Lecture 7: Intro to recognition and linear ML

COS 429: Computer Vision

PRINCETON UNIVERSITY

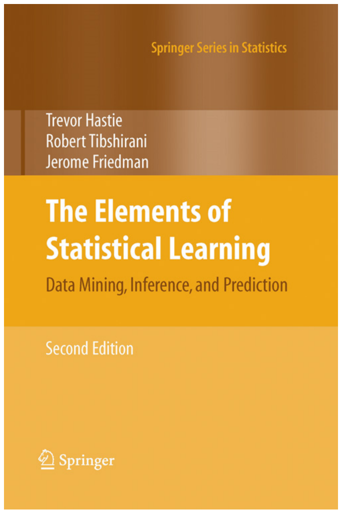
Slides adapted from D. Fouhey

Terminology

- ML is incredibly messy terminology-wise.
- Most things have at lots of names.
- I will try to write down multiple of them so if you see it later you'll know what it is.

Credit: D. Fouhey

Pointers



Useful book (Free too!):
The Elements of Statistical Learning
Hastie, Tibshirani, Friedman https://web.stanford.edu/~hastie/ElemStatLearn/

UCI Machine Learning Repository

Useful set of data:
UCI ML Repository
https://archive.ics.uci.edu/ml/datasets.html

A lot of important and hard lessons summarized:
https://homes.cs.washington.edu/~pedrod/papers/cacm12.pdf

Credit: D. Fouhey

Machine Learning (ML)

- Goal: make “sense” of data
- Overly simplified version: transform vector **x** into vector **y**=T(**x**) that’s somehow better
- Potentially you fit T using pairs of datapoints and desired outputs ($\mathbf{x}_i$,$\mathbf{y}_i$), or just using a set of datapoints ($\mathbf{x}_i$)
- Always are trying to find some transformation that minimizes or maximizes some **objective function** or goal.

Credit: D. Fouhey

Machine Learning

Input: **x**

Feature vector/Data point: Vector representation of datapoint. Each dimension or "**feature**" represents some aspect of the data.

Output: **y**

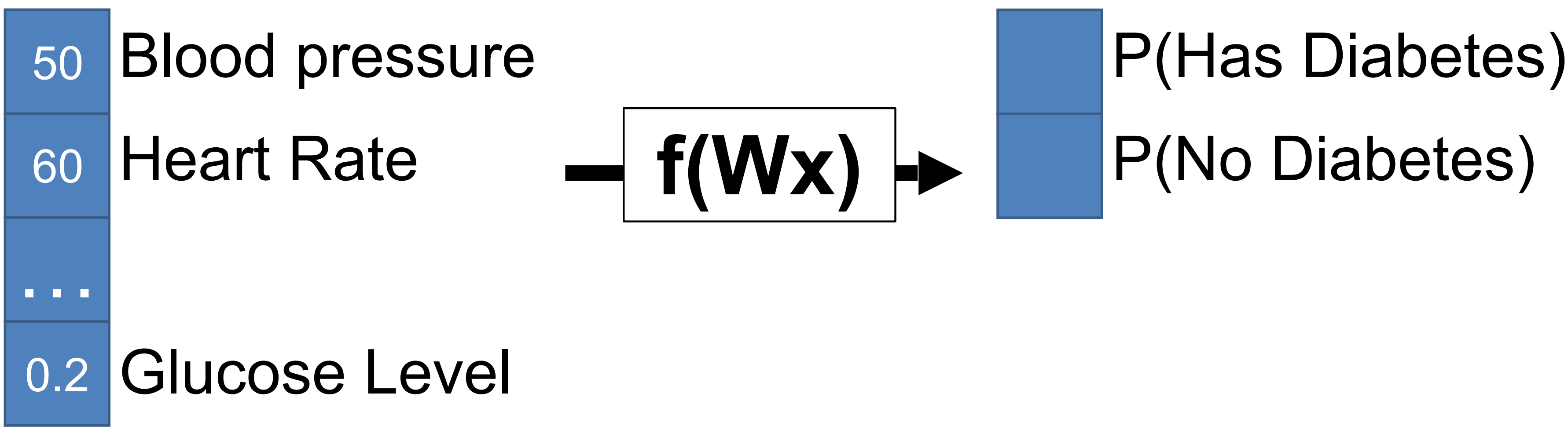
Label / target: Fixed length vector of desired output. Each dimension represents some aspect of the output data

Supervised: we are given y.
Unsupervised: we are not, and make our own ys.

Credit: D. Fouhey

Example – Health

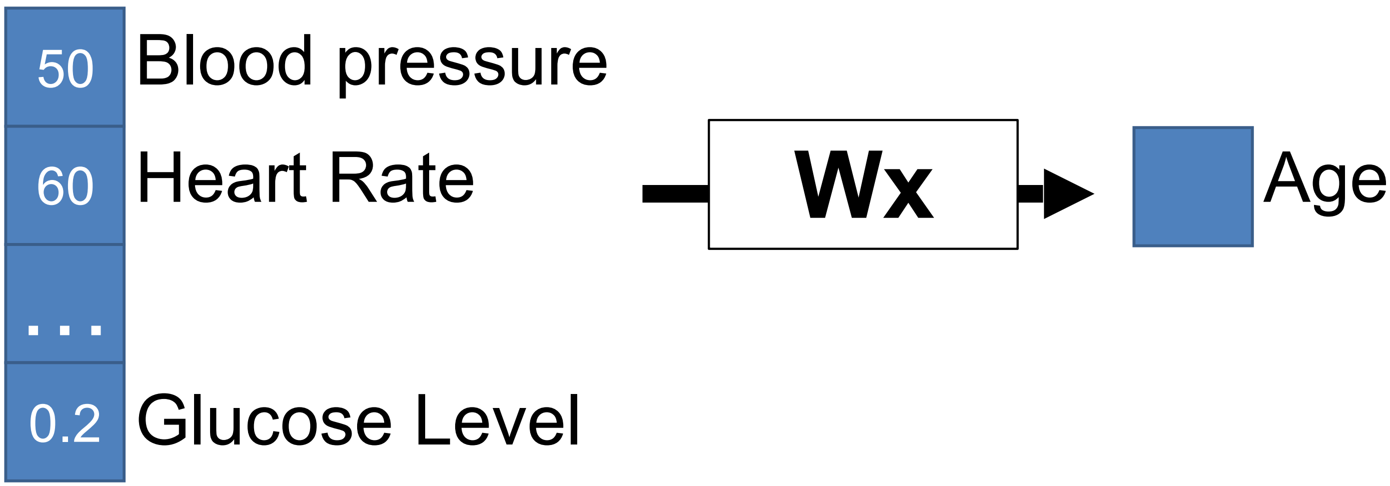
Input: **x** in R^N Output: **y**



Intuitive objective function: Want correct category to be likely with our model.

Example – Health

Input: $\mathbf{x}$ in R^N Output: $\mathbf{y}$



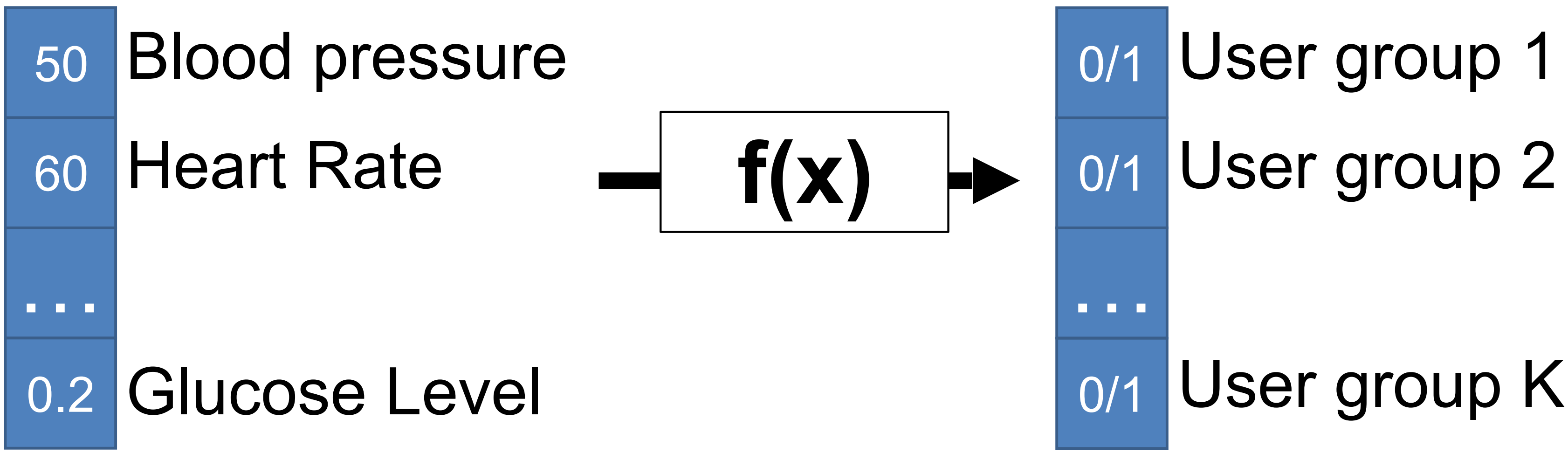
Intuitive objective function: Want our prediction of age to be "close" to true age.

Credit: D. Fouhey

Example – Health

Input: $\mathbf{x}$ in R^N

Output: **discrete y (unsupervised)**



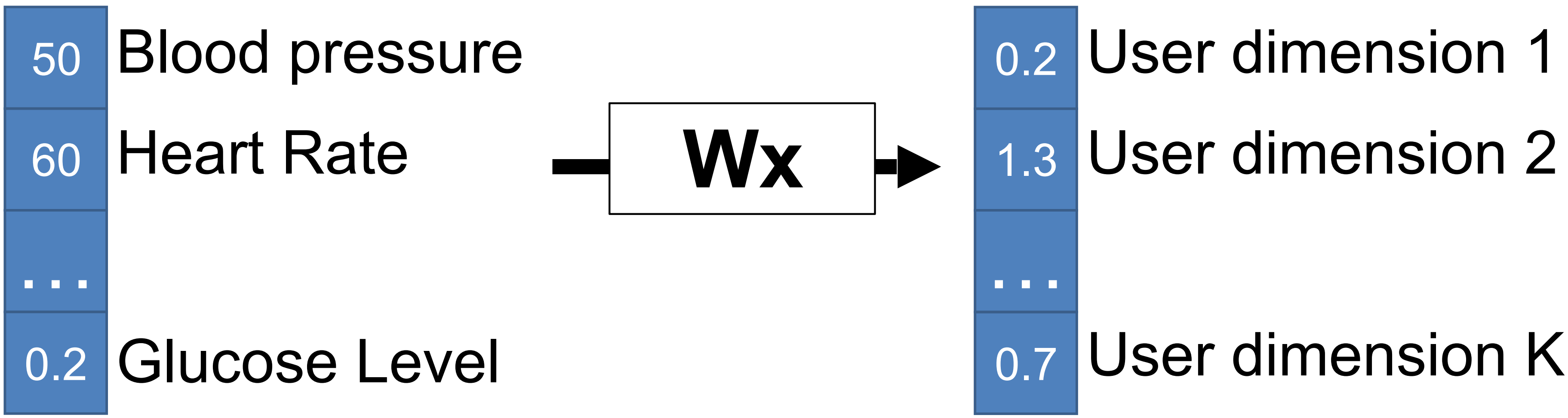
Intuitive objective function: Want to find K groups that explain the data we see.

