Midterm next ~~Wednesday~~ Tuesday! ö

- **Time:** Tuesday, 10/28, <u>10:00am sharp</u> to 11:20am
  - Please be seated 5 minutes beforehand.
- **Location:** CSE2 G20 (usual classroom)
- **Topics:** Everything up to and including this week's lectures.
- **Format:** Sample exams posted. <u>More open-ended.</u>
- **Cheat sheet:**
  - You may bring one 8.5x11 inch sheet of paper (can use front & back).
  - It may be handwritten or typeset.

Bring ID card!

Gradient Descent (continued^2)

CSE 446/546

Sewoong Oh & Pang Wei Koh

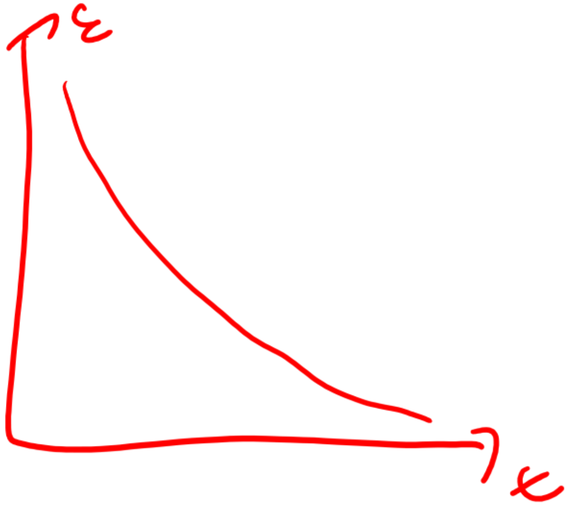
Convergence rate of gradient descent

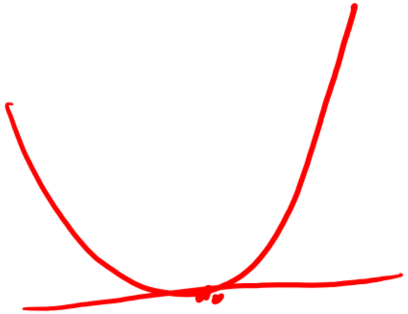
Lipschitz

initial weights

optimal weights

$$T \geq \frac{2L\big(f(w_0) - f(w^*)\big)}{\epsilon}$$





Gradient descent requires

$T = O(1/\epsilon)$ iterations

to achieve $\|\nabla f(w_t)\|^2 \leq \epsilon$

0.01

0.005

Convergence rate of gradient descent

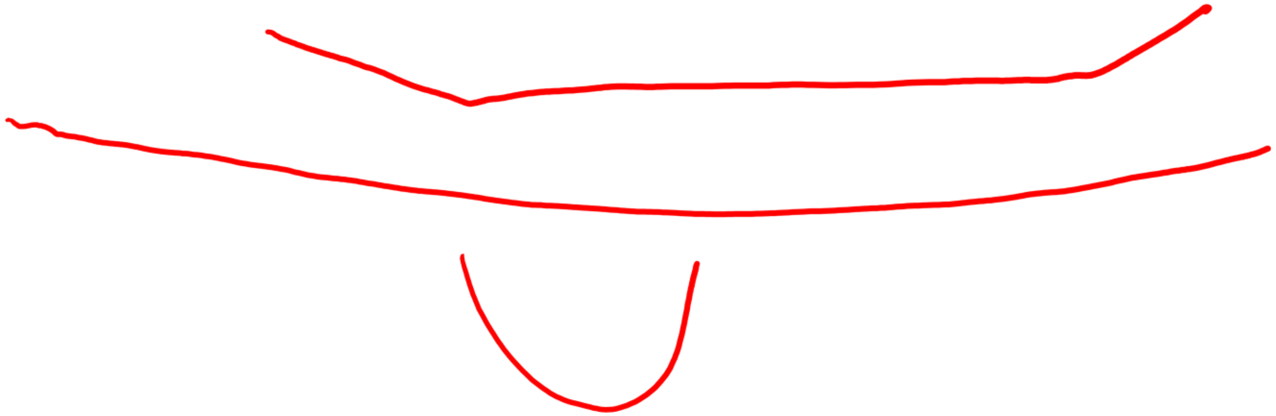
$\|\nabla f(w_t)\| \leq \epsilon$

$1/\varepsilon$

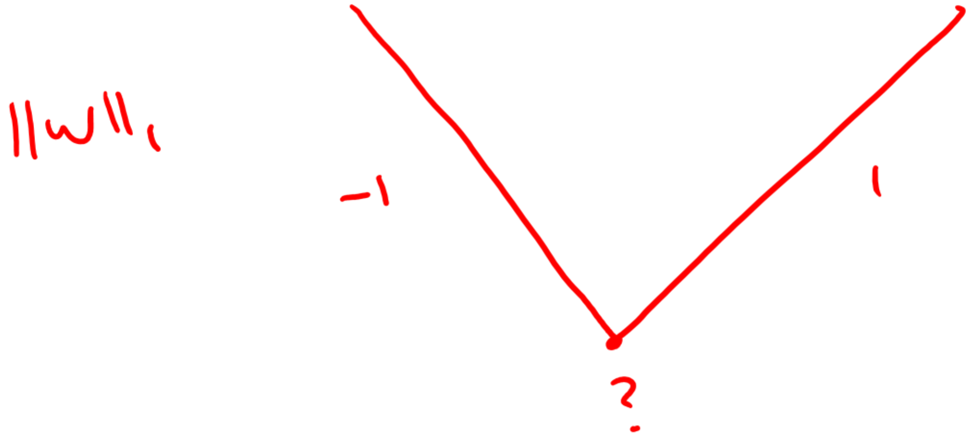
$f(w_t) - f(w^*) \leq \epsilon$

convex: $1/\varepsilon$

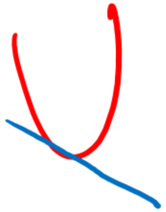
strongly convex: $\log 1/\varepsilon$

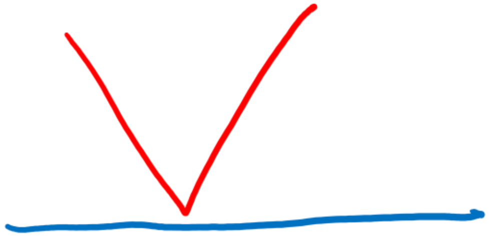


Lasso revisited



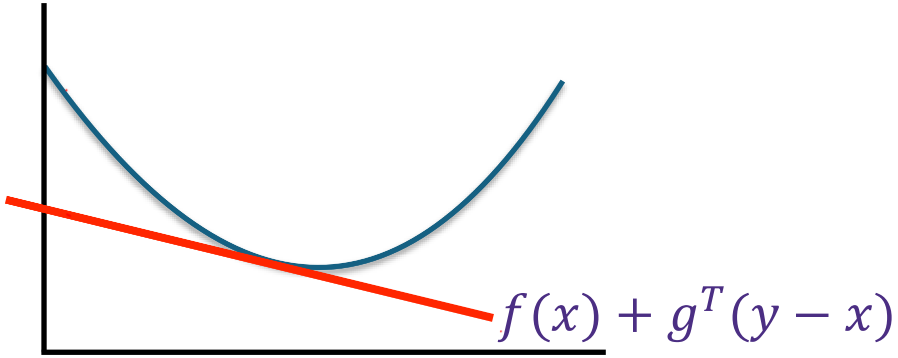
Subgradients





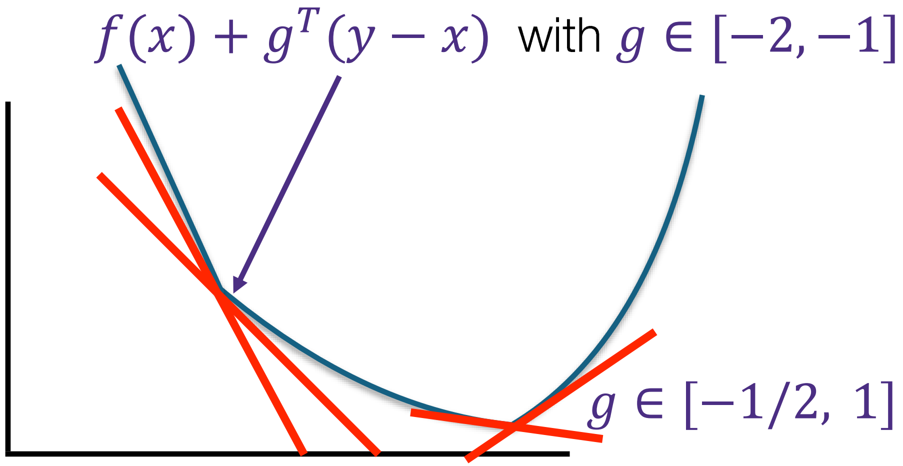
A vector $g \in \mathbb{R}^d$ is a **subgradient** at x if it satisfies
$f(y) \geq f(x) + g^T(y - x)$ for all $y \in \mathbb{R}^d$

