10-601 Introduction to Machine Learning

Machine Learning Department
School of Computer Science
Carnegie Mellon University

# Perceptron (Theory) + Linear Regression

Matt Gormley
Lecture 6
Feb. 5, 2018

# Q&A

**Q:** I can't read the chalkboard, can you write larger?

**A:** Sure. Just raise your hand and let me know if you can't read something.

**Q:** I'm concerned that you won't be able to read my solution in the homework template because it's so tiny, can I use my own template?

**A:** No. However, we do all of our grading online and can **zoom in** to view your solution! Make it as small as you need to.

# Reminders

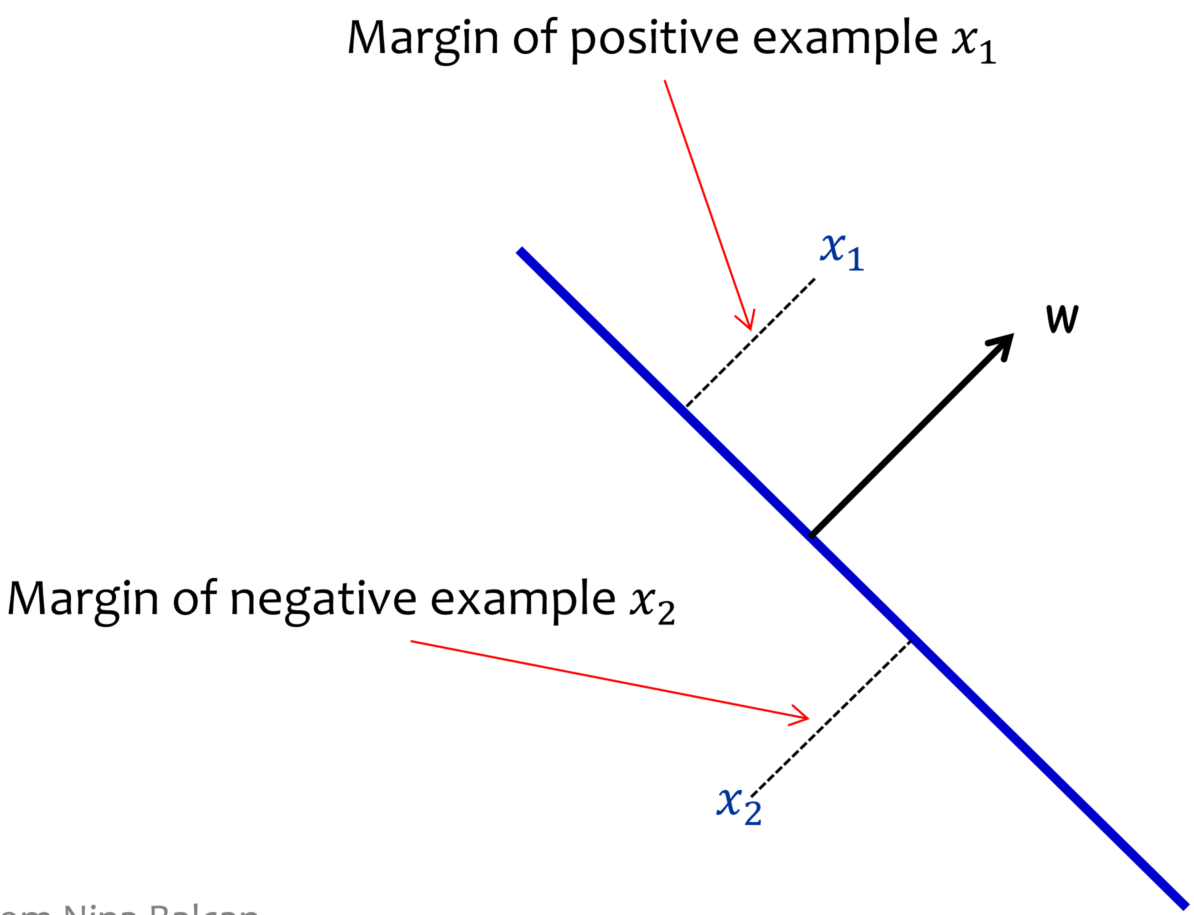
- **Homework 2: Decision Trees**
  - Out: Wed, Jan 24
  - Due: Mon, Feb 5 at 11:59pm
- **Homework 3: KNN, Perceptron, Lin.Reg.**
  - Out: Mon, Feb 5
  - Due: Mon, Feb 12 at 11:59pm

…possibly delayed by two days

# ANALYSIS OF PERCEPTRON

# Geometric Margin

**Definition:** The margin of example $x$ w.r.t. a linear sep. $w$ is the distance from $x$ to the plane $w \cdot x = 0$ (or the negative if on wrong side)