Credit: D. Fouhey

Example – Health

Input: **x** in R^N

Output: **continuous y (discovered)**



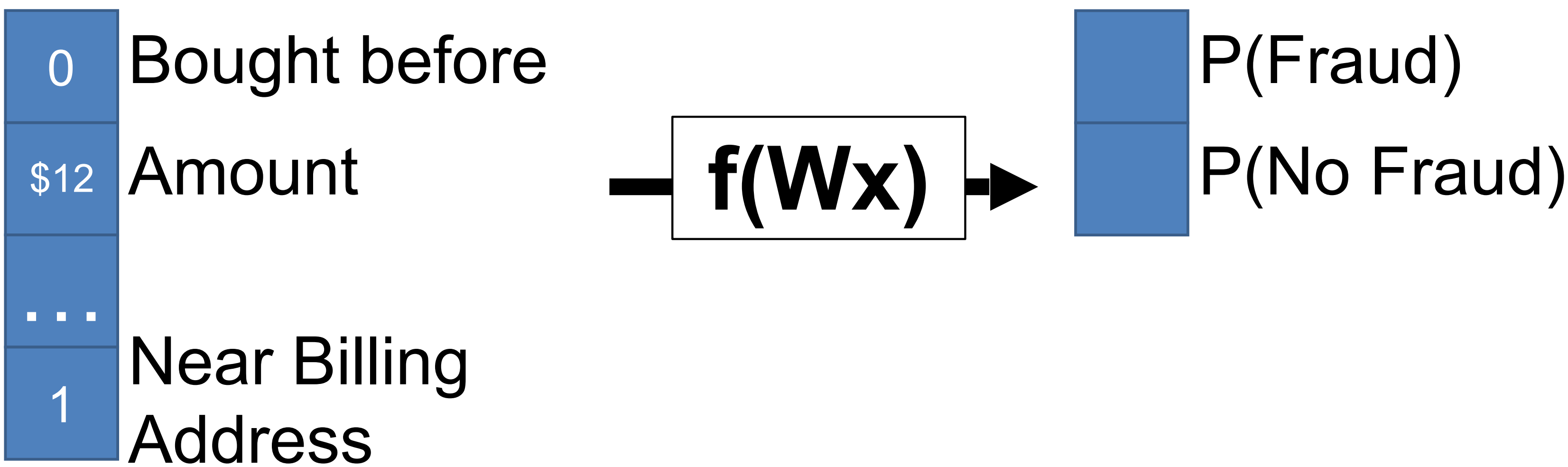
Intuitive objective function: Want to K dimensions (often two) that are easier to understand but capture the variance of the data.

Credit: D. Fouhey

Example – Credit Card Fraud

Input: **x** in R^N

Output: **y**

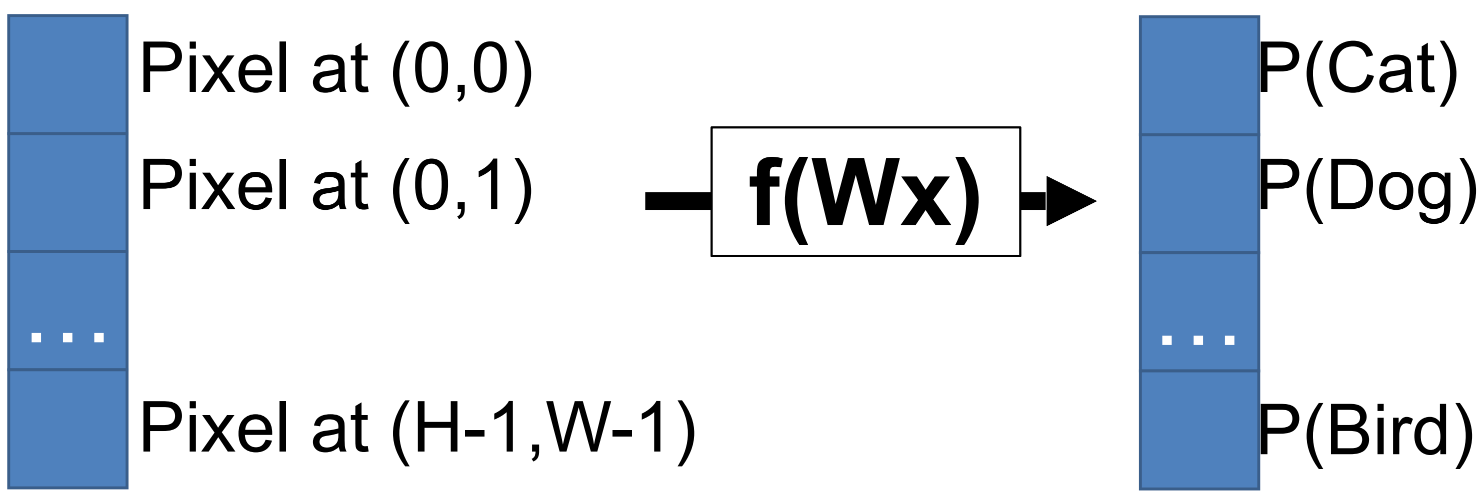


Intuitive objective function: Want correct category to be likely with our model.

Credit: D. Fouhey

Example – Computer Vision

Input: $\mathbf{x}$ in R^N Output: $\mathbf{y}$



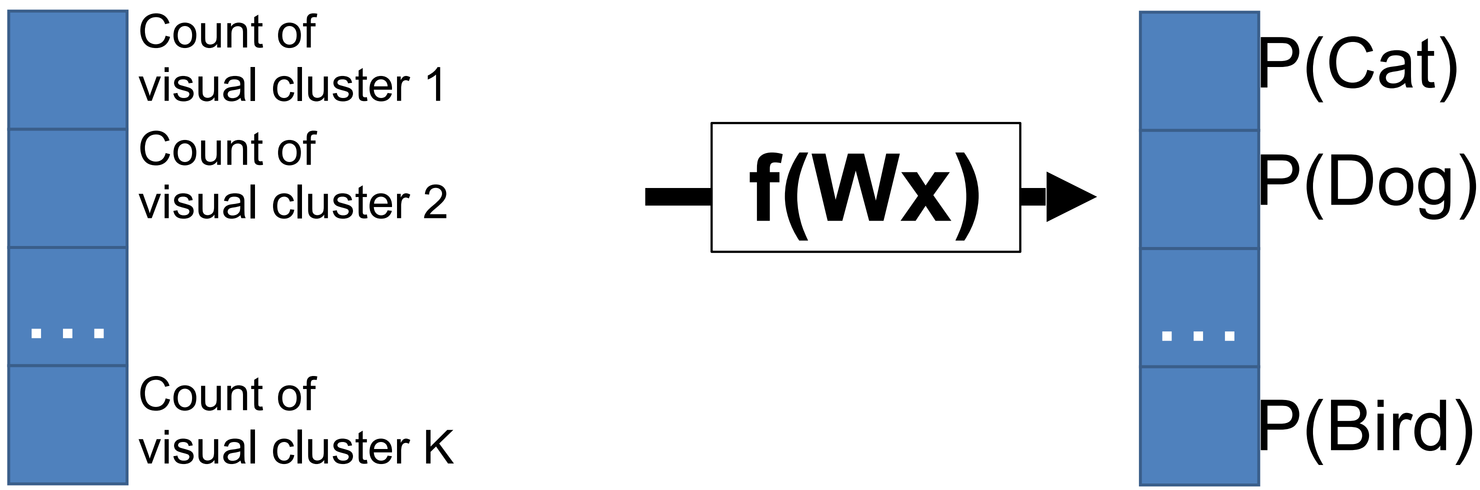
Intuitive objective function: Want correct category to be likely with our model.

Credit: D. Fouhey

Example – Computer Vision

Input: **x** in R^N

Output: **y**



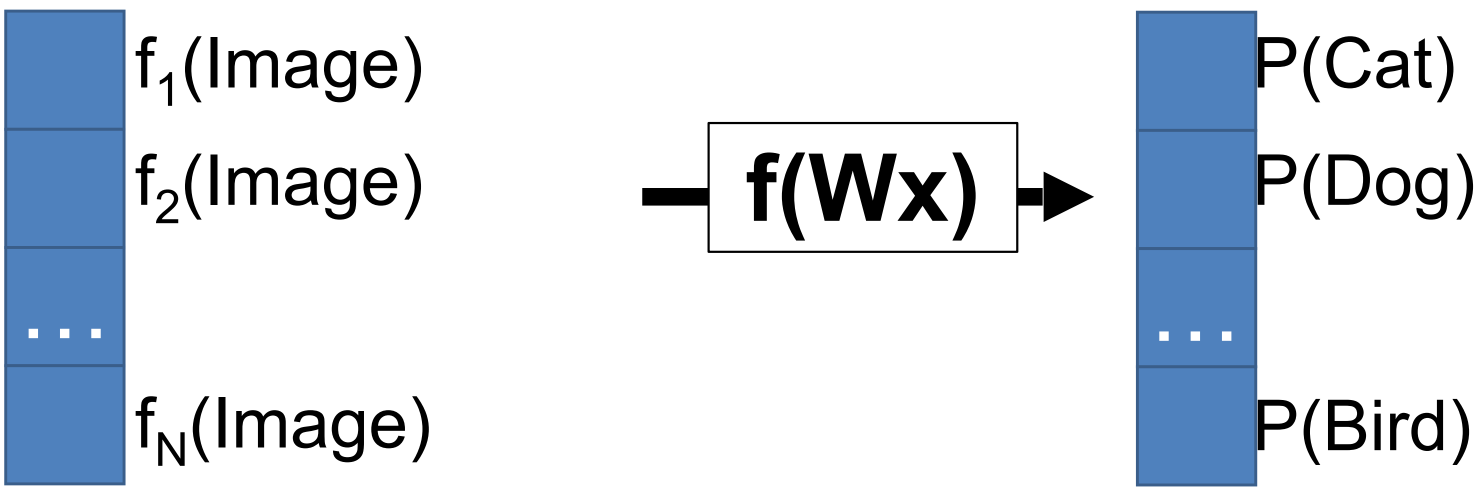
Intuitive objective function: Want correct category to be likely with our model.

Credit: D. Fouhey

Example – Computer Vision

Input: $\mathbf{x}$ in R^N

Output: $\mathbf{y}$



Intuitive objective function: Want correct category to be likely with our model.