Smooth convex function



Gradient is unique sub-gradient
Minimum at points where gradient is 0

Non-smooth convex function

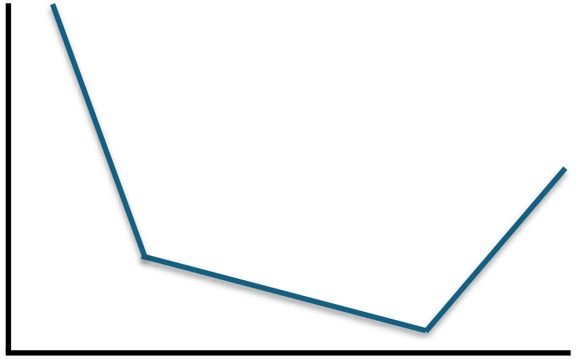


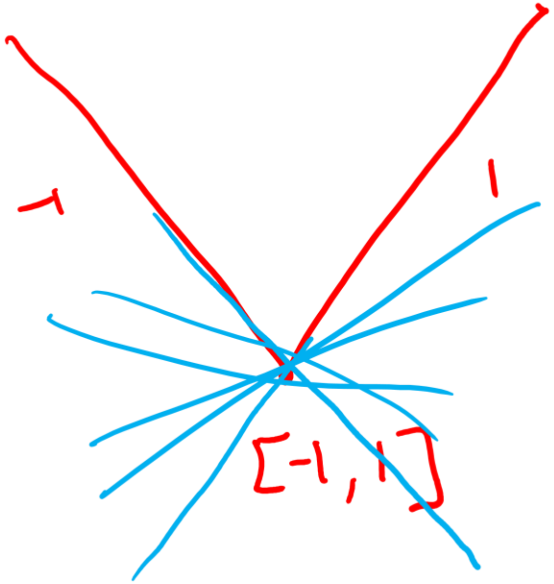
Minimum achieved at points where subgradient set includes 0 vector

Subgradient descent for non-smooth functions

For each t,

  Find any subgradient g_t, then set $w_{t+1} \leftarrow w_t - \eta_t g_t$





Works on non-smooth convex functions
Slower compared to smooth convex functions
Gradients don't get smaller near global minima
- Instead of last iterate w_t, keep track of best one
- Step size needs to decrease with t

Stochastic gradient descent (SGD)

$\ell_i(w)$ could be $\frac{1}{2}\|y_i - x_i^T w\|_2^2$

$$\widehat{w} = \arg\min_{w} \underbrace{\frac{1}{n}\sum_{i=1}^{n} \ell_i(w)}_{f(w)} + R(w)$$

Gradient descent: $w_{t+1} = w_t - \eta \nabla_w \overbrace{\left(\frac{1}{n}\sum_{i=1}^{n} \ell_i(w)\right)}^{\text{need to compute } n \text{ gradients}} |_{w=w_t}$

full batch gradient descent

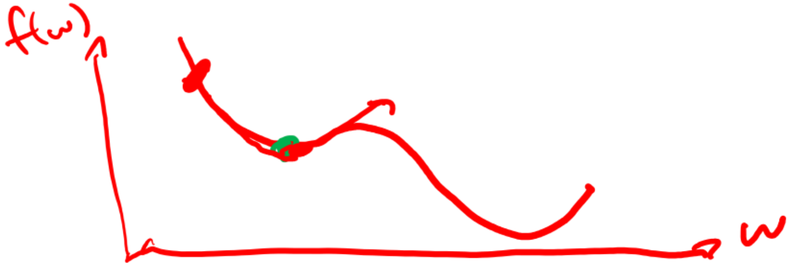
Stochastic gradient descent: $w_{t+1} = w_t - \underline{\eta \nabla_w \ell_{I_t}(w)|_{w=w_t}} - \nabla_w R(w)$

I_t drawn uniformly at random from $\{1, \ldots, n\}$

$\mathbb{E}[\nabla_w \ell_{I_t}(w)] = \frac{1}{n} \nabla_w \sum_{i=1}^{n} \ell_i(w)$

n times faster per iteration!

And can even be better minimizer.



Minibatch stochastic gradient descent

- Instead of one iterate, average B stochastic gradients together

  batch size
- Advantages:
  - Smaller variance (by $1/B$)
  - Parallelization: Each gradient in the minibatch can be computed in parallel

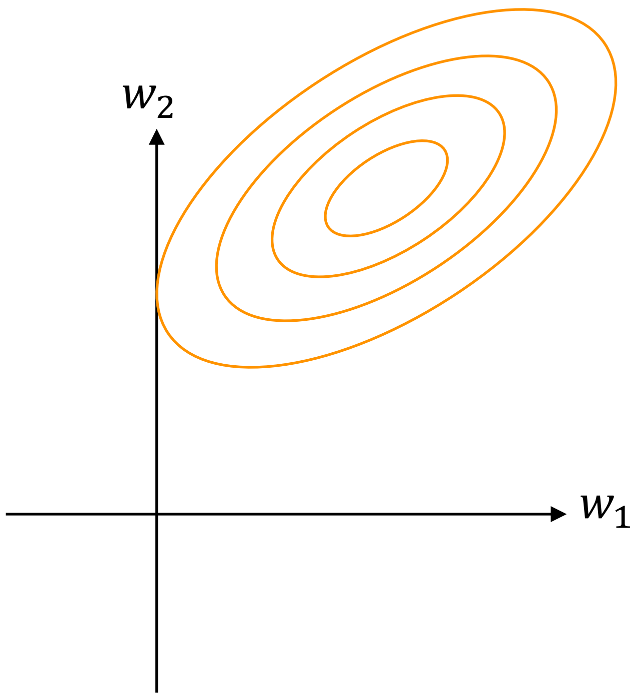
- This is very widely used!

Summary

$(X^TX)^{-1}X^TY$

- Closed form -> iterative methods
- (Minibatch stochastic) gradient descent as a general-purpose optimizer
- Key theoretical tool: Convexity
- Many many variants. Highly active research area!
  - Schedulers
  - Adaptive step sizes
  - Momentum
  - Higher-order methods
  - Non-convex analysis
  - …

Bonus: Coordinate descent



Coordinate descent for lasso

$$\hat{w} = \underset{w}{\operatorname{argmin}} \ \frac{1}{2}\sum_{i=1}^{n}\left(y_i - x_i^\top w\right)^2 + \lambda\|w\|_1$$

$$\frac{d}{dw_k} f(w)$$
$$= \sum_{i=1}^{n}\left(x_i^\top w - y_i\right)x_{ik} + \lambda \operatorname{sign}(w_k)$$
$$= \sum_{i=1}^{n}\left(\sum_{j\neq k} x_{ij}w_j + x_{ik}w_k - y_i\right)x_{ik} + \lambda \operatorname{sign}(w_k)$$
$$= \sum_{i=1}^{n}\left(\sum_{j\neq k} x_{ij}w_j - y_i\right)x_{ik} + w_k \sum_{i=1}^{n} x_{ik} + \lambda \operatorname{sign}(w_k) \ni 0$$
...