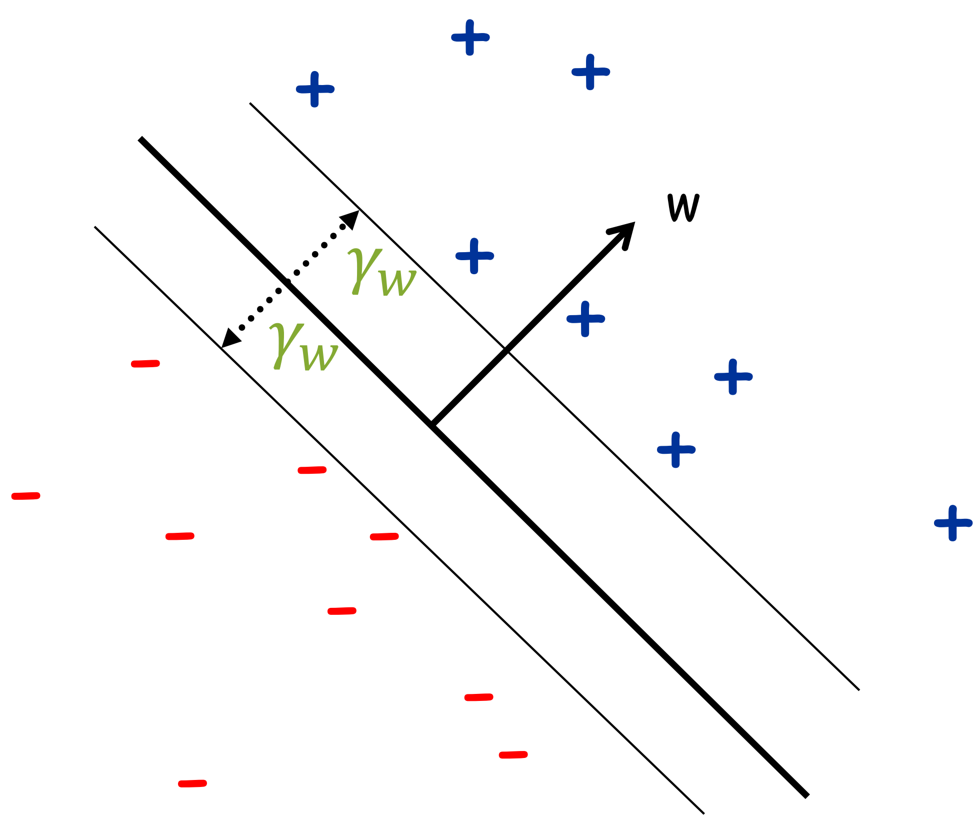



Slide from Nina Balcan

# Geometric Margin

**Definition:** The margin of example $x$ w.r.t. a linear sep. $w$ is the distance from $x$ to the plane $w \cdot x = 0$ (or the negative if on wrong side)

**Definition:** The margin $\gamma_w$ of a set of examples $S$ wrt a linear separator $w$ is the smallest margin over points $x \in S$.



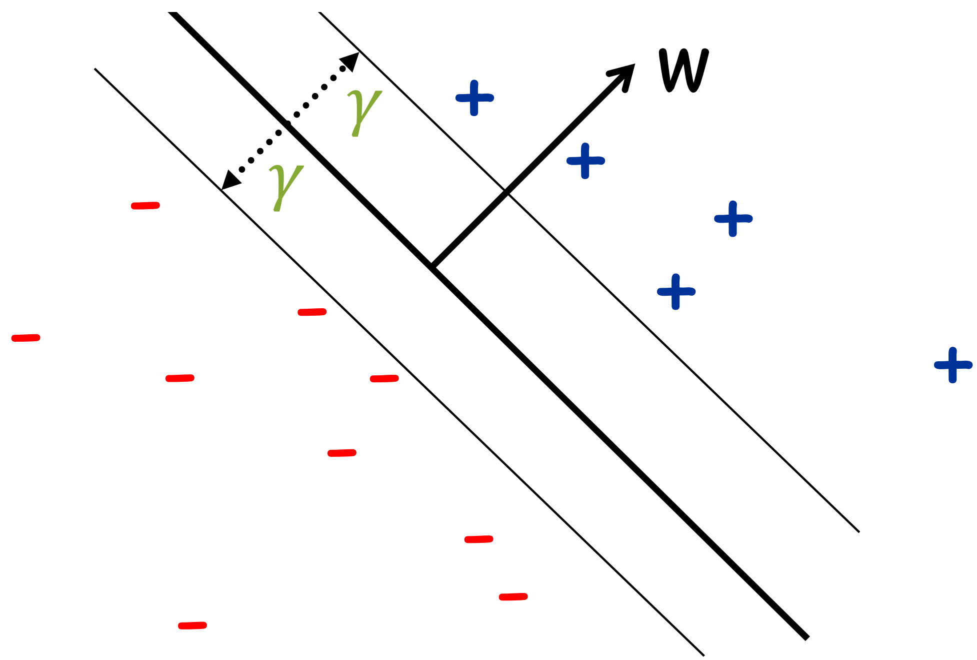

Slide from Nina Balcan

# Geometric Margin

**Definition:** The margin of example $x$ w.r.t. a linear sep. $w$ is the distance from $x$ to the plane $w \cdot x = 0$ (or the negative if on wrong side)

**Definition:** The margin $\gamma_w$ of a set of examples $S$ wrt a linear separator $w$ is the smallest margin over points $x \in S$.