Credit: D. Fouhey

Abstractions

- Throughout, assume we've converted data into a fixed-length feature vector. There are well-designed ways for doing this.
- But remember it could be big!
  - Image (e.g., 224x224x3): 151K dimensions
  - Patch (e.g., 32x32x3) in image: 3072 dimensions

Credit: D. Fouhey

ML Problems in Vision

Credit: D. Fouhey

Image credit: Wikipedia

ML Problem Examples in Vision

	Supervised (Data+Labels)	Unsupervised (Just Data)
Discrete Output	Classification/ Categorization	
Continuous Output		

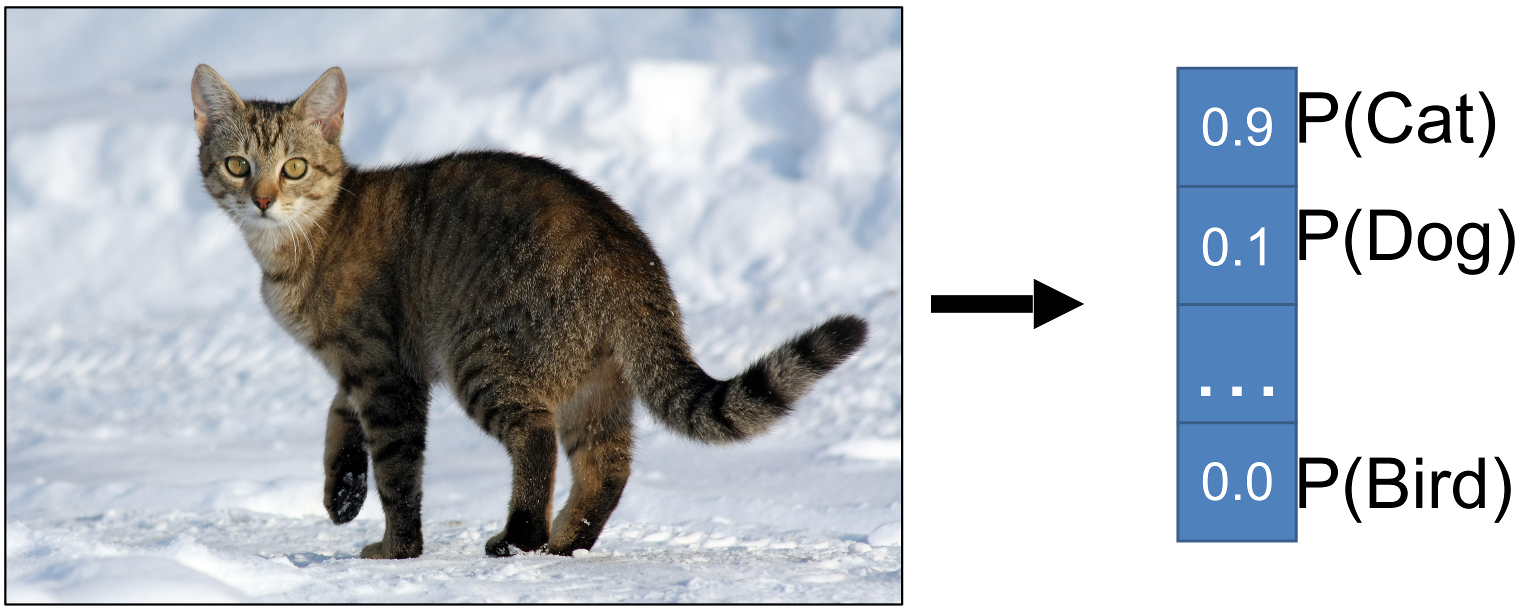
Credit: D. Fouhey

Slide adapted from J. Hays

ML Problem Examples in Vision

Categorization/Classification

Binning into K mutually-exclusive categories



Credit: D. Fouhey

Image credit: Wikipedia

ML Problem Examples in Vision

	Supervised (Data+Labels)	Unsupervised (Just Data)
Discrete Output	Classification/ Categorization	
Continuous Output	**Regression**	

ML Problem Examples in Vision

Regression

Estimating continuous variable(s)

→ 3.6 kg Cat weight

Credit: D. Fouhey

Image credit: Wikipedia

ML Problem Examples in Vision

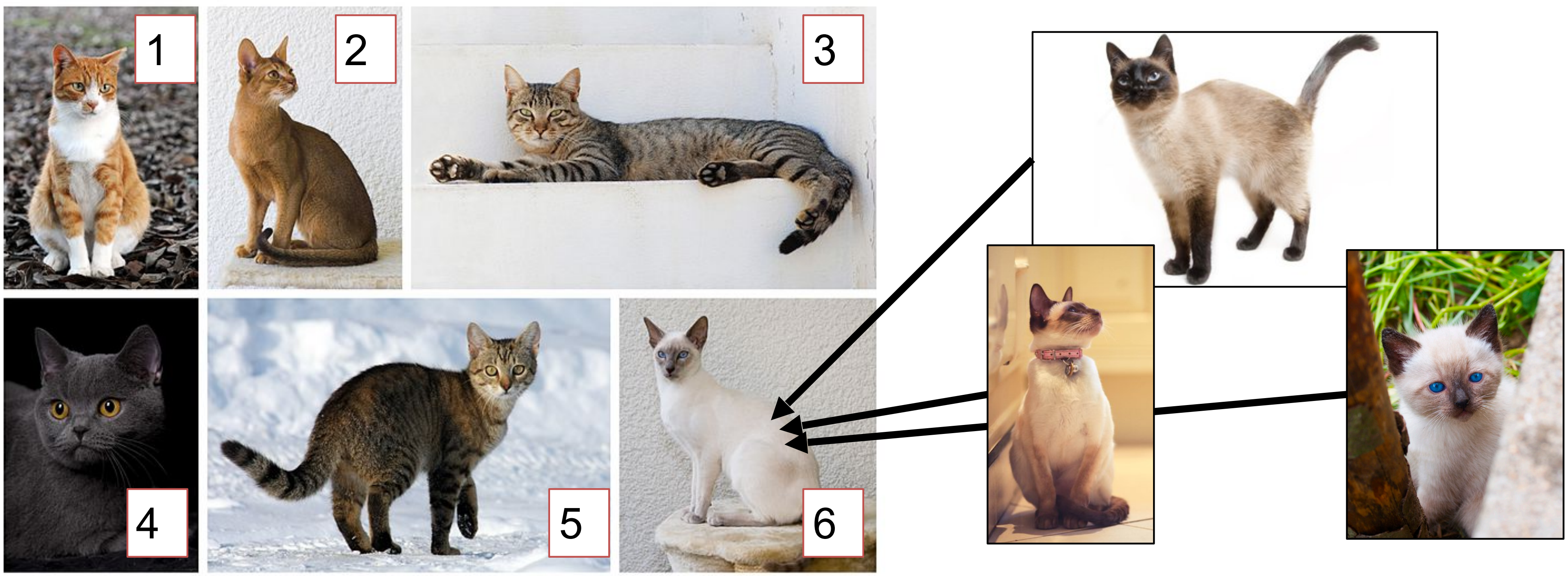
	Supervised (Data+Labels)	**Unsupervised (Just Data)**
Discrete Output	Classification/ Categorization	**Clustering**
Continuous Output	Regression	

 Slide adapted from J. Hays

ML Problem Examples in Vision

Clustering

Given a set of cats, automatically discover clusters or *cat*egories.



Credit: D. Fouhey   Image credit: Wikipedia, cattime.com

ML Problem Examples in Vision

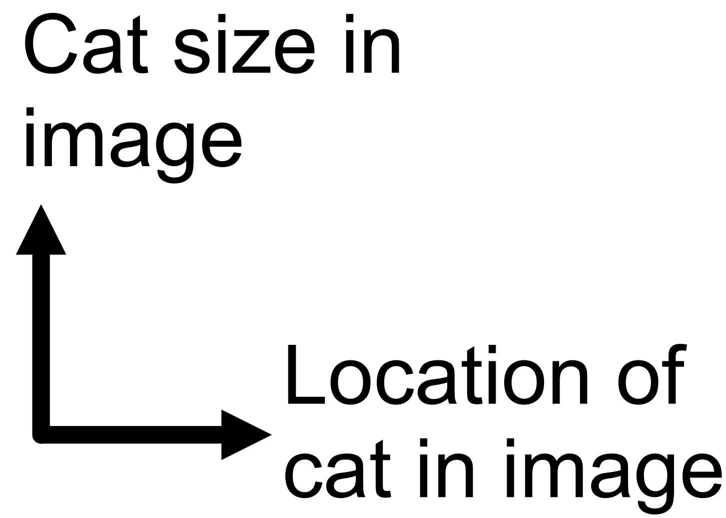
	Supervised (Data+Labels)	Unsupervised (Just Data)
Discrete Output	Classification/ Categorization	Clustering
Continuous Output	Regression	**Dimensionality Reduction**

Slide adapted from J. Hays

ML Problem Examples in Vision

Dimensionality Reduction

Find dimensions that best explain the whole image/input



For ordinary images, this is currently a totally hopeless task. For certain images (e.g., faces, this works reasonably well)

Credit: D. Fouhey Image credit: Wikipedia

Practical Example

- ML has a tendency to be mysterious
- Let's start with:
  - A simple model (a line)
  - A simple fitting method
- One thing to remember:
  - N eqns, <N vars = overdetermined (will have errors)
  - N eqns, N vars = exact solution
  - N eqns, >N vars = underdetermined (infinite solns)

Credit: D. Fouhey

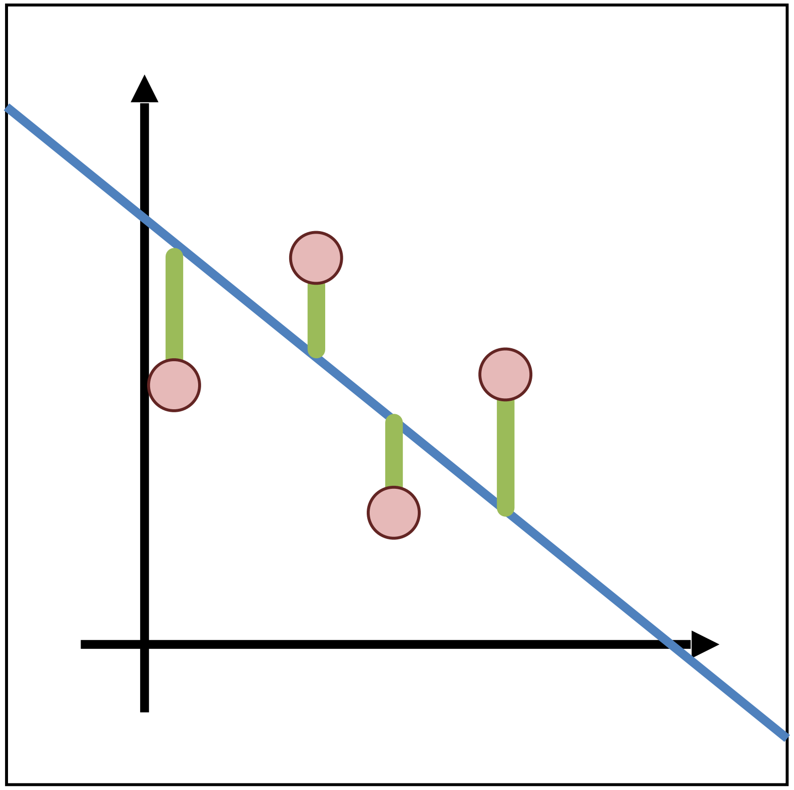
Example – Least Squares

Let's make the world's **worst** weather model

Data: (x_1,y_1), (x_2,y_2), …, (x_k,y_k)

Model: (m,b) $y_i=mx_i+b$
Or $(\mathbf{w})$ $y_i = \mathbf{w}^T\mathbf{x}_i$

Objective function:
$(y_i - \mathbf{w}^T\mathbf{x}_i)^2$

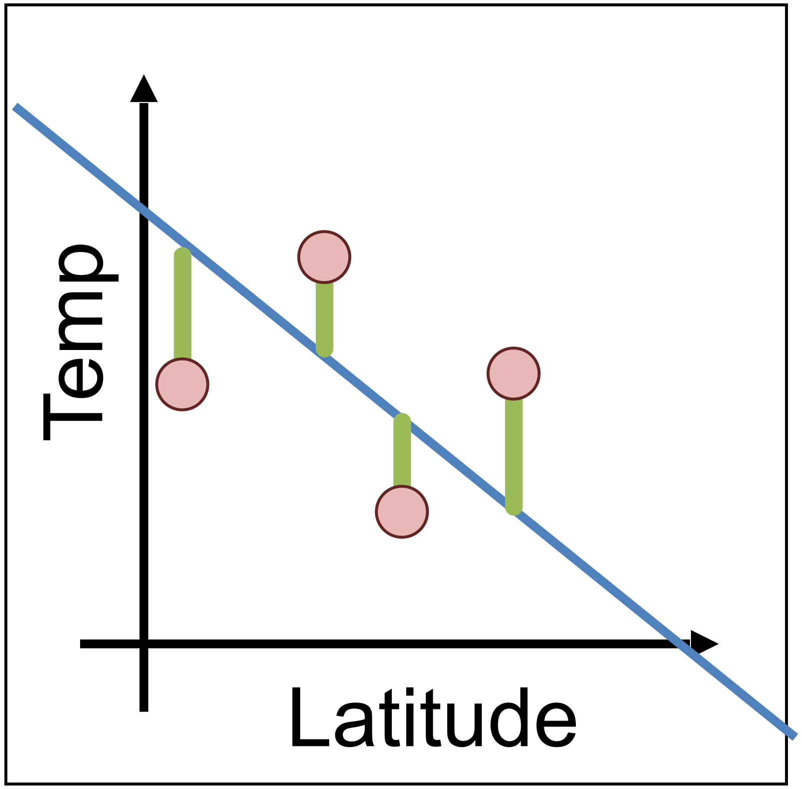


Credit: D. Fouhey

World’s Worst Weather Model

Given latitude (distance above equator), predict temperature in October by fitting a line