Further reading

- Example gradient descent code on class website
- Boyd and Vandenberghe, Convex Optimization
  https://stanford.edu/~boyd/cvxbook/
- Mark Schmidt's CPSC 540 notes:
  https://www.cs.ubc.ca/~schmidtm/Courses/540-W18/L4.pdf
- 3Blue1Brown

Classification

CSE 446/546

Sewoong Oh & Pang Wei Koh

Regression vs classification

- Regression: Given input x, predict continuous value $y \in \mathbb{R}$
- Classification: Given input x, predict discrete value $y \in \mathcal{Y}$

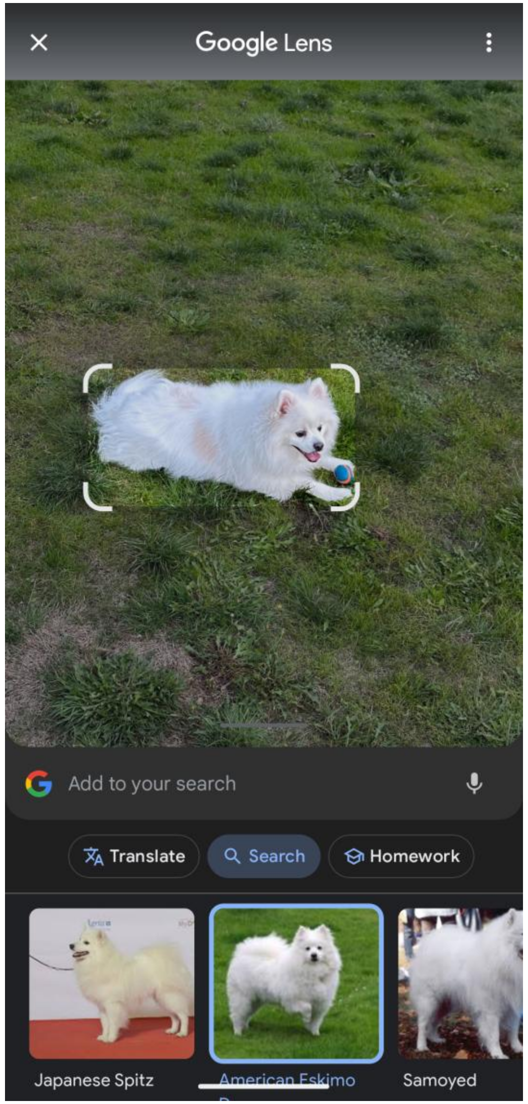
x $\qquad\qquad$ y

Temperature: 62F

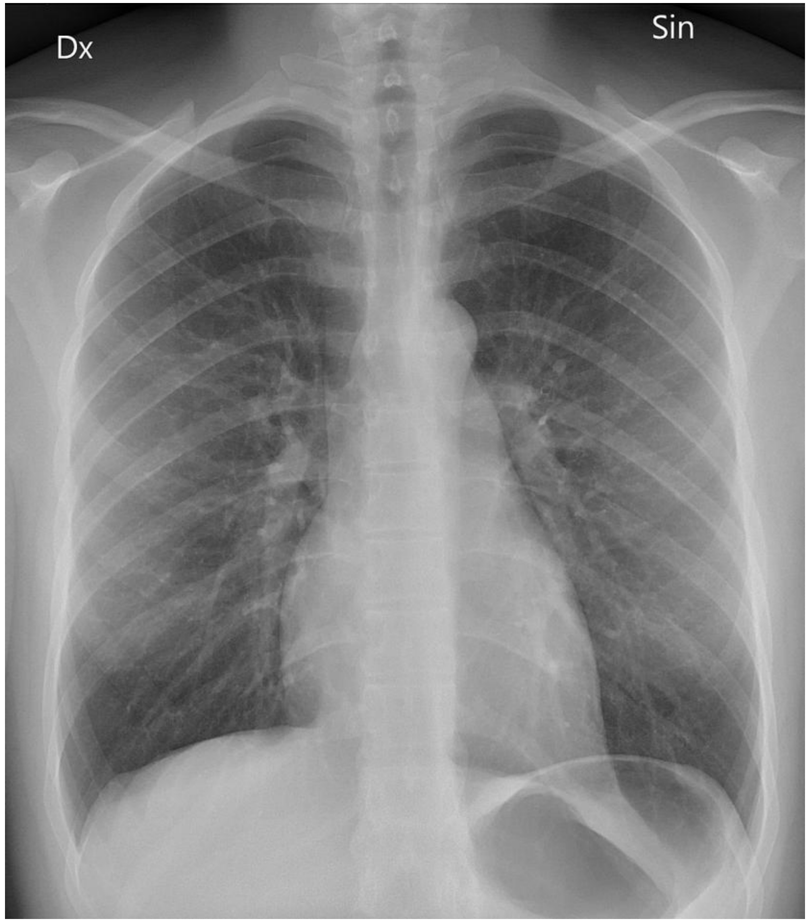
Classification problems are ubiquitous

x

y



Classification problems are ubiquitous



x

Healthy
0/1 Pneumonia
0/1 Pneumothorax
0/1 Pleural Effusion
...

y

Text generation as classification

x

Enter text:

One, two,

One , two ,

3198 11 734 11

3198 11 734 11

predict next token

Prediction

#	probs	next token ID	predicted next token
0	39.71%	1115	three
1	16.97%	290	and
2	7.55%	734	two
3	3.76%	1440	four
4	2.76%	393	or
5	2.18%	1936	five
6	1.57%	530	one
7	1.43%	345	you
8	1.15%	257	a
9	0.84%	3598	seven

https://huggingface.co/blog/alonsosilva/nexttokenprediction

Text generation as classification

Enter text:

One, two, three

One 3198 | , 11 | two 734 | , 11 | three 1115

3198 11 734 11 1115

Prediction

#	probs	next token ID	predicted next token
0	54.42%	11	,
1	5.45%	1399	
2	4.82%	13	.
3	4.51%	290	and
4	2.72%	986	...
5	2.51%	25	:
6	1.50%	393	or
7	1.23%	3926	…
8	0.85%	553	,"
9	0.84%	960	—

Text generation as classification

Enter text:

One, two, three,

One 3198 | , 11 | two 734 | , 11 | three 1115 | , 11

3198 11 734 11 1115 11

Prediction

#	probs	next token ID	predicted next token
0	46.44%	1440	four
1	7.48%	290	and
2	7.31%	1936	five
3	2.66%	393	or
4	2.54%	2237	six
5	2.09%	1115	three
6	1.86%	3863	maybe
7	1.62%	345	you
8	1.23%	257	a
9	0.92%	530	one

https://huggingface.co/blog/alonsosilva/nexttokenprediction

Text generation as classification

Enter text:

One, two, three, four

One , two , three , four

3198 11 734 11 1115 11 1440

Prediction

#	probs	next token ID	predicted next token
0	50.14%	11	,
1	6.66%	13	.
2	5.91%	1399	...
3	3.15%	25	:
4	2.63%	290	and
5	2.58%	986	...
6	1.42%	3926	...
7	1.17%	553	,"
8	1.09%	960	—
9	1.08%	526	."

https://huggingface.co/blog/alonsosilva/nexttokenprediction

Classification

- Learn $f: \mathcal{X} \to \mathcal{Y}$
  - $\mathcal{X} \subset \mathbb{R}^d$: features
  - $\mathcal{Y} = \{1, \dots, k\}$: target classes

$$\boxed{f^*(x) = \operatorname*{argmax}_y P(Y=y \mid X=x)}$$

Bayes-optimal classifier

- 0-1 loss function: $\ell(\underbrace{f(x)}_{\text{pred}}, \underbrace{y}_{\text{true}}) = \mathbf{1}\{f(x) \neq y\} = \begin{cases} 1 & \text{if } f(x) \neq y \\ 0 & \text{otherwise} \end{cases}$
- Expected loss:

$$\mathbb{E}_{X,Y}\left[1\{f(x) \neq y\}\right] = \mathbb{E}_X\, \mathbb{E}_{Y|X}\left[1\{f(x) \neq y\} \mid X=x\right]$$

$$= \mathbb{E}_X\left[P(f(x) \neq y \mid X=x)\right]$$

$$= \mathbb{E}_X\left[1 - P(f(x) = y \mid X=x)\right]$$

Bayes-optimal classifier

$x=1$, $y=0$ | $x=1$, $y=0$ | $x=1$, $y=1$ | $x=2$, $y=1$

$$f^*(x) = \text{argmax}_y\, P(Y=y|X=x)$$

In practice, we don't know $P(Y=y|X=x)$, but have n i.i.d. examples:

$$\{(x_i, y_i)\}_{i=1}^n \quad \text{training data}$$

Suppose $\mathcal{X}$ is discrete so that $x \in \{1, 2, \dots, m\}$. What is a natural estimator for $P(Y=y|X=x)$?

$$\hat{f}(x) = \text{argmax}_y \frac{\sum_{i=1}^n 1\{x_i = x, y_i = y\}}{\sum_{i=1}^n 1\{x_i = x\}}$$

If $\mathcal{X}$ is continuous, we need a <u>model</u> to explain observations!