**Definition:** The margin $\gamma$ of a set of examples $S$ is the maximum $\gamma_w$ over all linear separators $w$.

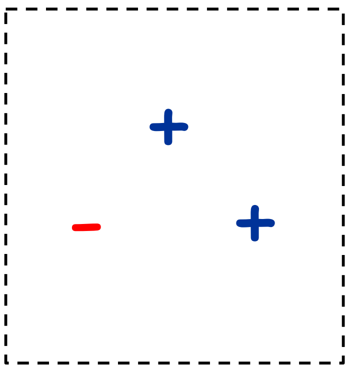



Slide from Nina Balcan

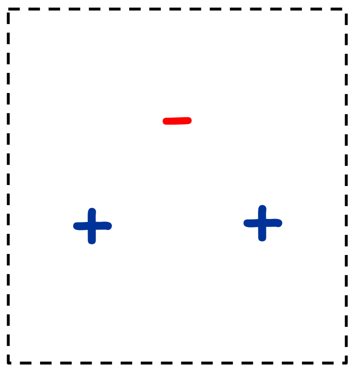
# Linear Separability

***Def***: For a **binary classification** problem, a set of examples $S$ is **linearly separable** if there exists a linear decision boundary that can separate the points

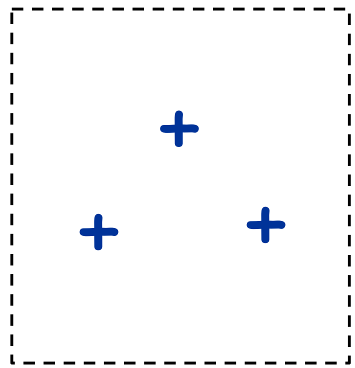
Case 1:



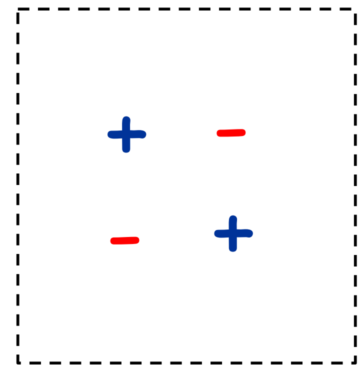
Case 2:



Case 3:



Case 4:

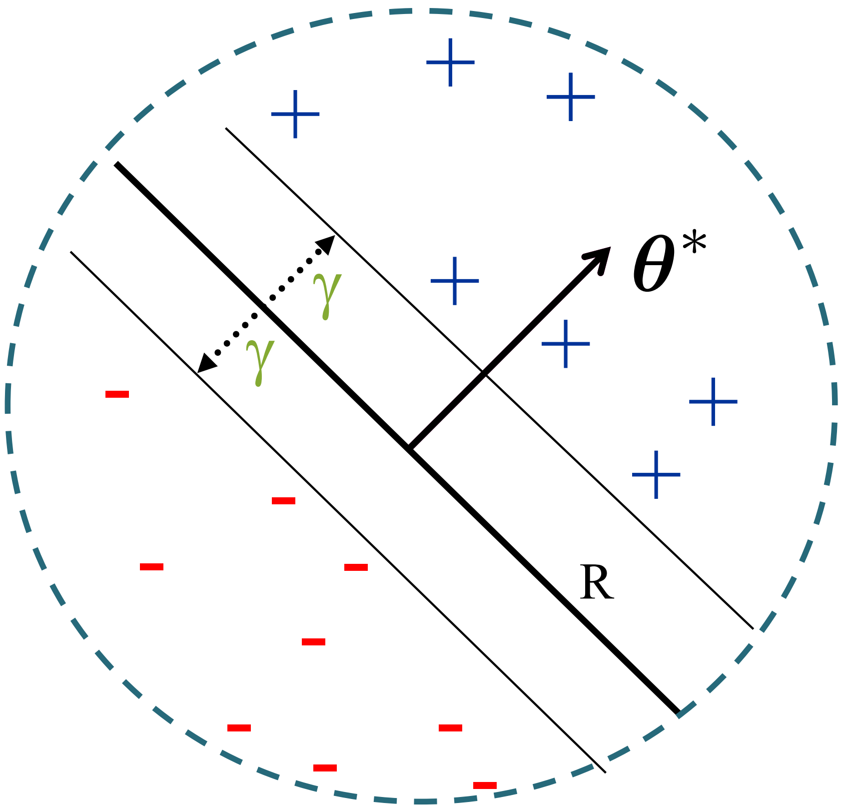


# Analysis: Perceptron

**Perceptron Mistake Bound**

**Guarantee:** If data has margin $\gamma$ and all points inside a ball of radius $R$, then Perceptron makes $\leq (R/\gamma)^2$ mistakes.