City	*Latitude (°)*	*Temp (F)*
Princeton	40	66
Boston, MA	42	62
Austin, TX	30	82
Mexico City	19	75
Vancouver	49	57



Credit: D. Fouhey

Example – Least Squares

$$\sum_{i=1}^{k} \left(y_i - \boldsymbol{w}^T \boldsymbol{x_i}\right)^2 \longrightarrow \left\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\right\|_2^2$$

Output:
Temperature

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_k \end{bmatrix}$$

Inputs:
Latitude, 1

$$\boldsymbol{X} = \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_k & 1 \end{bmatrix}$$

Model/Weights:
Latitude, “Bias”

$$\boldsymbol{w} = \begin{bmatrix} m \\ b \end{bmatrix}$$

Credit: D. Fouhey

Example – Least Squares

$$\sum_{i=1}^{k} \left(y_i - \boldsymbol{w}^T \boldsymbol{x_i}\right)^2 \longrightarrow \left\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\right\|_2^2$$

Output:
Temperature

$$\boldsymbol{y} = \begin{bmatrix} 66 \\ \vdots \\ 57 \end{bmatrix}$$

Inputs:
Latitude, 1

$$\boldsymbol{X} = \begin{bmatrix} 40 & 1 \\ \vdots & \vdots \\ 49 & 1 \end{bmatrix}$$

Model/Weights:
Latitude, “Bias”

$$\boldsymbol{w} = \begin{bmatrix} m \\ b \end{bmatrix}$$

Intuitively why do we add a one to the inputs?

Credit: D. Fouhey

Example – Least Squares

Training ($\mathbf{x}_i$,y_i):

$$\underset{\boldsymbol{w}}{\operatorname{argmin}} \left\| \boldsymbol{y} - \boldsymbol{X}\boldsymbol{w} \right\|_2^2 \quad \textbf{or}$$

$$\underset{\boldsymbol{w}}{\operatorname{argmin}} \sum_{i=1}^{n} \left\| \boldsymbol{w}^T \boldsymbol{x}_{\boldsymbol{i}} - y_i \right\|^2$$

Loss function/objective: evaluates correctness.
Here: Squared L2 norm / Sum of Squared Errors

Training/Learning/Fitting: try to find model that *optimizes*/*minimizes* an objective / loss function

Recall: optimal **w*** is $\boldsymbol{w}^* = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}$

Credit: D. Fouhey

Example – Least Squares

Training ($\mathbf{x}_i$,y_i):

$$\underset{\boldsymbol{w}}{\operatorname{argmin}} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 \quad \textbf{or}$$

$$\underset{\boldsymbol{w}}{\operatorname{argmin}} \sum_{i=1}^{n} \left\|\boldsymbol{w}^T \boldsymbol{x_i} - y_i\right\|^2$$

Inference (x):

$$\boldsymbol{w}^T \boldsymbol{x} = w_1 x_1 + \cdots + w_F x_F$$

Testing/Inference: Given a new output, what's the prediction?

Credit: D. Fouhey

Least Squares: Learning

Data

City	*Latitude*	*Temp*
Princeton	40	66
Boston, MA	42	62
Austin, TX	30	82
Mexico City	19	75
Vancouver	49	57

Model

Temp =
-0.7*Lat + 94

$$\boldsymbol{X}_{5x2} = \begin{bmatrix} 40 & 1 \\ 42 & 1 \\ 30 & 1 \\ 19 & 1 \\ 49 & 1 \end{bmatrix} \quad \boldsymbol{y}_{5x1} = \begin{bmatrix} 66 \\ 62 \\ 82 \\ 75 \\ 57 \end{bmatrix}$$

$$\left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

$$w_{2x1} = \begin{bmatrix} -0.7 \\ 94 \end{bmatrix}$$

Credit: D. Fouhey

Let’s Predict The Weather

The COS429 Weather Channel

City	Latitude	Temp	Temp	Error
Princeton	40	66	65.5	0.5
Boston, MA	42	62	64.1	2.1
Austin, TX	30	82	72.7	9.3
Mexico City	19	75	80.5	5.5
Vancouver	49	57	59.1	2.1

Credit: D. Fouhey

Is This a Minimum Viable Product?

The COS429 Weather Channel

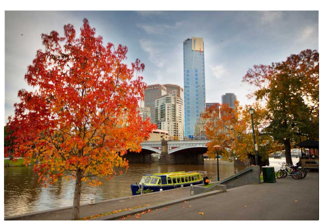
The Weather Channel

Washington, DC:
Temp = -0.7*39 + 94 = 66.7

Actual in DC:
69

Atlanta, GA:
Temp = -0.7*34 + 94 = 70.2

Actual in Atlanta:
74



Melbourne, Australia:
Temp = -0.7*(-38) + 94 = **120.6**

Actual in Melbourne:
68

Won’t do so well in the Australian market…

Credit: D. Fouhey

Where Can This Go Wrong?

Where Can This Go Wrong?

Data

City	*Latitude*	*Temp*
Princeton	40	66
Boston, MA	42	62

Model

$$\text{Temp} = -2.0 * \text{Lat} + 146.0$$

How well can we predict Princeton and Boston and why?

Credit: D. Fouhey

Always Need Separated Testing

Model might be fit data too precisely "*overfitting*"
Remember: #datapoints = #params = perfect fit

Model may only work under some conditions (e.g., trained on northern hemisphere).

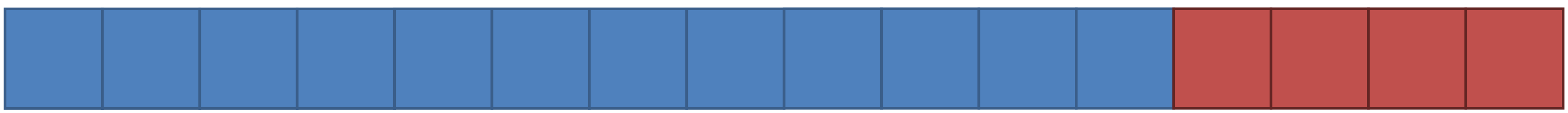
Melbourne:
Temp = -0.7*(-38) + 94 = **121**

Credit: D. Fouhey

Training and Testing

Fit model parameters on **training** set;
evaluate on *entirely unseen* **test** set.

Training Test



Nearly any model can predict data it's seen. If your model can't accurately interpret "unseen" data, it's probably useless. We have no clue whether it has just memorized.

Credit: D. Fouhey

Let's Improve Things

If one feature does ok, what about more features!?

City Name	Latitude (deg)	Avg March High (F)	Avg Snowfall	Oct Temp (F)
Princeton	40	51	24	66
Boston, MA	42	46	48	62
Austin, TX	30	73	0.6	82
Mexico City	19	79	0	75
Vancouver	49	51	17.2	57

$\boldsymbol{X}_{5x4}$ 4 features + a feature of 1s for intercept/bias

$\boldsymbol{y}_{5x1}$

Credit: D. Fouhey

Let's Improve Things

All the math works out!

Data

$\boldsymbol{X}_{5x4}$ $\boldsymbol{y}_{5x1}$

$$\boxed{\boldsymbol{w}^* = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}}$$

Model

$\boldsymbol{w}_{4x1}$

New COS429 Weather Rule:

w_1*latitude + w_2*(avg July high) + w_3*(avg snowfall) + w_4*1

In general called linear regression

Credit: D. Fouhey

Let's Improve Things More

If one feature does ok, what about **LOTS** of features!?

City Name	Latitude (deg)	Avg March High (F)	Avg Snowfall	Month of Year	Elevation (ft)	% Letter P	Oct Temp (F)
Princeton	40	51	24	10	203	100	66
Boston, MA	42	46	48	10	141	3	62
Austin, TX	30	73	0.6	10	489	2	82
Mexico City	19	79	0	10	7382	4	75
Vancouver	49	51	17.2	10	197	1	57

$\boldsymbol{X}_{5x7}$ 6 features + a feature of 1s for intercept/bias

$\boldsymbol{y}_{5x1}$

Credit: D. Fouhey

Let's Improve Things More

Data

$\boldsymbol{X}_{5x7}$ $\boldsymbol{y}_{5x1}$

$$\boxed{\boldsymbol{w}^* = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}}$$

Model

$\boldsymbol{w}_{7x1}$

$$\boldsymbol{w}^* = \underbrace{\left(\boldsymbol{X}^T\boldsymbol{X}\right)}^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

X^TX is a 7x7 matrix but is **rank deficient** (rank 5) *and has no inverse. There are an infinite number of solutions.*

Have to express some preference for which of the infinite solutions we want.

Credit: D. Fouhey

The Fix – Regularized Least Squares

Add **regularization** to objective that prefers some solutions:

Before: $\arg\min_{\boldsymbol{w}} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 \longrightarrow$ Loss

After: $\arg\min_{\boldsymbol{w}} \underbrace{\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2}_{\text{Loss}} + \underbrace{\lambda}_{\text{Trade-off}} \underbrace{\|\boldsymbol{w}\|_2^2}_{\text{Regularization}}$

Want model "smaller": pay a penalty for **w** with big norm

Intuitive Objective: accurate model (low loss) but not too complex (low regularization). λ controls how much of each.

Credit: D. Fouhey

The Fix – Regularized Least Squares

Objective: $$\underset{\boldsymbol{w}}{\operatorname{argmin}} \|\boldsymbol{y}-\boldsymbol{X}\boldsymbol{w}\|_2^2 + \lambda\|\boldsymbol{w}\|_2^2$$

Loss Trade-off Regularization

Take $\frac{\partial}{\partial \boldsymbol{w}}$, set to **0**, solve

$$\boldsymbol{w}^* = \underbrace{\left(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I}\right)}^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

$\mathbf{X^TX+\lambda I}$ is full-rank (and thus invertible) for $\lambda>0$

Called *lots of things:* regularized least-squares, Tikhonov regularization (after Andrey Tikhonov), ridge regression, Bayesian linear regression with a multivariate normal prior.