Maximum likelihood estimation for classification

$$f^*(x) = \text{argmax}_y \, P(Y = y | X = x)$$
$$\hat{f}_w(x) = \text{argmax}_y \, P_w(Y = y | X = x)$$

General MLE procedure:

1. Parameterize $P_w(Y = y|X = x)$ as a function of w
2. Learn w on i.i.d. training data $\{(x_i, y_i)\}_{i=1}^n$

$$\widehat{w}_{\text{MLE}} = \text{argmax}_w \prod_{i=1}^{n} P_w(y_i | x_i)$$
$$= \text{argmax}_w \sum_{i=1}^{n} \log P_w(y_i | x_i)$$

Modeling conditional probabilities

$$\widehat{w}_{\mathrm{MLE}} = \mathrm{argmax}_w \sum_{i=1}^{n} \log P_w(y_i | x_i)$$

- Recall linear regression: $P_w(Y = y | X = x) = \frac{1}{\sqrt{2\pi}} e^{(y - w^\top x)^2/2}$
  - Prediction: $E[Y | X = x] = w^\top x$

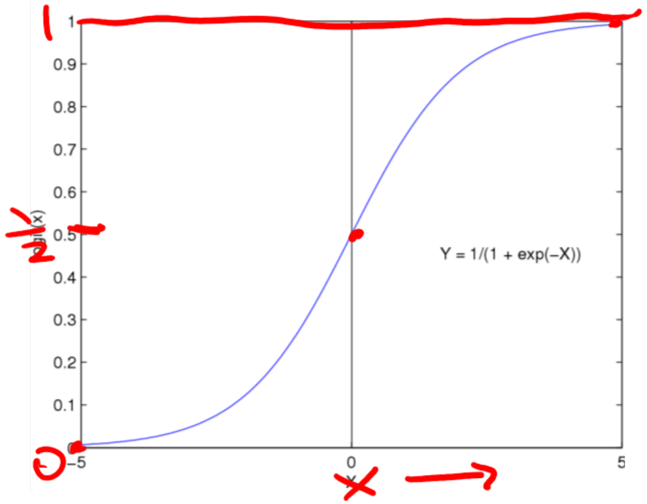
μ

$y = 0$ or 1

- Logistic regression uses a model specialized for (binary) classification:

$$P(Y = 1 | X = x) = \frac{1}{1 + e^{-w^\top x}}$$

$$P(Y = 0 | X = x) = \frac{e^{-w^\top x}}{1 + e^{-w^\top x}}$$

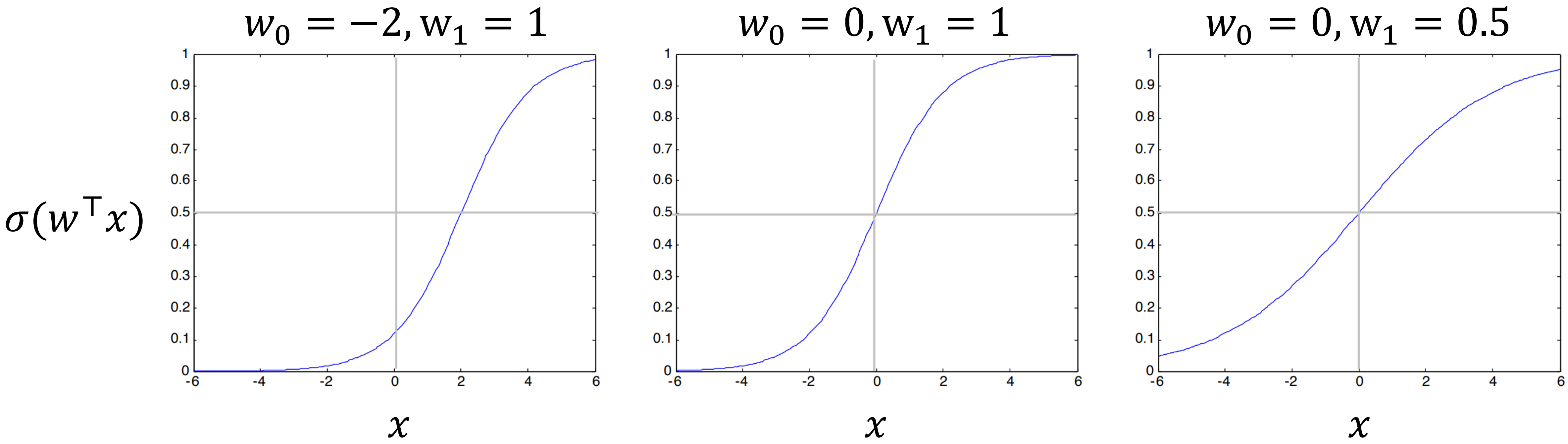


sigmoid curve

Understanding the sigmoid

$$P(Y=1|X=x) = \frac{1}{1+e^{-w^\top x}} = \sigma(w^\top x)$$

$$\sigma\left(w_0 + \sum_k w_k x_k\right) = \frac{1}{1+e^{-w_0+\sum_k w_k x_k}}$$



Sigmoid for binary classification

- What's the shape of the decision rule $P(Y = 1|X) \geq P(Y = 0|X)$?

$$P(Y=1|X) = \frac{1}{1+e^{-w^Tx}}$$

$$P(Y=0|X) = \frac{e^{-w^Tx}}{1+e^{-w^Tx}}$$

Predict $y=1$ if $P(Y=1|X) \geq P(Y=0|X)$

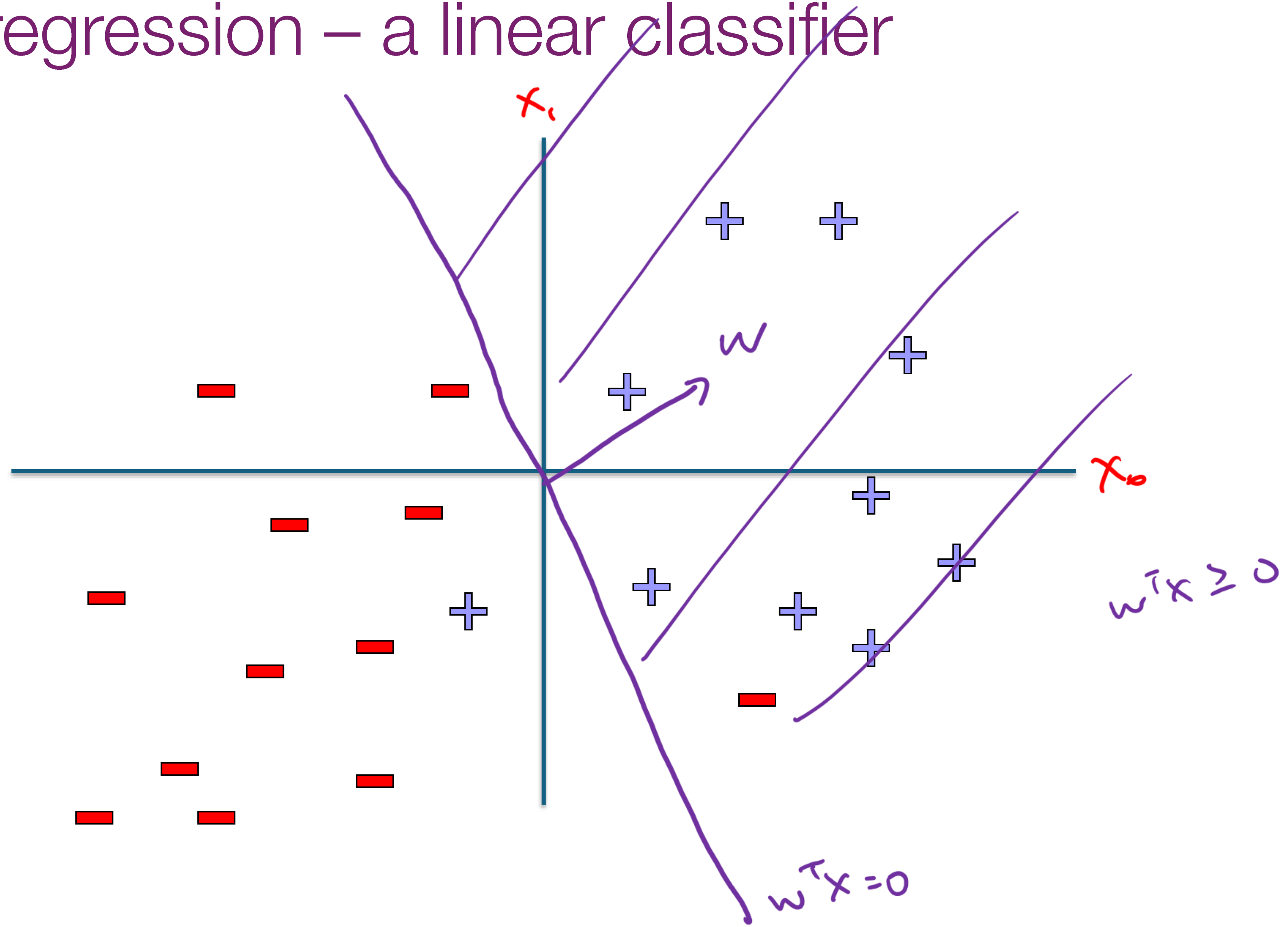
$$\frac{1}{\cancel{1+e^{-w^Tx}}} \geq \frac{e^{-w^Tx}}{\cancel{1+e^{-w^Tx}}}$$

$$1 \geq e^{-w^Tx}$$

$$0 \geq -w^Tx$$

$$w^Tx \geq 0$$

Logistic regression – a linear classifier



Logistic regression for binary classification

$$\widehat{w}_{\mathrm{MLE}} = \mathrm{argmax}_w \sum_{i=1}^{n} \log P_w(y_i|x_i)$$

