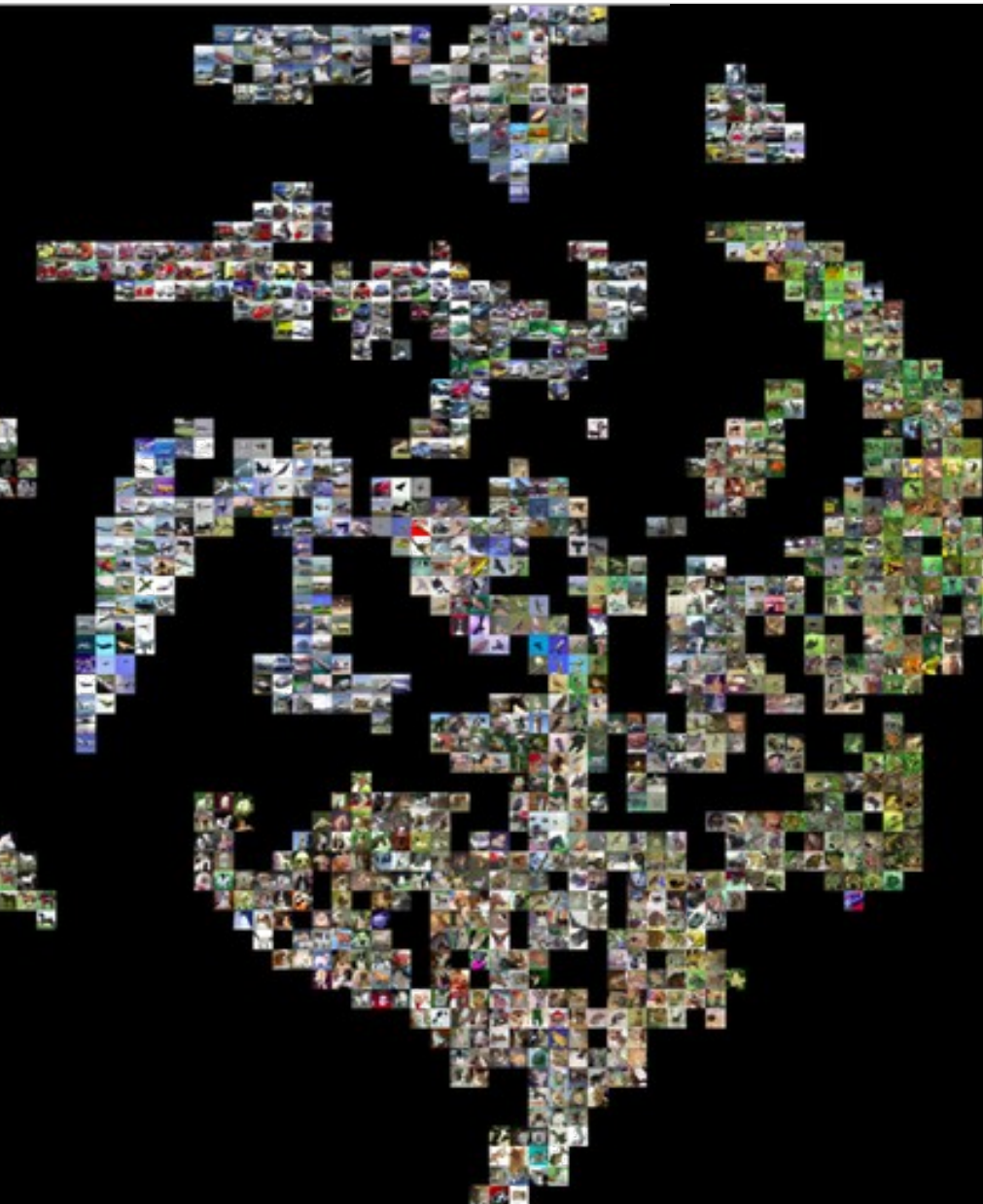


Read More:
[Original paper](#)
[Interactive demo](#)