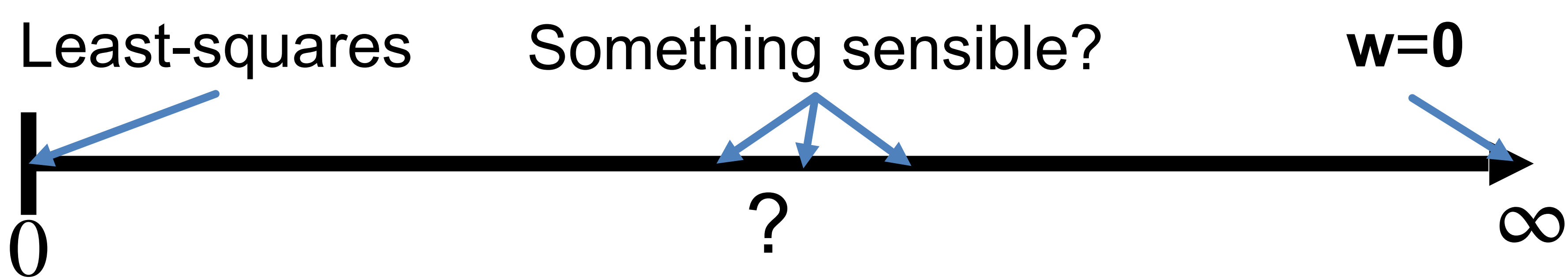
Credit: D. Fouhey

The Fix – Regularized Least Squares

Objective: $$\arg\min_{\boldsymbol{w}} \underbrace{\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2}_{\text{Loss}} + \underbrace{\lambda}_{\text{Trade-off}} \underbrace{\|\boldsymbol{w}\|_2^2}_{\text{Regularization}}$$

What happens (and why) if:

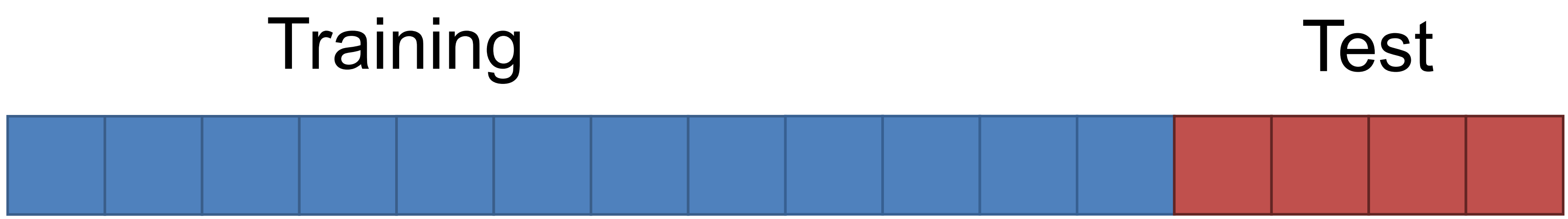
- λ=0
- λ=∞



Credit: D. Fouhey

Training and Testing

Fit model parameters on training set;
evaluate on *entirely unseen* test set.



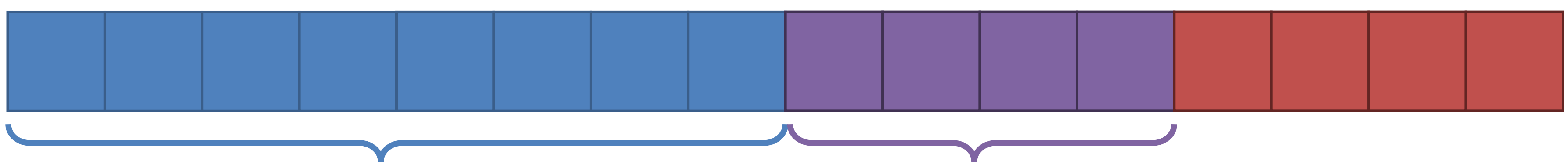
How do we pick λ?

Credit: D. Fouhey

Training and Testing

Fit model parameters on training set;
find *hyperparameters* by testing on validation set;
evaluate on *entirely unseen* test set.

Training Validation Test



Use these data points to fit $\mathbf{w}^*=(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$

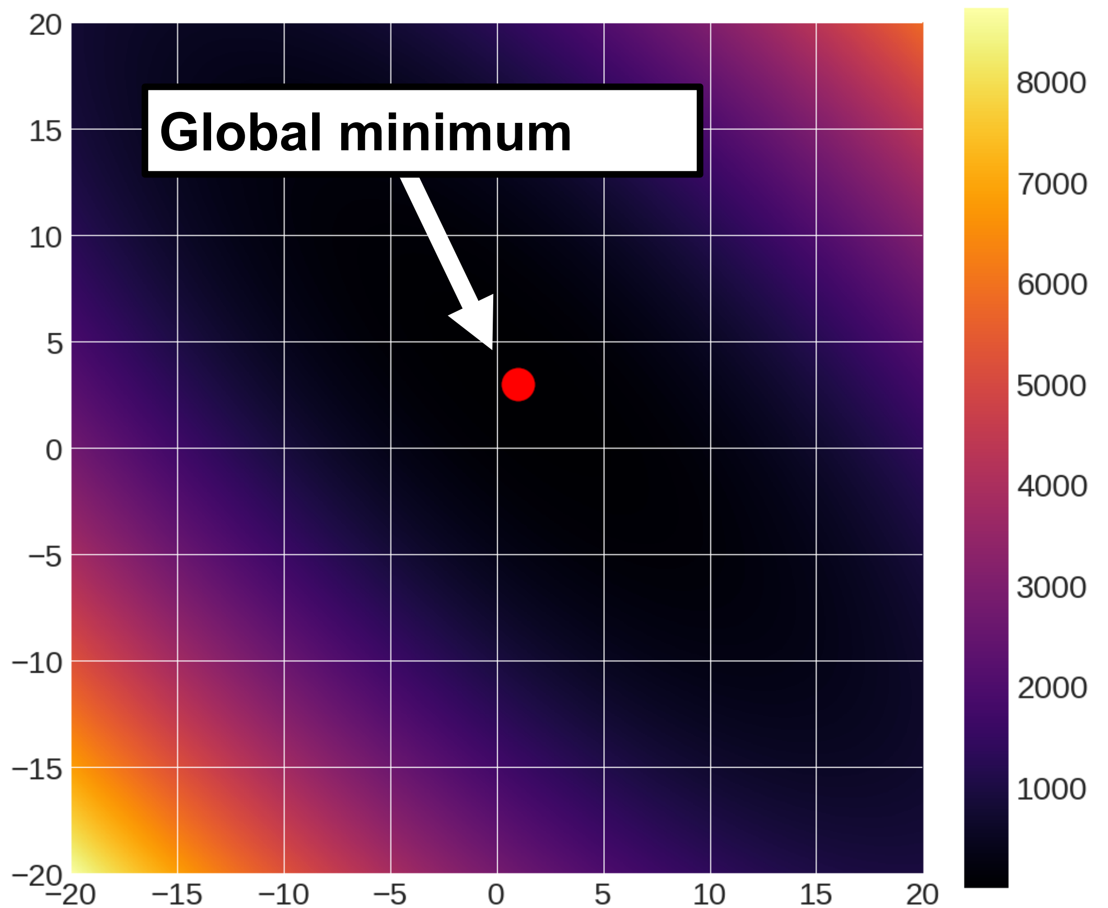
Evaluate on these points for different λ, pick the best

Credit: D. Fouhey

Optimization

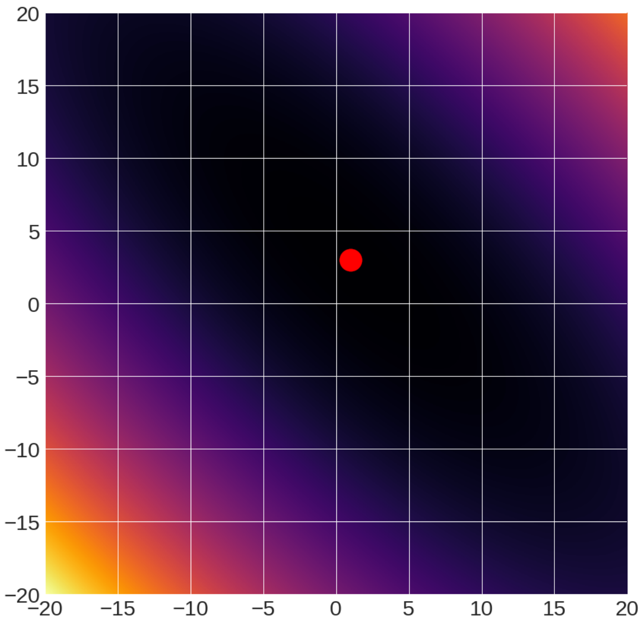
Sample Function to Optimize

$$f(x,y) = (x+2y-7)^2 + (2x+y-5)^2$$



Credit: D. Fouhey

A Caveat

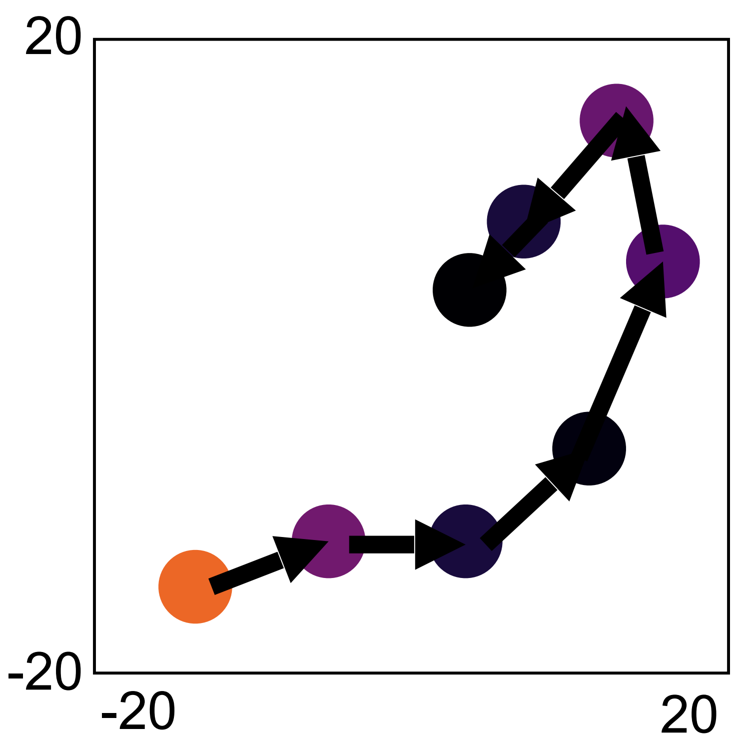


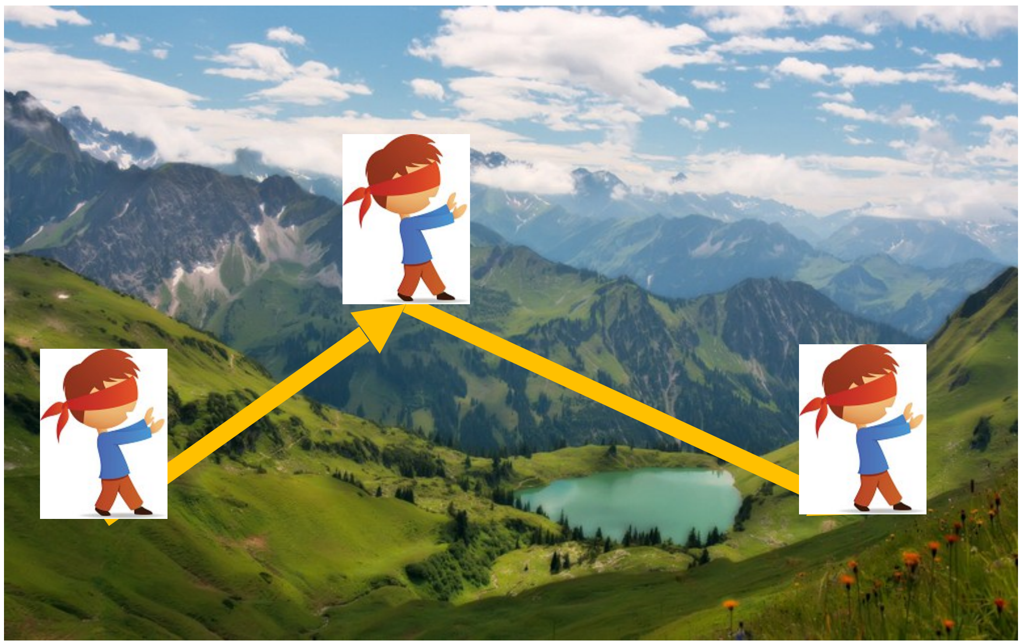
- Each point in the picture is a function evaluation
- Here it takes microseconds – so we can easily see the answer
- Functions we want to optimize may take hours to evaluate