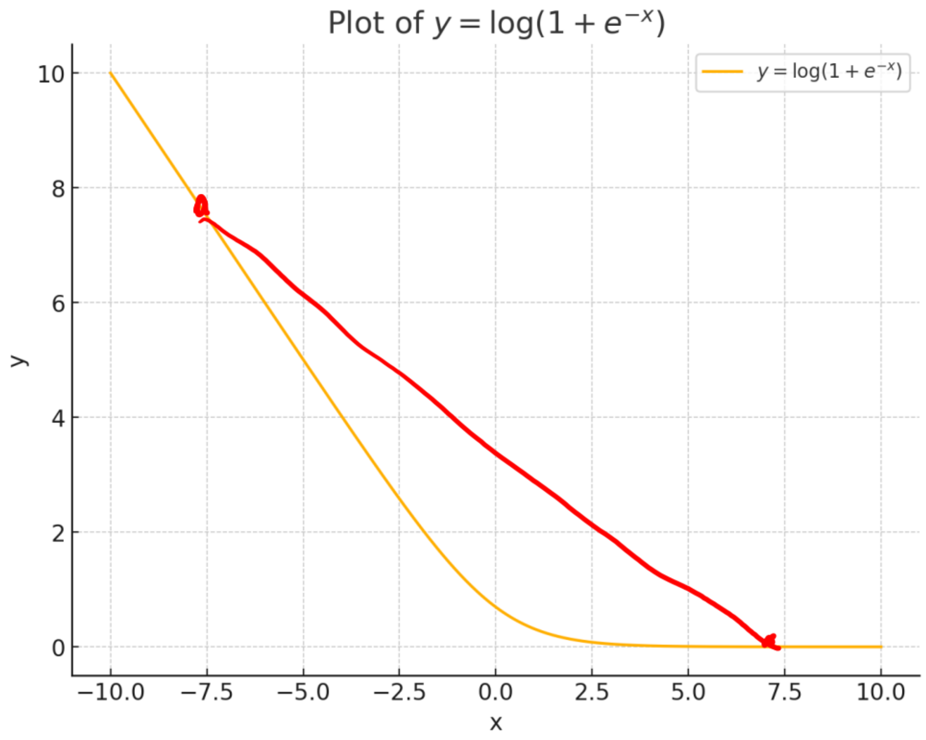
- We have i.i.d. training data $\{(x_i, y_i)\}_{i=1}^{n}$, $x_i \in \mathbb{R}^d$, $y_i \in \{-1,1\}$
- $P_w(y_i|x_i) = \sigma(y_i w^\top x_i) = 1/(1 + \exp(-y_i w^\top x_i))$
- $\ell_i(w) = -\log P_w(y_i|x_i) = \log(1 + \exp(-y_i w^\top x_i))$

$$\widehat{w}_{\mathrm{MLE}} = \mathrm{argmin}_w \sum_{i=1}^{n} \log(1 + \exp(-y_i w^\top x_i))$$

Computing the logistic regression solution

$$\widehat{w}_{\mathrm{MLE}} = \mathrm{argmin}_w \sum_{i=1}^{n} \log(1 + \exp(-y_i w^\top x_i))$$

- No closed form…
- But, this is a smooth convex problem!
- Optimize via gradient descent

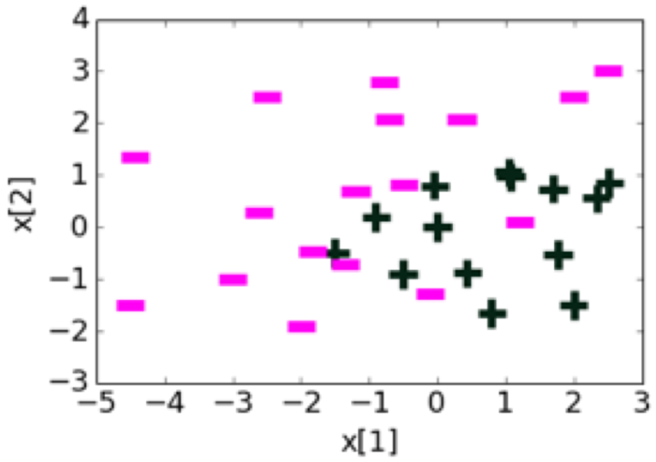


Example: Adding more polynomial features

$w^T x$

$w^T h(x)$

- $x \in \mathbb{R}^2, y \in \{-1, +1\}$
- Features: Polynomials
- Model: Linear on polynomial features

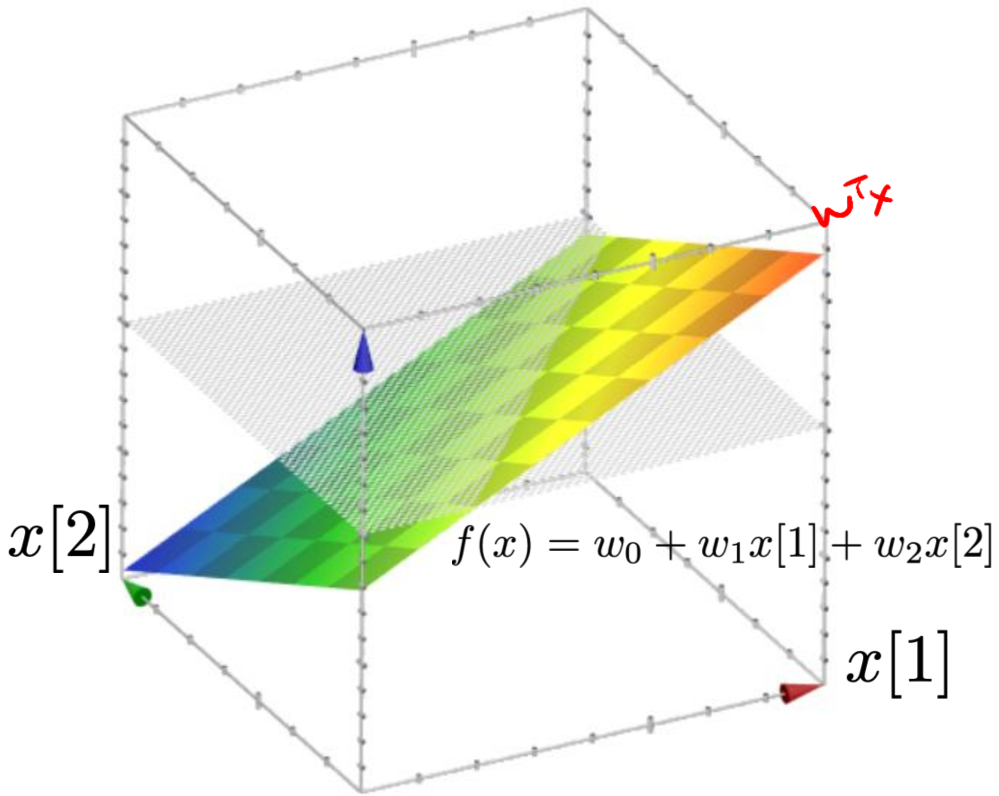


Polynomial features

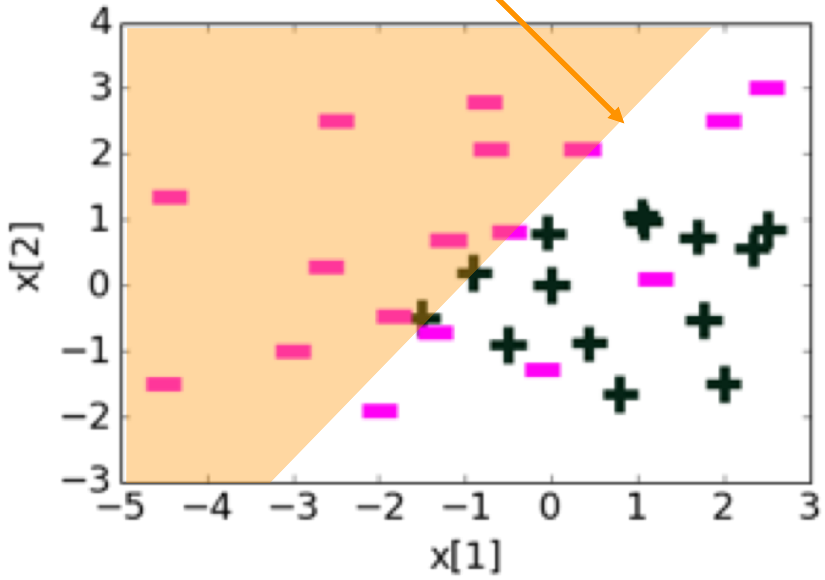
$$\begin{bmatrix} h_0(x) = 1 \\ h_1(x) = x[1] \\ h_2(x) = x[2] \\ h_3(x) = x[1]^2 \\ h_4(x) = x[2]^2 \\ \vdots \end{bmatrix}$$