(Normalized margin: multiplying all points by 100, or dividing all points by 100, doesn't change the number of mistakes; algo is invariant to scaling.)





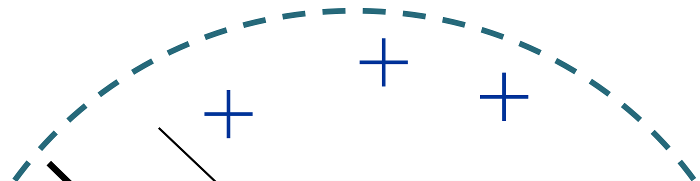

Slide adapted from Nina Balcan

# Analysis: Perceptron

**Perceptron Mistake Bound**

**Guarantee:** If data has margin $\gamma$ and all points inside a ball of radius $R$, then Perceptron makes $\leq (R/\gamma)^2$ mistakes.

(Normalized margin: multiplying all points by 100, or dividing all points by 100, doesn't change the number of mistakes; algo is invariant to scaling.)




***Def:*** We say that the (batch) perceptron algorithm has **converged** if it stops making mistakes on the training data (perfectly classifies the training data).

***Main Takeaway***: For **linearly separable** data, if the perceptron algorithm cycles repeatedly through the data, it will **converge** in a finite # of steps.

Slide adapted from Nina Balcan

# Analysis: Perceptron

**Perceptron Mistake Bound**

**Theorem 0.1** (Block (1962), Novikoff (1962)).
*Given dataset:* $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$.
*Suppose:*

1. *Finite size inputs:* $||x^{(i)}|| \leq R$
2. *Linearly separable data:* $\exists \boldsymbol{\theta}^*$ *s.t.* $||\boldsymbol{\theta}^*|| = 1$ *and* $y^{(i)}(\boldsymbol{\theta}^* \cdot \mathbf{x}^{(i)}) \geq \gamma, \forall i$

*Then: The number of mistakes made by the Perceptron algorithm on this dataset is*

$$k \leq (R/\gamma)^2$$

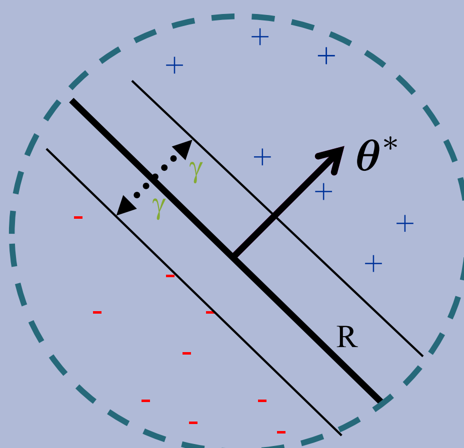



Figure from Nina Balcan

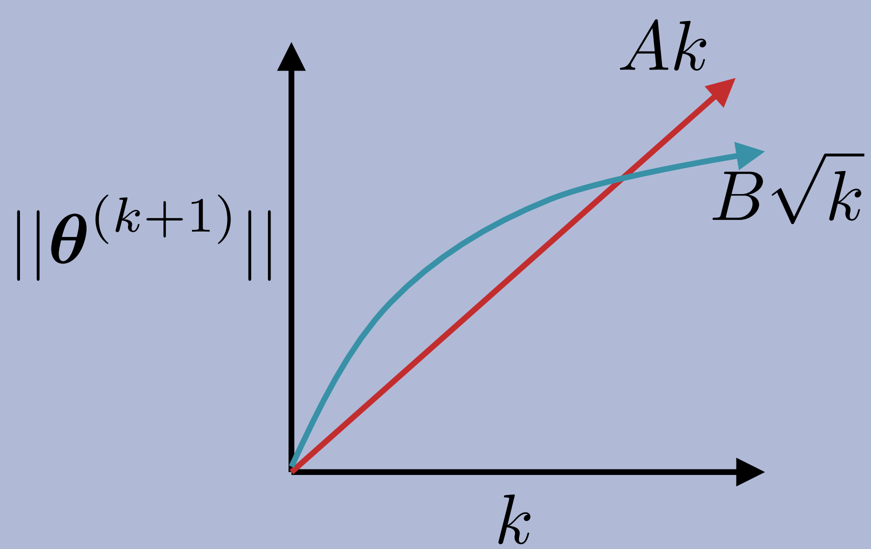
# Analysis: Perceptron

Covered in Recitation

**Proof of Perceptron Mistake Bound:**

We will show that there exist constants A and B s.t.