Credit: D. Fouhey

More Caveat

Model in your head: moving around a landscape with a teleportation device





Landscape diagram: Karpathy and Fei-Fei

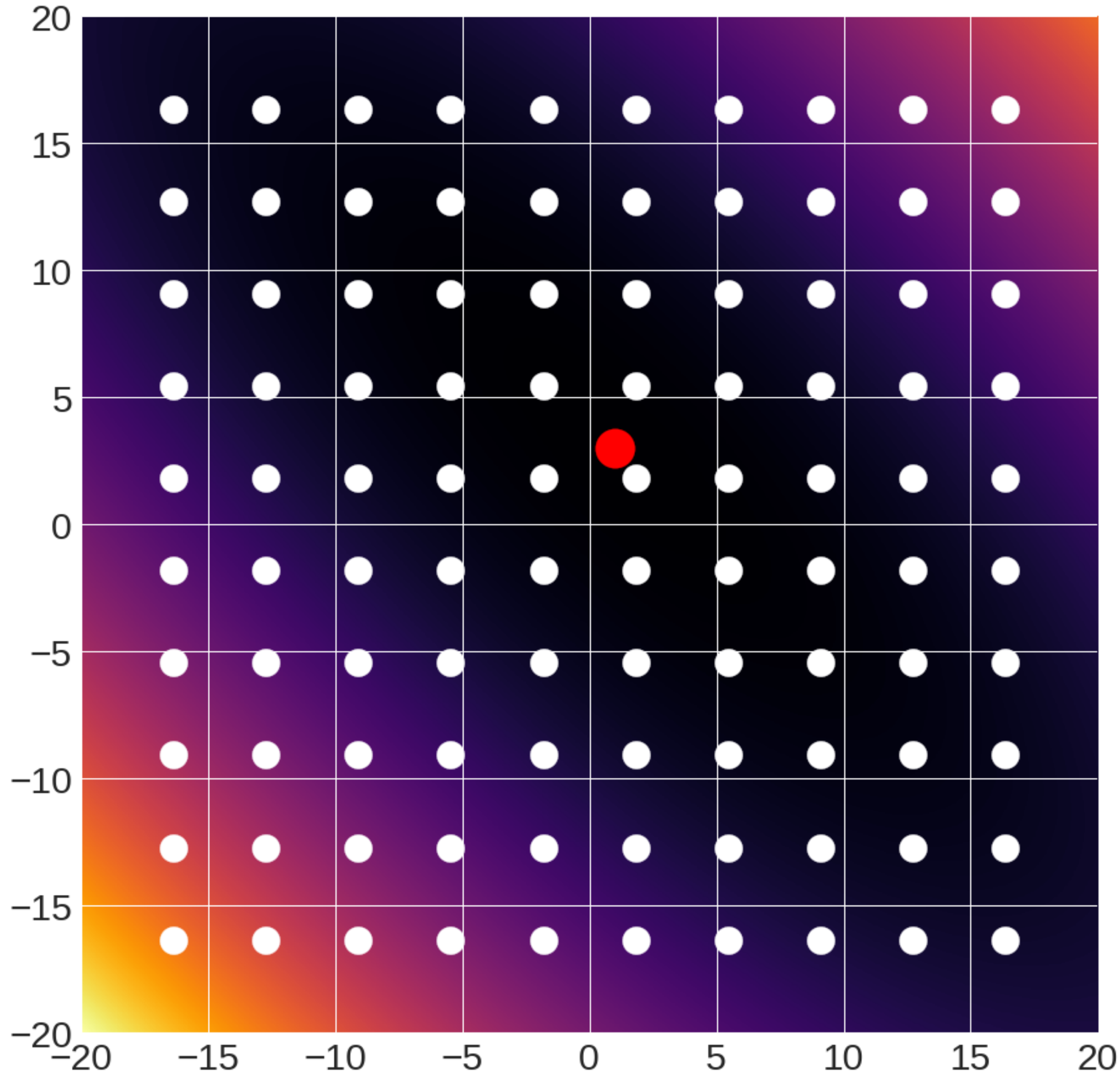
Credit: D. Fouhey

Option #1A – "Grid Search"

```
#systematically try things
best, bestScore = None, Inf
for dim1Value in dim1Values:
        ….
        for dimNValue in dimNValues:
                w = [dim1Value, …, dimNValue]
                if L(w) < bestScore:
                        best, bestScore = w, L(w)
return best
```

Credit: D. Fouhey

Option #1A – "Grid Search"



Credit: D. Fouhey

Option #1A – "Grid Search"

Pros:
1. Super simple
2. Only requires being able to evaluate model

Cons:
1. Scales horribly to high dimensional spaces

Complexity: $\text{samplesPerDim}^{\text{numberOfDims}}$

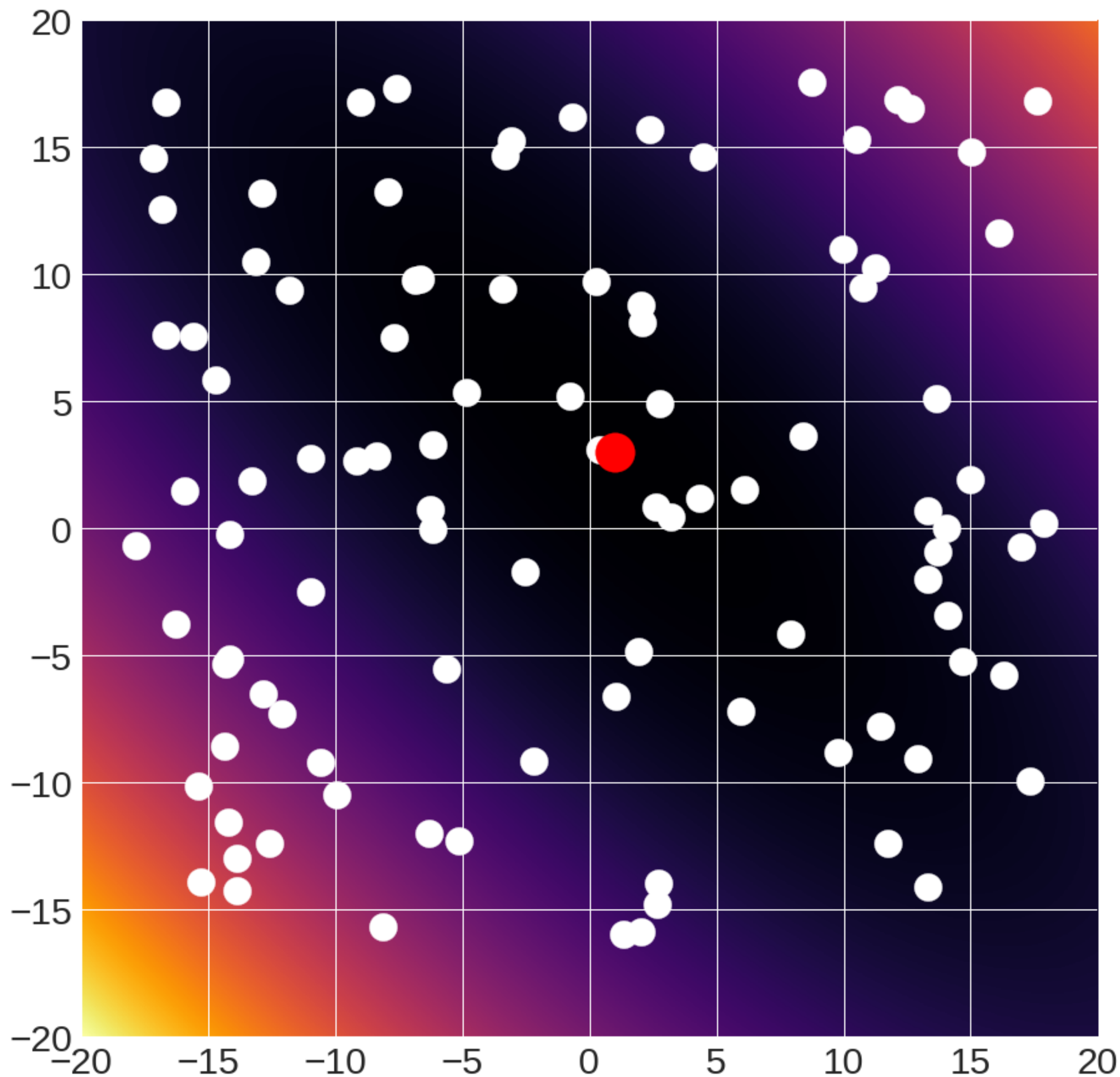
Credit: D. Fouhey

Option #1B – Random Search

```
#Do random stuff RANSAC Style
best, bestScore = None, Inf
for iter in range(numIters):
        w = random(N,1) #sample
        score = L(w) #evaluate
        if score < bestScore:
                best, bestScore = w, score
return best
```

Credit: D. Fouhey

Option #1B – Random Search



Credit: D. Fouhey

Option #1B – Random Search

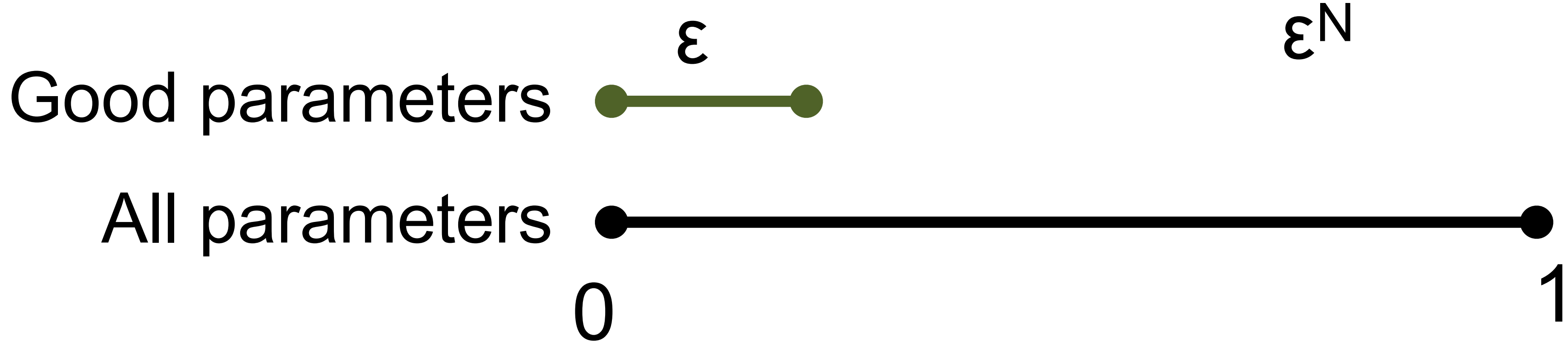
Pros:

1. Super simple
2. Only requires being able to sample model and evaluate it

Cons:

1. Slow and stupid – throwing darts at high dimensional dart board
2. Might miss something

P(all correct) = ε^N



Credit: D. Fouhey

When Do You Use Options 1A/1B?