$$f(x) = w_0 h_0(x) + w_1 h_1(x) + w_2 h_2(x) + \cdots$$

Linear features



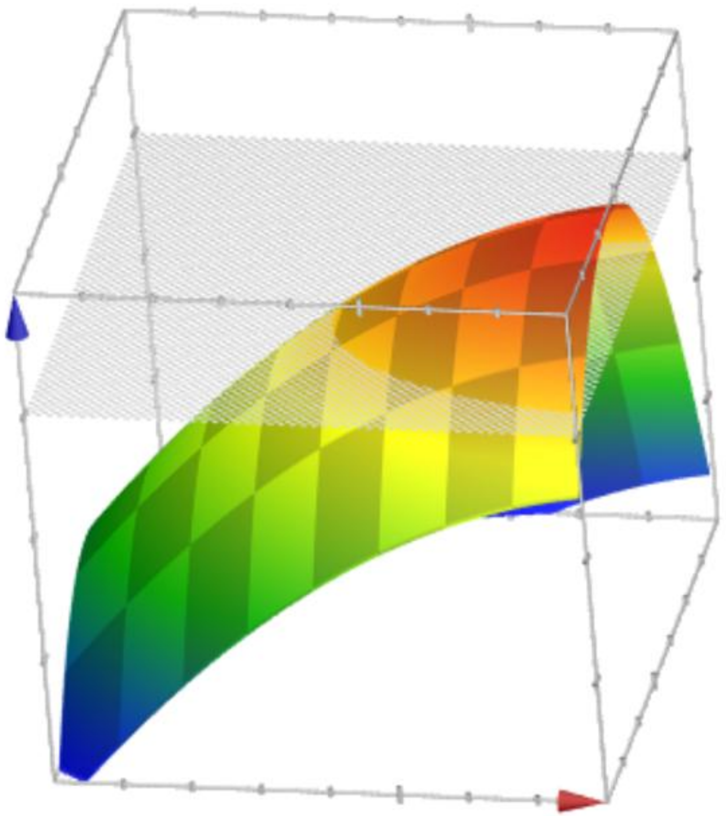
Decision boundary

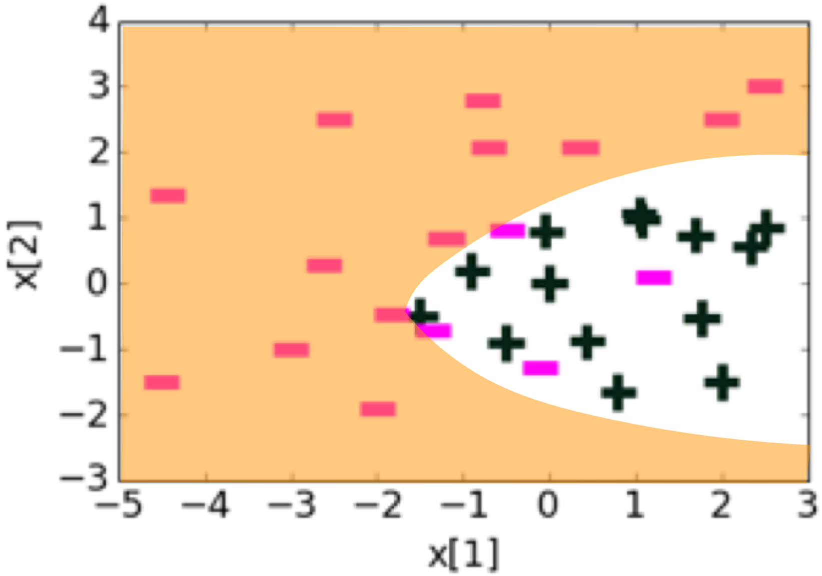


Feature	Value	Coefficient
$h_0(x)$	1	0,23
$h_1(x)$	x[1]	1,12
$h_2(x)$	x[2]	-1,07

- Simple regression models → smooth predictors
- Simple classifier models → smooth decision boundaries

Quadratic features

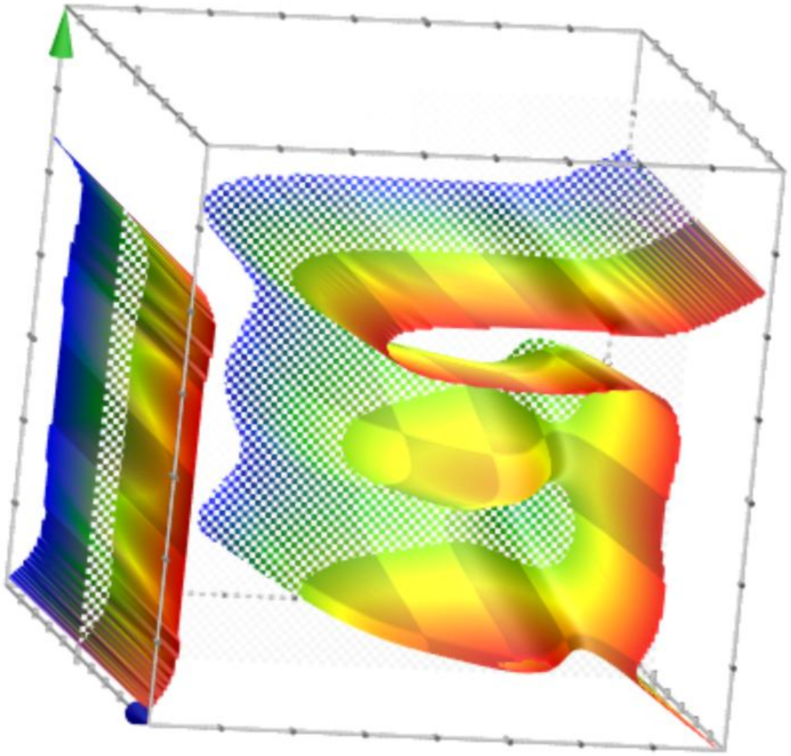


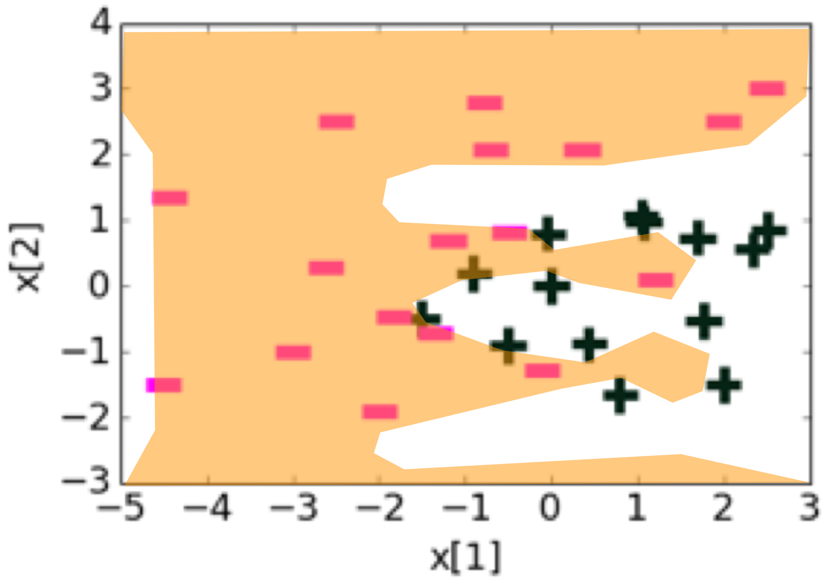


Feature	Value	Coefficient
$h_0(x)$	1	1,68
$h_1(x)$	x[1]	1,39
$h_2(x)$	x[2]	-0,59
$h_3(x)$	$(x[1])^2$	-0,17
$h_4(x)$	$(x[2])^2$	-0,96
$h_5(x)$	x[1]x[2]	Omitted

- Adding more features gives more complex models
- Decision boundary becomes more complex