$$Ak \leq ||\boldsymbol{\theta}^{(k+1)}|| \leq B\sqrt{k}$$



Covered in Recitation

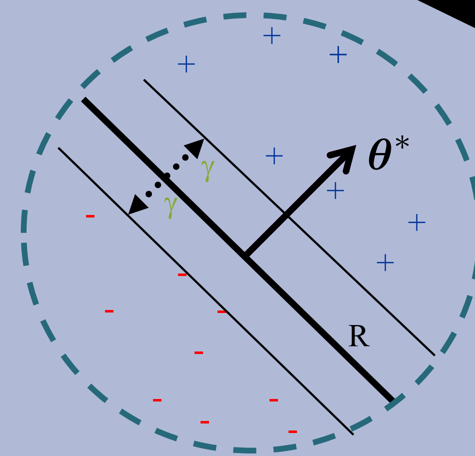
# Analysis: Perceptron

**Theorem 0.1** (Block (1962), Novikoff (1962)).
*Given dataset:* $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$.
*Suppose:*

1. *Finite size inputs:* $||x^{(i)}|| \leq R$
2. *Linearly separable data:* $\exists \boldsymbol{\theta}^*$ *s.t.* $||\boldsymbol{\theta}^*|| = 1$ *and* $y^{(i)}(\boldsymbol{\theta}^* \cdot \mathbf{x}^{(i)}) \geq \gamma, \forall i$

*Then: The number of mistakes made by the Perceptron algorithm on this dataset is*

$$k \leq (R/\gamma)^2$$




**Algorithm 1** Perceptron Learning Algorithm (Online)

```
procedure PERCEPTRON(D = {(x^(1), y^(1)), (x^(2), y^(2)), ...})
    θ ← 0, k = 1                              ▷ Initialize parameters
    for i ∈ {1, 2, ...} do                    ▷ For each example
        if y^(i)(θ^(k) · x^(i)) ≤ 0 then      ▷ If mistake
            θ^(k+1) ← θ^(k) + y^(i) x^(i)     ▷ Update parameters
            k ← k + 1
    return θ
```

# Analysis: Perceptron

Covered in Recitation

**Proof of Perceptron Mistake Bound:**

Part 1: for some A, $Ak \leq ||\boldsymbol{\theta}^{(k+1)}||$

$$\boldsymbol{\theta}^{(k+1)} \cdot \boldsymbol{\theta}^* = (\boldsymbol{\theta}^{(k)} + y^{(i)}\mathbf{x}^{(i)})\boldsymbol{\theta}^*$$

by Perceptron algorithm update

$$= \boldsymbol{\theta}^{(k)} \cdot \boldsymbol{\theta}^* + y^{(i)}(\boldsymbol{\theta}^* \cdot \mathbf{x}^{(i)})$$

$$\geq \boldsymbol{\theta}^{(k)} \cdot \boldsymbol{\theta}^* + \gamma$$

by assumption

$$\Rightarrow \boldsymbol{\theta}^{(k+1)} \cdot \boldsymbol{\theta}^* \geq k\gamma$$

by induction on $k$ since $\theta^{(1)} = \mathbf{0}$

$$\Rightarrow ||\boldsymbol{\theta}^{(k+1)}|| \geq k\gamma$$

since $||\mathbf{w}|| \times ||\mathbf{u}|| \geq \mathbf{w} \cdot \mathbf{u}$ and $||\theta^*|| = 1$

Cauchy-Schwartz inequality

# Analysis: Perceptron

Covered in Recitation

**Proof of Perceptron Mistake Bound:**

Part 2: for some B, $||\boldsymbol{\theta}^{(k+1)}|| \leq B\sqrt{k}$

$$||\boldsymbol{\theta}^{(k+1)}||^2 = ||\boldsymbol{\theta}^{(k)} + y^{(i)}\mathbf{x}^{(i)}||^2$$

by Perceptron algorithm update

$$= ||\boldsymbol{\theta}^{(k)}||^2 + (y^{(i)})^2||\mathbf{x}^{(i)}||^2 + 2y^{(i)}(\boldsymbol{\theta}^{(k)} \cdot \mathbf{x}^{(i)})$$

$$\leq ||\boldsymbol{\theta}^{(k)}||^2 + (y^{(i)})^2||\mathbf{x}^{(i)}||^2$$

since $k$th mistake $\Rightarrow y^{(i)}(\boldsymbol{\theta}^{(k)} \cdot \mathbf{x}^{(i)}) \leq 0$