Use these when

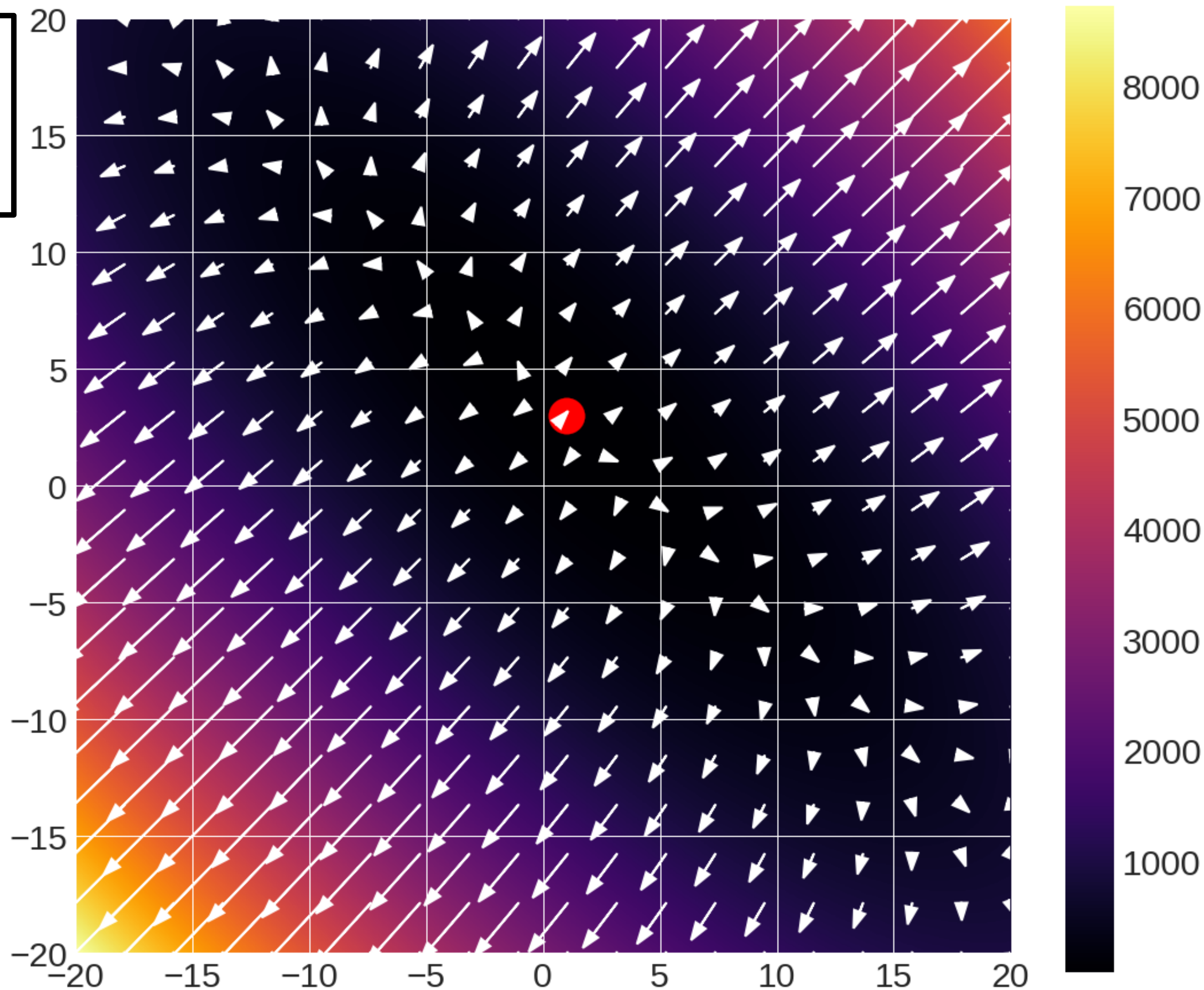
- Number of dimensions small, space bounded
- Objective is impossible to analyze (e.g., test accuracy if we use this distance function)

Random search is arguably more effective; grid search makes it easy to systematically test something (people love certainty)

Credit: D. Fouhey

Option 2 – Use The Gradient

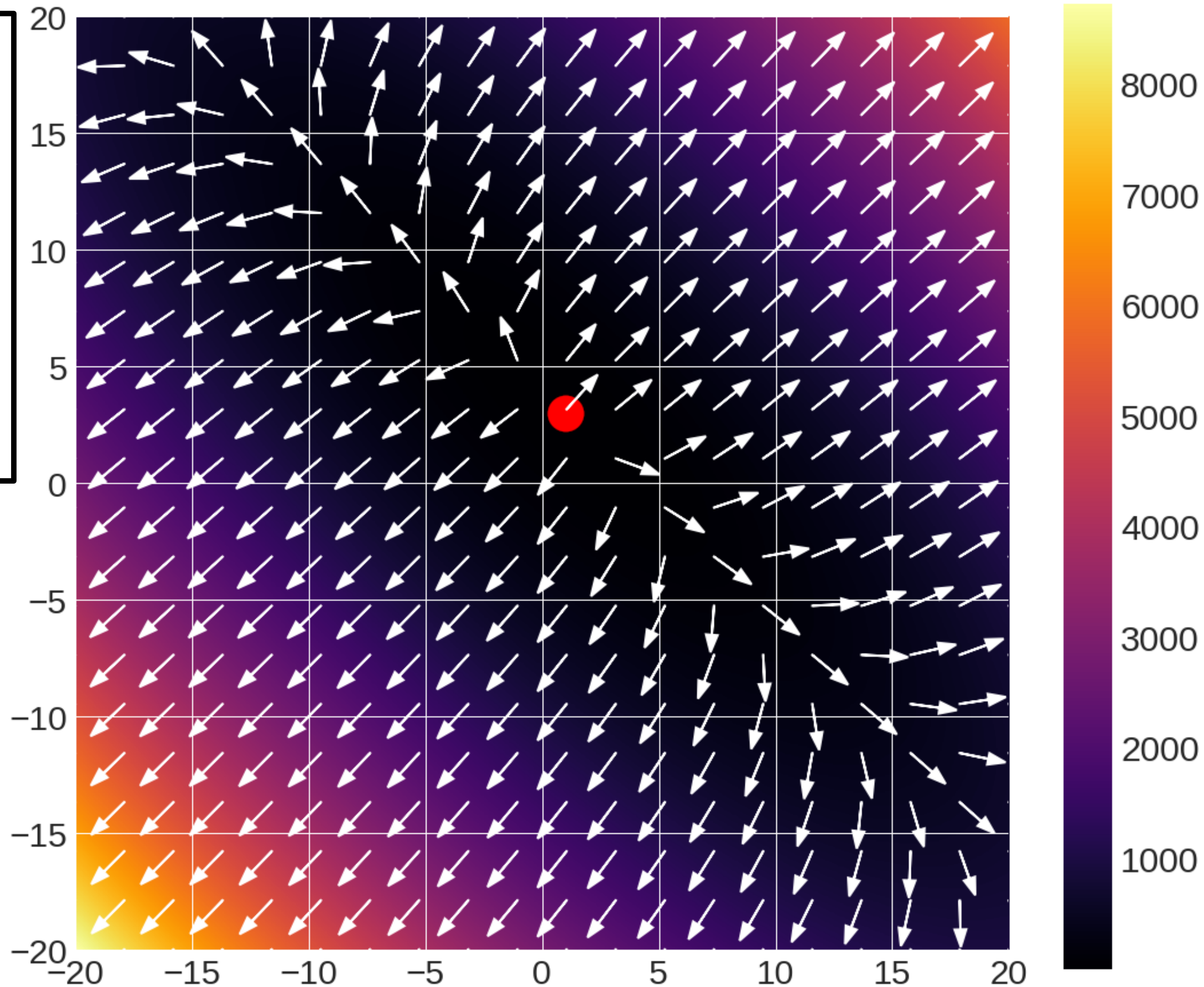
Arrows:
gradient



Credit: D. Fouhey

Option 2 – Use The Gradient

Arrows:
gradient direction
(scaled to unit length)



Credit: D. Fouhey

Option 2 – Use The Gradient

Want: $\arg\min_{\boldsymbol{w}} L(\boldsymbol{w})$

What's the geometric interpretation of:

$$\nabla_{\boldsymbol{w}} L(\boldsymbol{w}) = \begin{bmatrix} \partial L / \partial \boldsymbol{x}_1 \\ \vdots \\ \partial L / \partial \boldsymbol{x}_N \end{bmatrix}$$

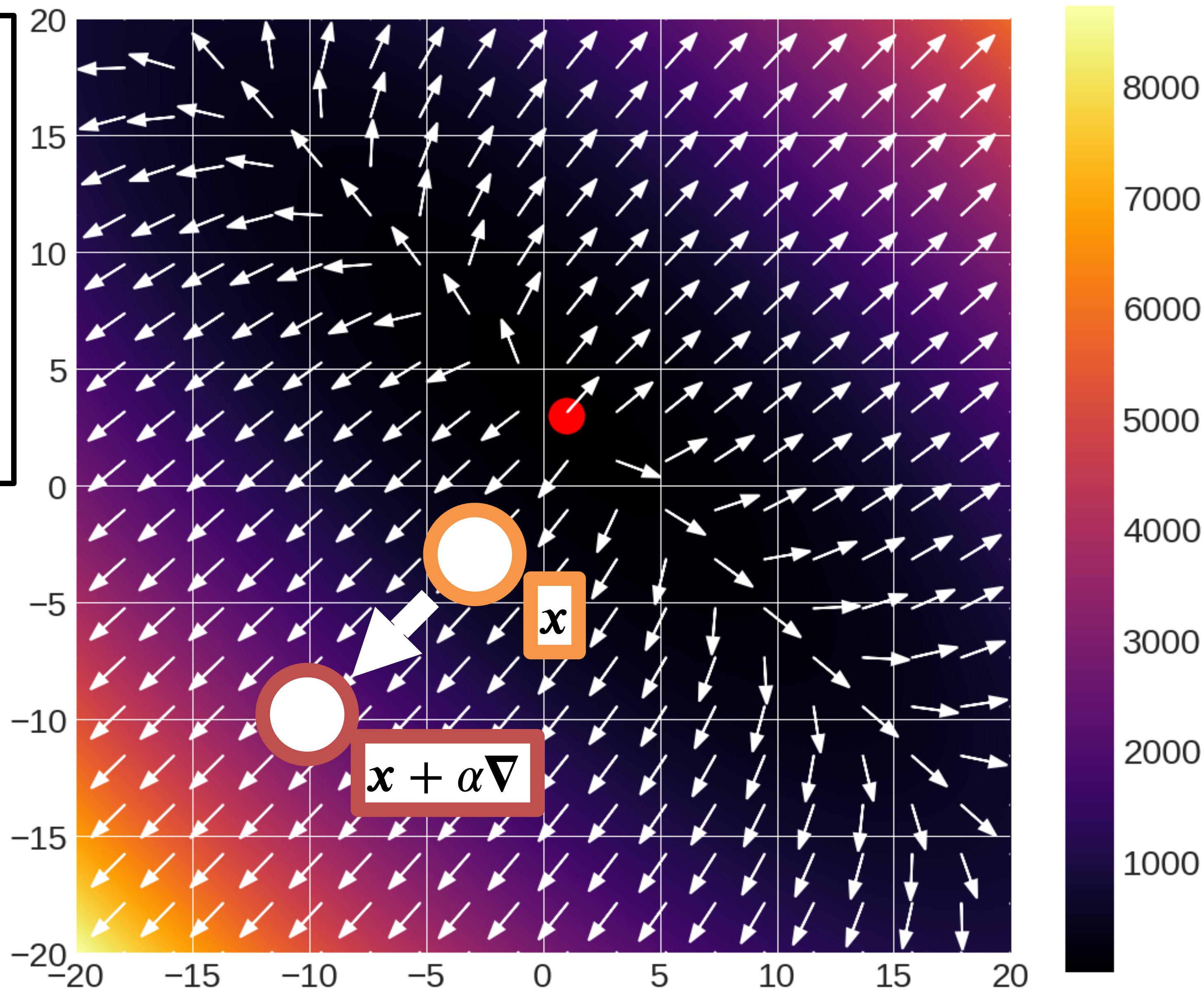
Which is bigger (for small α)?