Higher-degree polynomial features





Feature	Value	Coefficient learned
$h_0(x)$	1	21.6
$h_1(x)$	x[1]	5.3
$h_2(x)$	x[2]	-42.7
$h_3(x)$	$(x[1])^2$	-15.9
$h_4(x)$	$(x[2])^2$	-48.6
$h_5(x)$	$(x[1])^3$	-11.0
$h_6(x)$	$(x[2])^3$	67.0
$h_7(x)$	$(x[1])^4$	1.5
$h_8(x)$	$(x[2])^4$	48.0
$h_9(x)$	$(x[1])^5$	4.4
$h_{10}(x)$	$(x[2])^5$	-14.2
$h_{11}(x)$	$(x[1])^6$	0.8
$h_{12}(x)$	$(x[2])^6$	-8.6

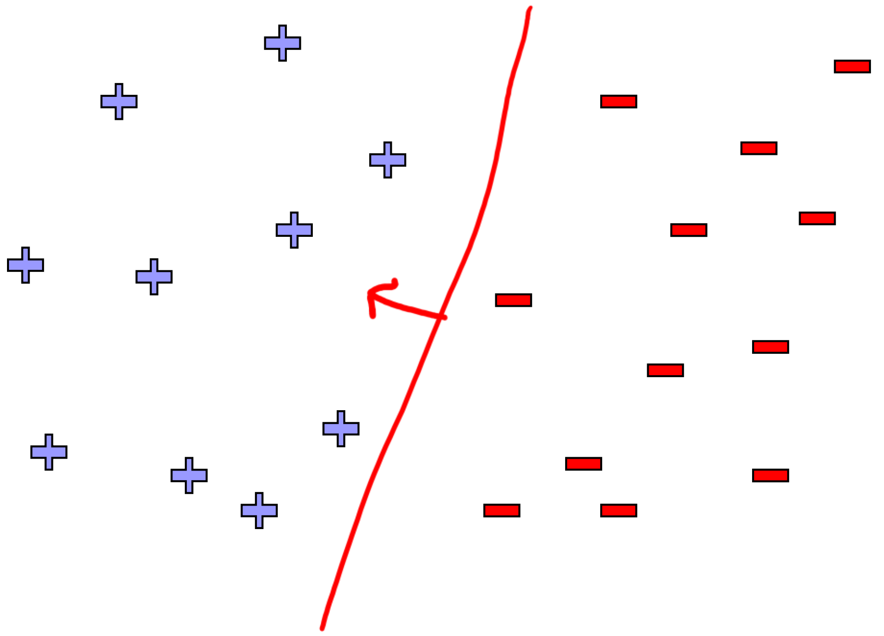
- Overfitting leads to poor generalization

Overfitting

want $y w^T x$ to be high

$$\widehat{w}_{\text{MLE}} = \text{argmin}_w \sum_{i=1}^{n} \log(1 + \exp(-y_i w^\top x_i))$$

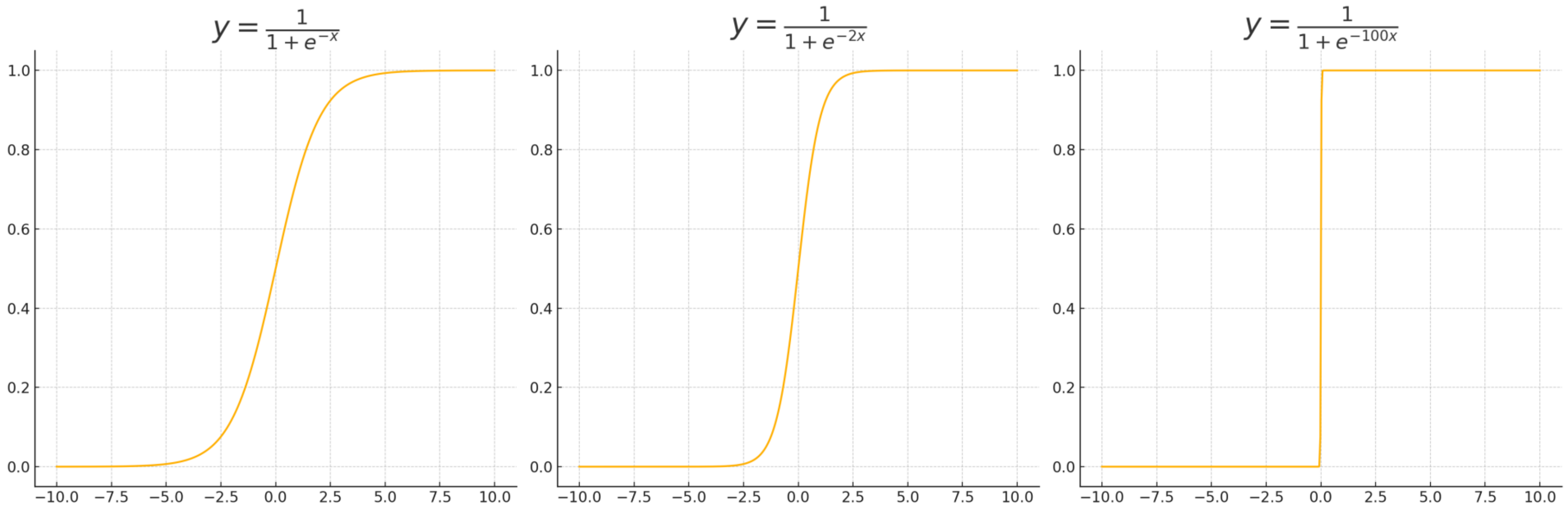
- What happens in this case?



Overfitting and linear separability

$$\widehat{w}_{\mathrm{MLE}} = \operatorname{argmin}_w \sum_{i=1}^{n} \log(1 + \exp(-y_i w^\top x_i))$$

- When data is linearly separable, $||w|| \to \infty$



Regularized logistic regression

sparse

$$\widehat{w}_{\text{reg}} = \text{argmin}_w \sum_{i=1}^{n} \log(1 + \exp(-y_i w^\top x_i)) \quad + \lambda \|w\|_1$$

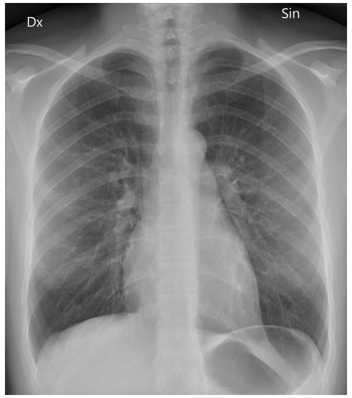
$$+ \frac{\lambda}{2}\|w\|_2^2$$

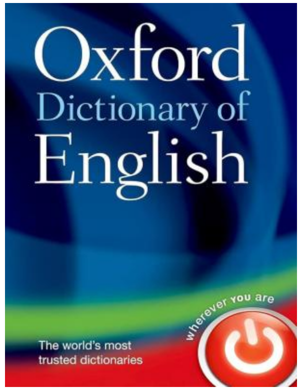
Section: Midterm prep this Friday

- Come with your questions!

Multi-class classification

- So far: binary $y \in \{-1, +1\}$
- In general: $y \in \{c_1, c_2, \dots, c_k\}$
- c_j's are called classes or labels

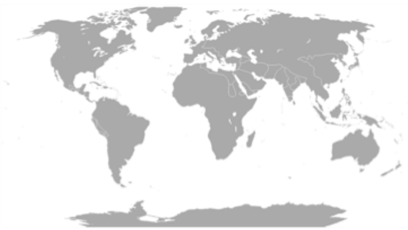




- A k-class classifier predicts y given x

Encoding categorical labels c_j

- For optimization, we need to embed raw c_j into real-valued vectors
- We typically use one-hot embeddings (a.k.a. one-hot encodings)
  - Each class is a standard basis vector in $\mathbb{R}^k$



Country of birth
(Argentina, Brazil, USA,...)