$$= ||\boldsymbol{\theta}^{(k)}||^2 + R^2$$

since $(y^{(i)})^2||\mathbf{x}^{(i)}||^2 = ||\mathbf{x}^{(i)}||^2 = R^2$ by assumption and $(y^{(i)})^2 = 1$

$$\Rightarrow ||\boldsymbol{\theta}^{(k+1)}||^2 \leq kR^2$$

by induction on $k$ since $(\theta^{(1)})^2 = 0$

$$\Rightarrow ||\boldsymbol{\theta}^{(k+1)}|| \leq \sqrt{k}R$$

# Analysis: Perceptron

Covered in Recitation

**Proof of Perceptron Mistake Bound:**

Part 3: Combining the bounds finishes the proof.

$$k\gamma \leq ||\boldsymbol{\theta}^{(k+1)}|| \leq \sqrt{k}R$$

$$\Rightarrow k \leq (R/\gamma)^2$$

The total number of mistakes must be less than this

# Analysis: Perceptron

**What if the data is *not* linearly separable?**

1. Perceptron will **not converge** in this case (it can't!)
2. However, Freund & Schapire (1999) show that by projecting the points (hypothetically) into a higher dimensional space, we can achieve a similar bound on the number of mistakes made on **one pass** through the sequence of examples

**Theorem 2.** *Let* $\langle(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\rangle$ *be a sequence of labeled examples with* $\|\mathbf{x}_i\| \leq R$. *Let* $\mathbf{u}$ *be any vector with* $\|\mathbf{u}\| = 1$ *and let* $\gamma > 0$. *Define the deviation of each example as*

$$d_i = \max\{0, \gamma - y_i(\mathbf{u} \cdot \mathbf{x}_i)\},$$

*and define* $D = \sqrt{\sum_{i=1}^{m} d_i^2}$. *Then the number of mistakes of the online perceptron algorithm on this sequence is bounded by*

$$\left(\frac{R + D}{\gamma}\right)^2.$$

# Summary: Perceptron

- Perceptron is a **linear classifier**
- **Simple learning algorithm**: when a mistake is made, add / subtract the features
- Perceptron will converge if the data are **linearly separable**, it will **not** converge if the data are **linearly inseparable**
- For linearly separable and inseparable data, we can **bound the number of mistakes** (geometric argument)
- **Extensions** support nonlinear separators and structured prediction

# Perceptron Learning Objectives

*You should be able to…*

- Explain the difference between online learning and batch learning
- Implement the perceptron algorithm for binary classification [CIML]
- Determine whether the perceptron algorithm will converge based on properties of the dataset, and the limitations of the convergence guarantees
- Describe the inductive bias of perceptron and the limitations of linear models
- Draw the decision boundary of a linear model
- Identify whether a dataset is linearly separable or not
- Defend the use of a bias term in perceptron

# LINEAR REGRESSION

# Linear Regression Outline

- **Regression Problems**
  - Definition
  - Linear functions
  - Residuals
  - Notation trick: fold in the intercept
- **Linear Regression as Function Approximation**
  - Objective function: Mean squared error
  - Hypothesis space: Linear Functions
- **Optimization for Linear Regression**
  - Normal Equations (Closed-form solution)
    - Computational complexity
    - Stability
  - SGD for Linear Regression
    - Partial derivatives
    - Update rule
  - Gradient Descent for Linear Regression
- **Probabilistic Interpretation of Linear Regression**
  - Generative vs. Discriminative
  - Conditional Likelihood
  - Background: Gaussian Distribution
  - Case #1: 1D Linear Regression
  - Case #2: Multiple Linear Regression

# Regression Problems

*Whiteboard*

- Definition
- Linear functions
- Residuals
- Notation trick: fold in the intercept