$$L(\boldsymbol{w}) \quad \begin{matrix} \leq ? \\ \\ > ? \end{matrix} \quad L\big(\boldsymbol{w} + \alpha \nabla_{\boldsymbol{w}} L(\boldsymbol{w})\big)$$

Credit: D. Fouhey

Option 2 – Use The Gradient

Arrows: gradient direction (scaled to unit length)



Credit: D. Fouhey

Option 2 – Use The Gradient

Method: at each step, move in direction of negative gradient

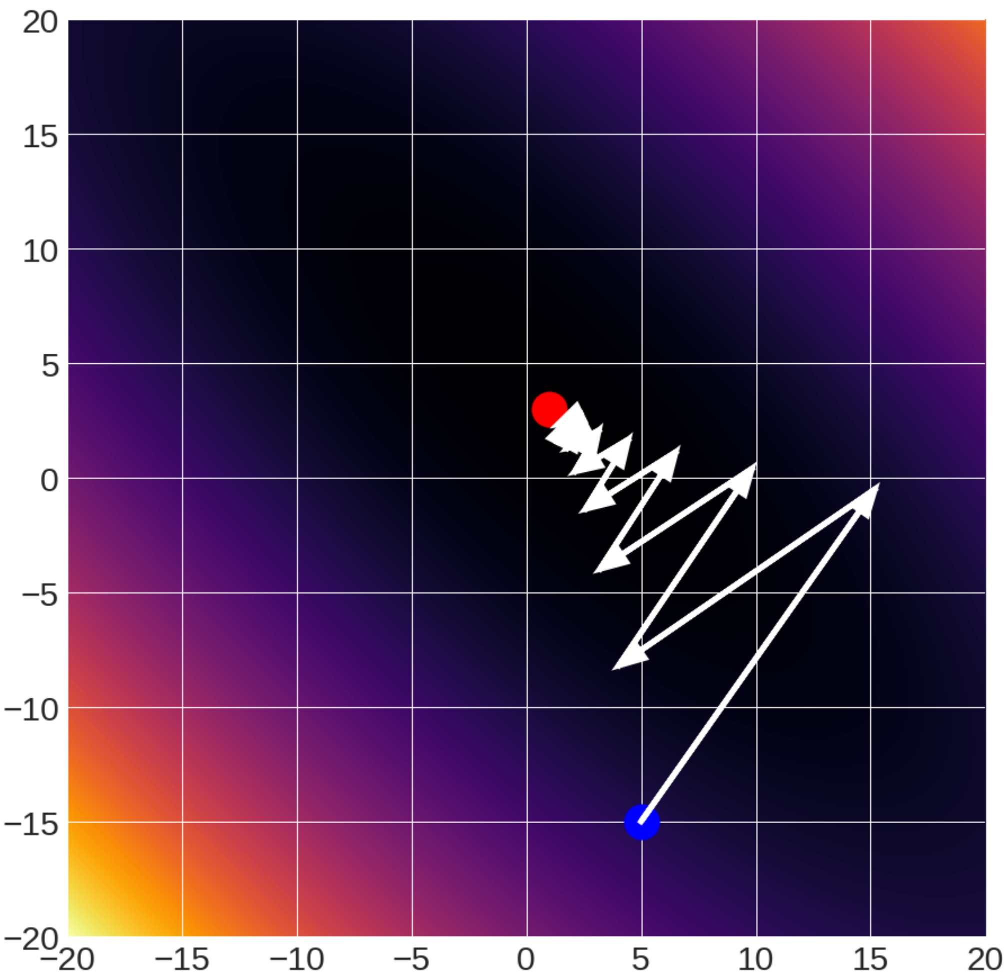
w0 = *initialize*() #initialize
for iter in range(numIters):
    g = $\nabla_{\boldsymbol{w}} L(\boldsymbol{w})$ #eval gradient
    w = **w** + -*stepsize*(iter)***g** #update w
return **w**

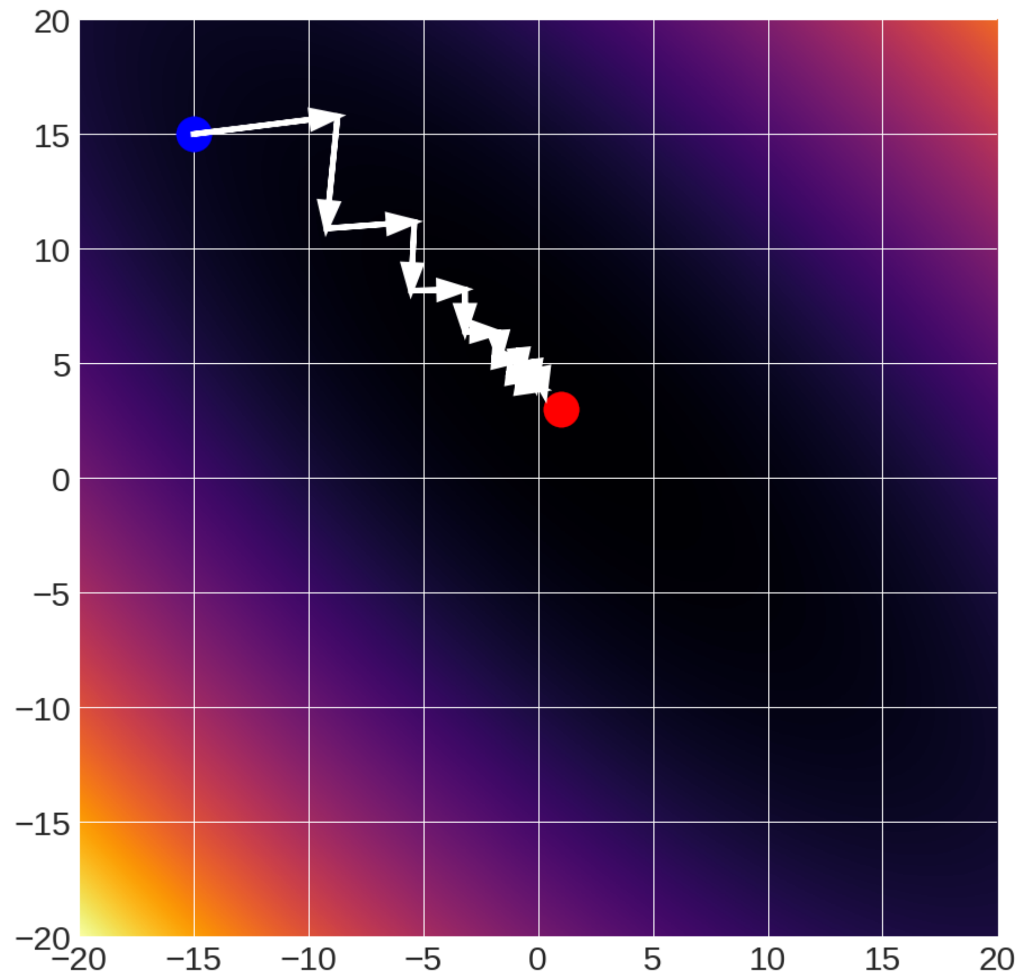
Credit: D. Fouhey

Gradient Descent

Given starting point (blue)

$$w_{i+1} = w_i + -9.8 \times 10^{-2} \text{ x gradient}$$





Credit: D. Fouhey

Computing the Gradient

How Do You Compute The Gradient?
Numerical Method:

$$\nabla_{\boldsymbol{w}} L(\boldsymbol{w}) = \begin{bmatrix} \boxed{\dfrac{\partial L(w)}{\partial x_1}} \\ \vdots \\ \dfrac{\partial L(w)}{\partial x_n} \end{bmatrix}$$

How do you compute this?

$$\frac{\partial f(x)}{\partial x} = \lim_{\epsilon \to 0} \frac{f(x+\epsilon) - f(x)}{\epsilon}$$

In practice, use:

$$\frac{f(x+\epsilon) - f(x-\epsilon)}{2\epsilon}$$

Credit: D. Fouhey

Computing the Gradient

How Do You Compute The Gradient?

Numerical Method:

$$\nabla_{\boldsymbol{w}} L(\boldsymbol{w}) = \begin{bmatrix} \boxed{\dfrac{\partial L(w)}{\partial x_1}} \\ \vdots \\ \dfrac{\partial L(w)}{\partial x_n} \end{bmatrix}$$

Use: $\dfrac{f(x+\epsilon) - f(x-\epsilon)}{2\epsilon}$

How many function evaluations per dimension?

Computing the Gradient

How Do You Compute The Gradient?
Analytical Method:

$$\nabla_{\boldsymbol{w}} L(\boldsymbol{w}) = \begin{bmatrix} \frac{\partial L(w)}{\partial x_1} \\ \vdots \\ \frac{\partial L(w)}{\partial x_n} \end{bmatrix}$$

Use calculus!

Credit: D. Fouhey

Computing the Gradient

$$L(\boldsymbol{w}) = \lambda \|\boldsymbol{w}\|_2^2 + \sum_{i=1}^{n} \left(y_i - \boldsymbol{w}^T \boldsymbol{x}_i\right)^2$$

$$\Big\downarrow \quad \frac{\partial}{\partial \boldsymbol{w}} \quad \Big\downarrow$$

$$\nabla_{\boldsymbol{w}} L(\boldsymbol{w}) = 2\lambda \boldsymbol{w} + \sum_{i=1}^{n} -\left(2\left(y_i - \boldsymbol{w}^T \boldsymbol{x}_i\right) \boldsymbol{x}_i\right)$$

Note: if you look at other derivations, things are written either $(y\text{-}w^T x)$ or $(w^T x - y)$; the gradients will differ by a minus.

Credit: D. Fouhey

Interpreting the Gradient

Recall:

$$\mathbf{w} = \mathbf{w} + -\nabla_{\boldsymbol{w}} L(\boldsymbol{w}) \quad \text{\#update w}$$

$$\nabla_{\boldsymbol{w}} L(\boldsymbol{w}) = 2\lambda \boldsymbol{w} + \sum_{i=1}^{n} -\left(2\left(y_i - \boldsymbol{w}^T \boldsymbol{x_i}\right)\boldsymbol{x}_i\right)$$

$$-\nabla_{\boldsymbol{w}} L(\boldsymbol{w}) = \overbrace{-2\lambda \boldsymbol{w}}^{\text{Push } \mathbf{w} \text{ towards 0}} + \sum_{i=1}^{n} \underbrace{\left(\overbrace{2\left(y_i - \boldsymbol{w}^T \boldsymbol{x_i}\right)}^{\alpha}\boldsymbol{x}_i\right)}$$

If $y_i > w^T x_i$ (too *low*): then $w = w + \alpha x_i$ for some α

Before: $w^T x$

After: $(w + \alpha x)^T x = w^T x + \alpha x^T x$

Computing The Gradient

- Numerical: foolproof but slow
- Analytical: can mess things up ☺️
- In practice: do analytical, but check with numerical (called a gradient check)

Slide: Karpathy and Fei-Fei

Summary of terminology

$$\underset{\boldsymbol{w}}{\operatorname{argmin}} \lambda \|\boldsymbol{w}\|_2^2 + \sum_{i=1}^{n} \left\|\boldsymbol{w}^T \boldsymbol{x}_{\boldsymbol{i}} - y_i\right\|_2^2$$

$\boldsymbol{x}$ Inputs, features, Xs, data

y Outputs, targets, labels, ys

$\boldsymbol{w}$ Weights, weight vector, parameters, params

λ Trade-off parameters, regularization strength

Credit: D. Fouhey

Next time