196 categories

1-hot encoding

x	$h_1(x)$	$h_2(x)$	...	$h_{195}(x)$	$h_{196}(x)$
Brazil					
Zimbabwe					

196 features

Multi-class logistic regression (aka softmax classification)

- Data: features $x_i \in \mathbb{R}^d$, categorical $y \in \{c_1, \dots c_k\}$
- One-hot encoding $\in \mathbb{R}^k$ s.t. $y = [1,0,0, \dots]$ implies $y = c_1$
- Model: linear prediction $\hat{y}_i = f(x_i) = \text{softmax}(w^\top x) \in \mathbb{R}^k$
- Parameter matrix $\mathrm{w} \in \mathbb{R}^{d \times k}$

Softmax classification

without loss of generality, set $\mathrm{k} = 2$, $w_1 = 0$

2 classes

k classes

Conditional probabilities

$$P(y = 1|x) = \frac{1}{1 + e^{-w^\top x}} = \frac{e^{w^\top x}}{1 + e^{w^\top x}}$$

$$P(y = -1|x) = \frac{1}{1 + e^{w^\top x}}$$

$$P\left(y = c_j \middle| x\right) = \frac{e^{w_j^\top x}}{\sum_{j'} e^{w_{j'}^\top x}}$$

MLE

$$\mathrm{argmax}_w \sum_{i=1}^{n} \log\left(\frac{1}{1 + \mathrm{e}^{-y_i w^\top x_i}}\right)$$

$$\mathrm{argmax}_w \sum_{i=1}^{n} \sum_{j=1}^{k} 1\{y_i = c_j\} \log\left(\frac{\mathrm{e}^{w_j^\top x_i}}{\sum_{j'=1}^{k} e^{w_{j'}^\top x_i}}\right)$$

Regression and classification

- ML paradigm: define prediction $f_w(x)$ and loss $\ell(f_w(x), y)$
- Then optimize:

$$\hat{w} = \mathrm{argmin}_w \sum_{i=1}^{n} \ell(f_w(x_i), y_i)$$

- Squared error loss: $\ell(f_w(x), y) = \left(y - f_w(x)\right)^2$
- Logistic loss: $\ell(f_w(x), y) = \log(1 + \exp\left(-y f_w(x)\right))$

Regression and classification

- Can we treat classification as a regression problem?

- Can we treat regression as a classification problem?

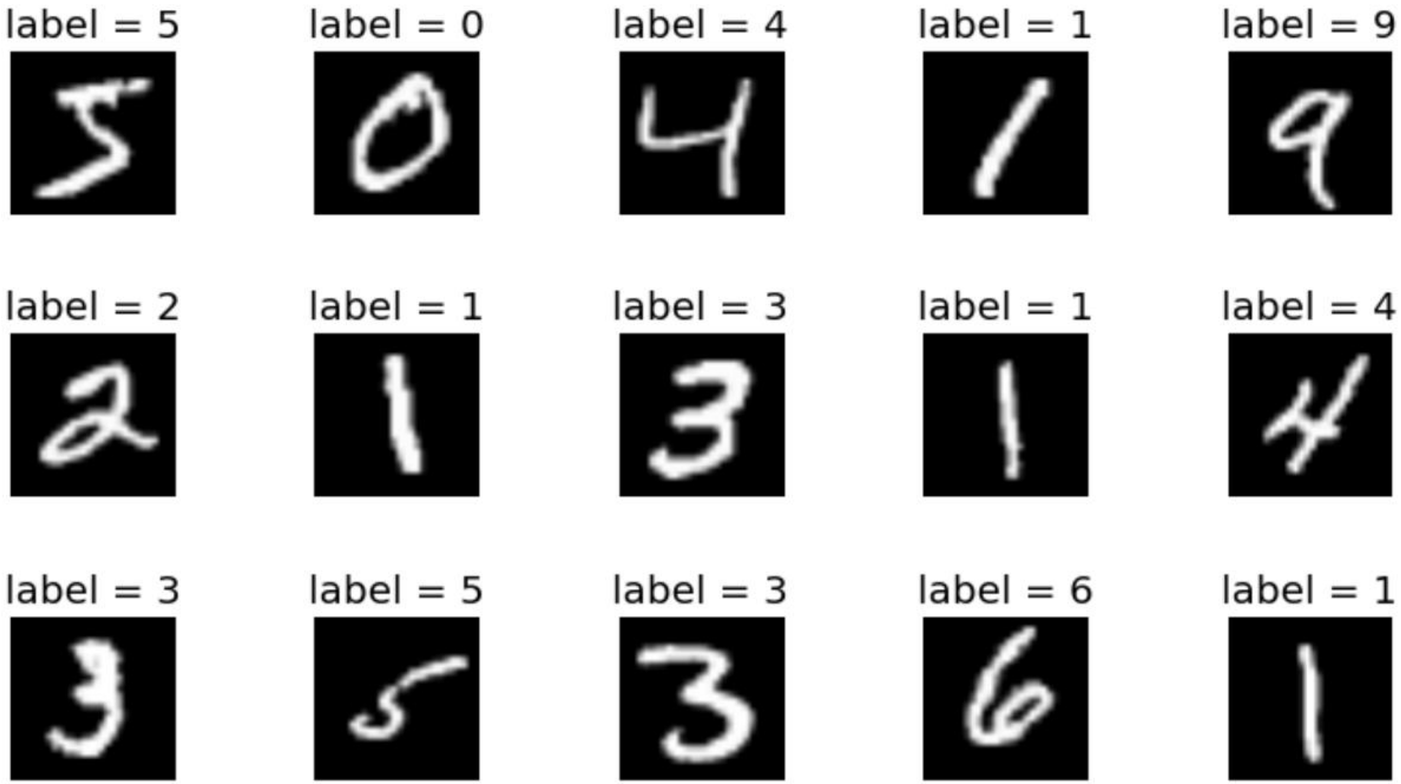
Regression and classification

Temperature: 62F

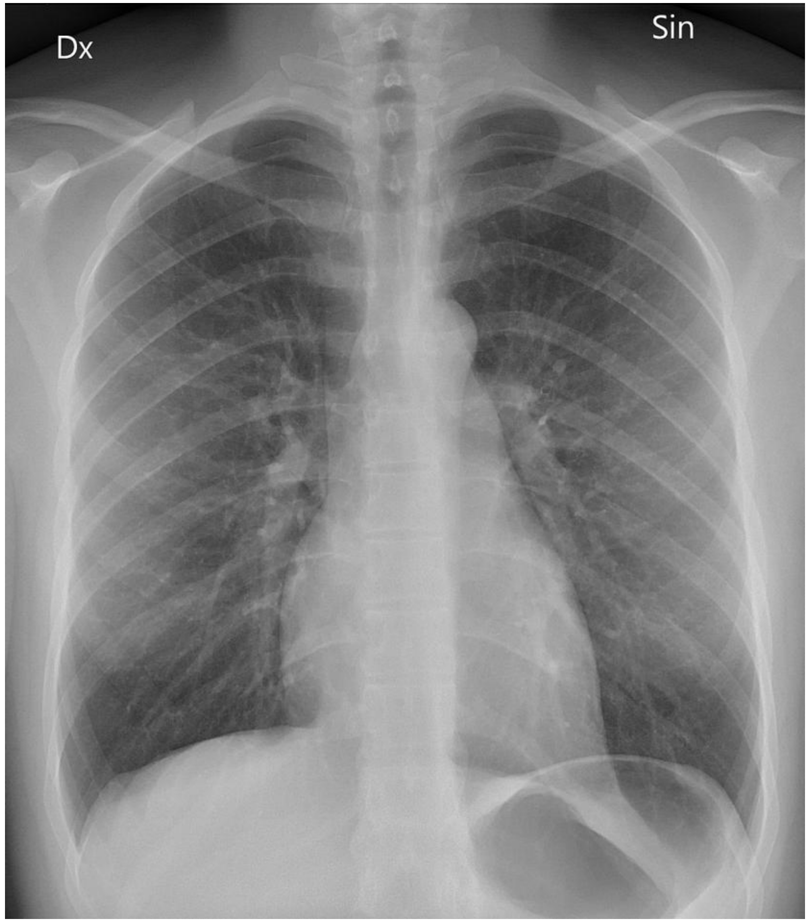
Regression and classification

$\longrightarrow$ 98105

Regression and classification



Multi-class vs multi-label classification



Healthy
Pneumonia
Pneumothorax
Pleural Effusion
…

Summary

- Classification problems (where y is categorical) are everywhere
- Regression losses are not typically appropriate
- Logistic regression: model conditional probability $\mathrm{P}(y|x)$ as sigmoid (then apply usual MLE machinery)
- Softmax classification: generalized to $k > 2$ classes
- Regularization is still important

Recap: The ML pipeline

1. Define the **task** (what type of data, what type of eval metrics?)
2. Collect and preprocess **data**
3. Choose **model** family/parameterization
4. Choose **training loss**
4. For each choice of hyperparameters:
   - **Optimize** model (minimize loss) on training data
   - **Evaluate** on validation data
5. Pick best hyperparameters according to validation performance
6. **Evaluate** final model on test data

Cross-validation

- Cross-validation refers to splitting available data into training vs validation portions. It includes:
  - K-fold cross validation
  - Fixed train/validation split

- Should we also train on the validation set after selecting hyperparameters?

The ML pipeline

1. Task
2. Data
3. Model family
4. Training loss
4. Optimize
5. Pick hyperparameters
6. Evaluate

The ML pipeline

1. Task
2. Data
3. Model family
4. Training loss
4. Optimize
5. Pick hyperparameters
6. Evaluate