# Linear Regression as Function Approximation

*Whiteboard*

- Objective function: Mean squared error
- Hypothesis space: Linear Functions

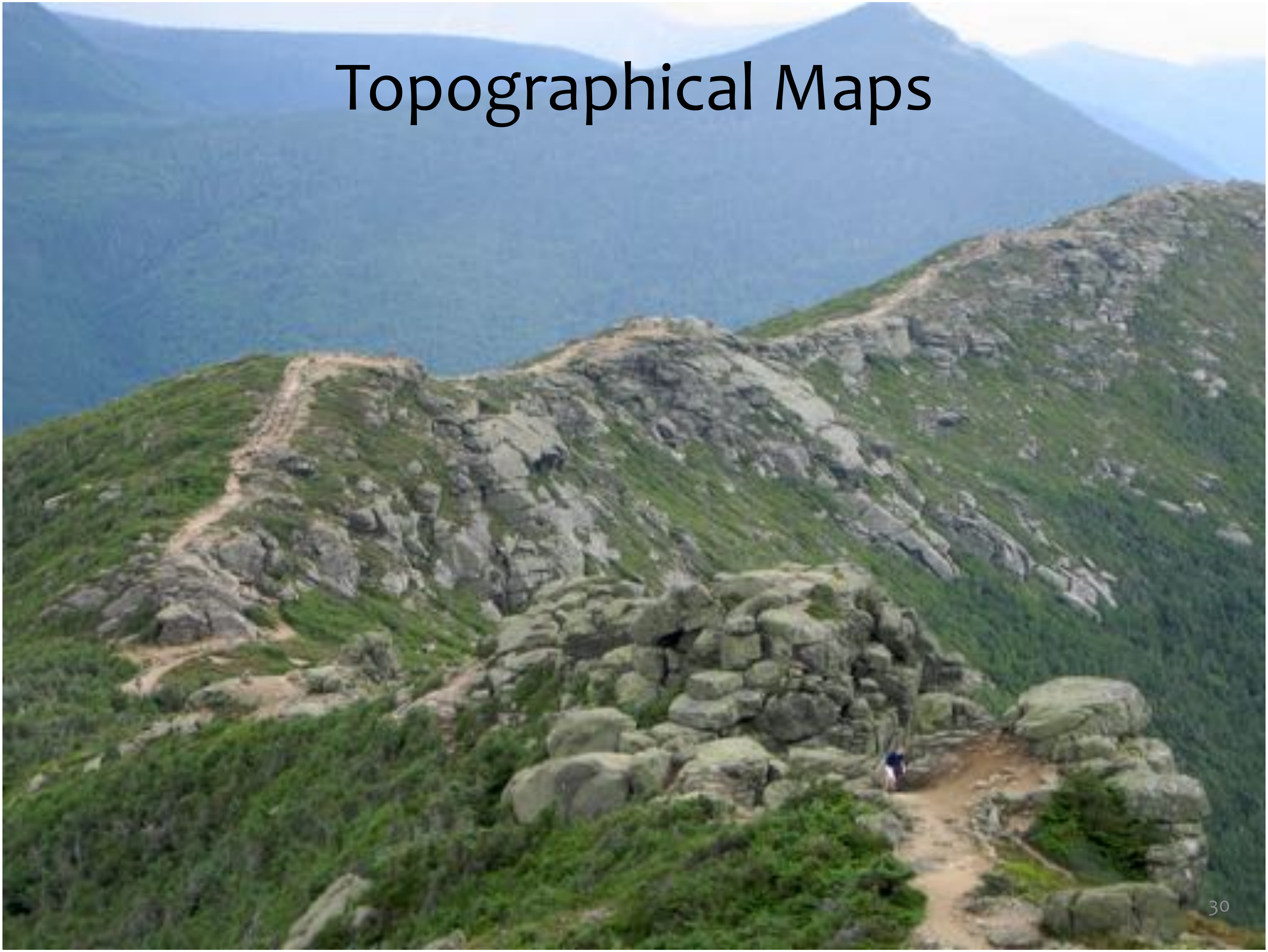
# OPTIMIZATION FOR ML

# Optimization for ML

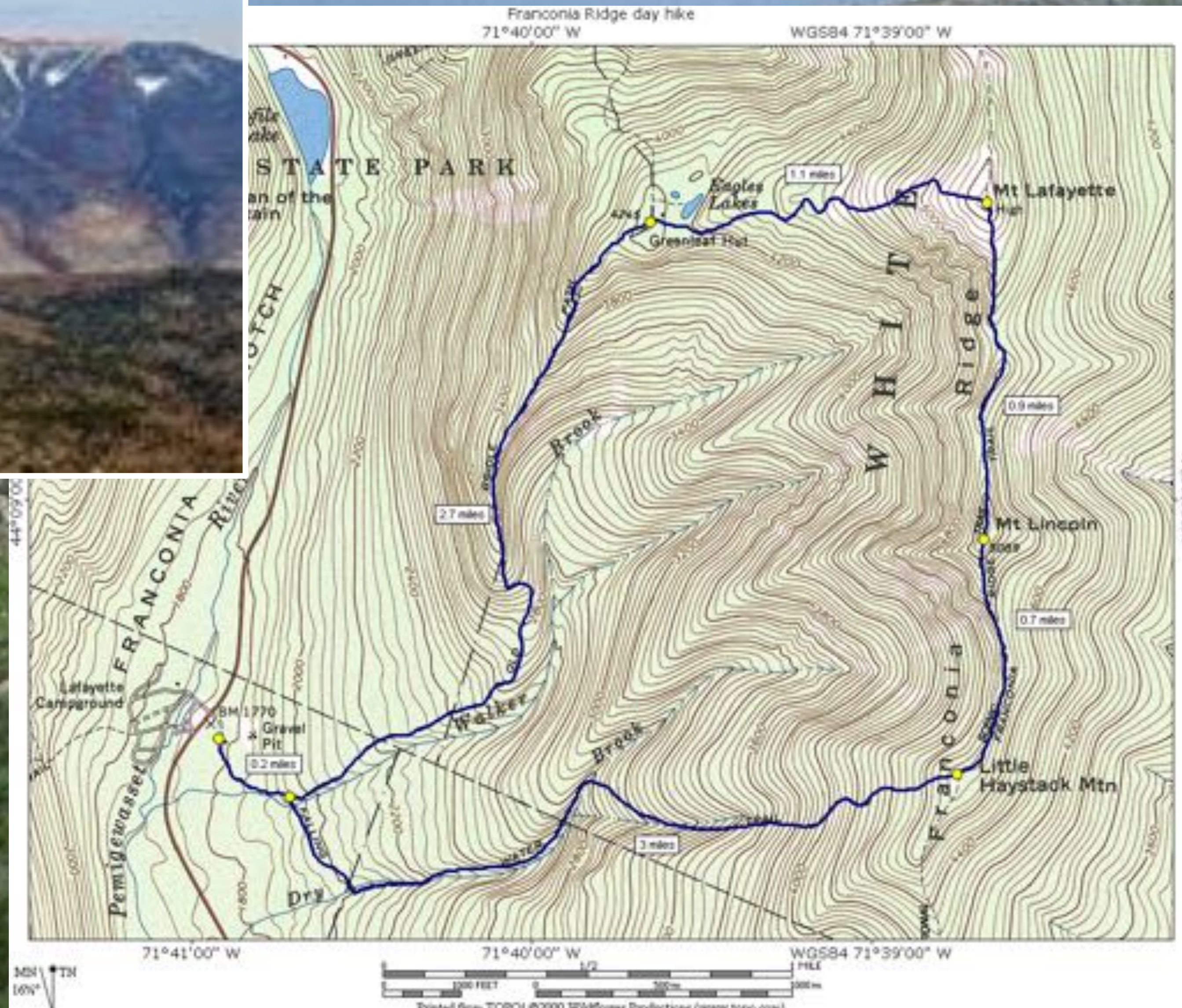
Not quite the same setting as other fields…

- Function we are optimizing might not be the true goal
  (e.g. likelihood vs generalization error)
- Precision might not matter
  (e.g. data is noisy, so optimal up to 1e-16 might not help)