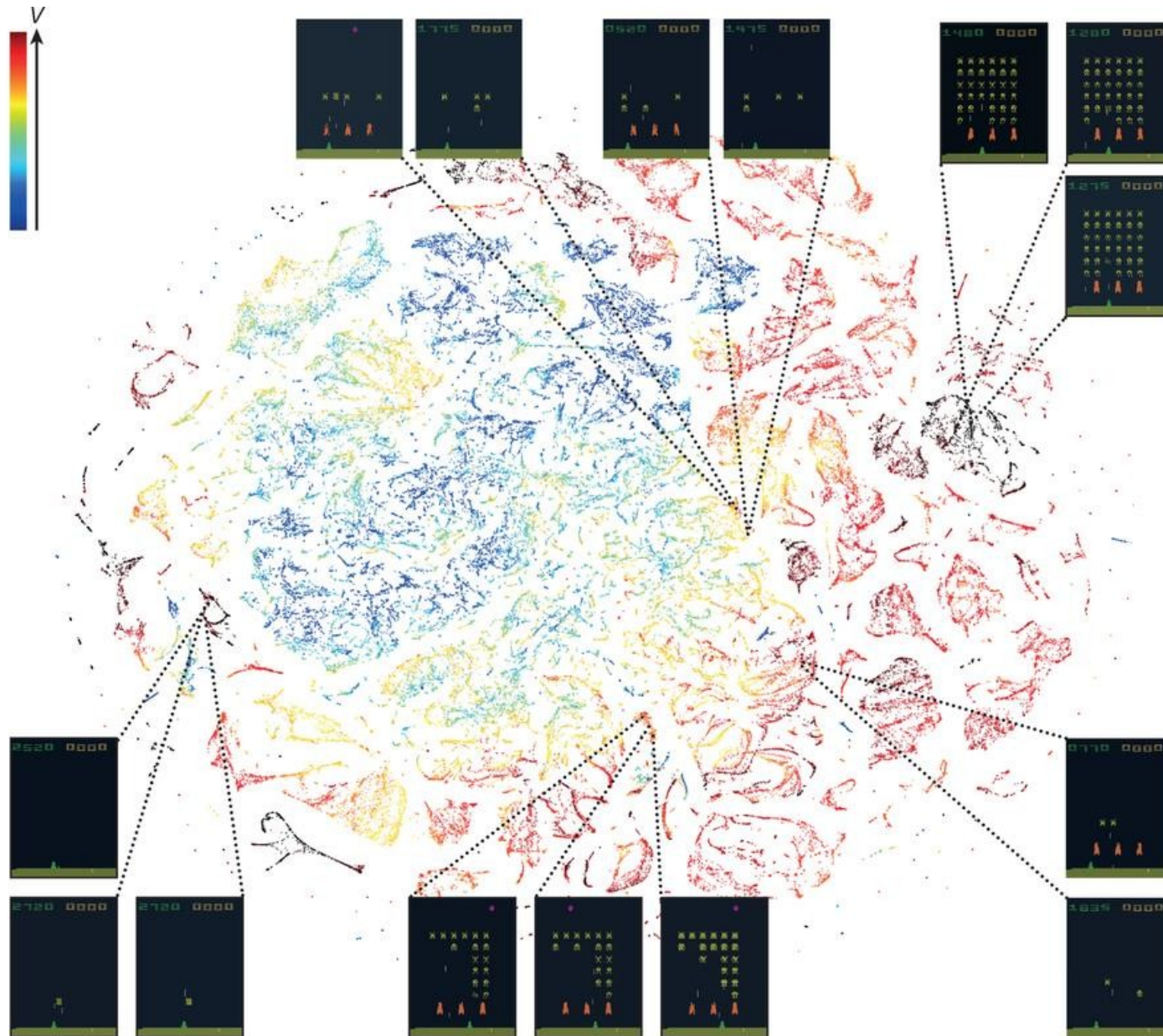
T-SNE + deep encoder



T-SNE + deep encoder (CIFAR10)



T-SNE + deep encoder (atari DQN)



Thank you

[question time!]