- Stopping early can help generalization error
  (i.e. "early stopping" is a technique for regularization – discussed more next time)

# Topographical Maps



# Topographical Maps



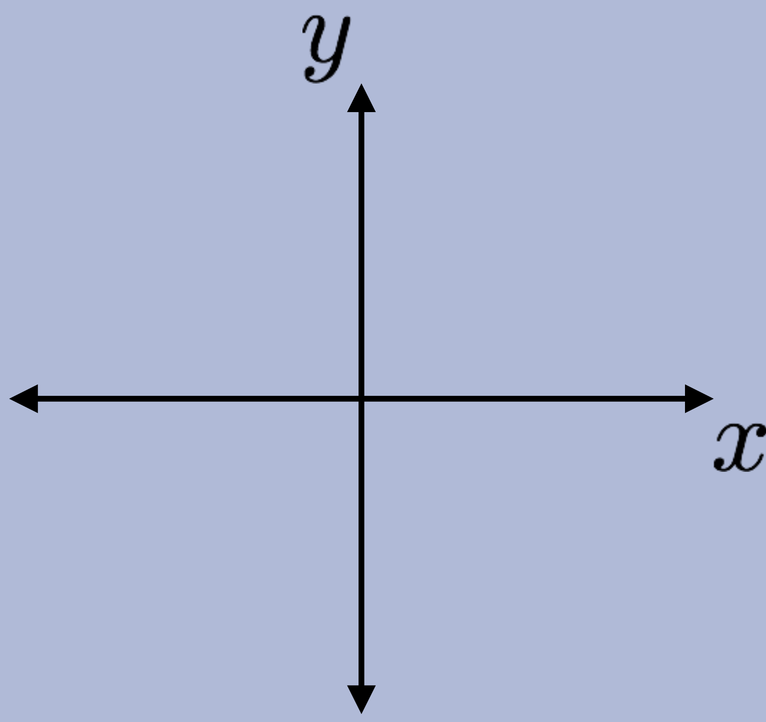
# Calculus

**In-Class Exercise**

Plot three functions:

1. $f(x) = x^3 - x$
2. $f'(x) = \frac{\partial y}{\partial x}$
3. $f''(x) = \frac{\partial^2 y}{\partial x^2}$

**Answer Here:**



# Optimization for ML

*Whiteboard*

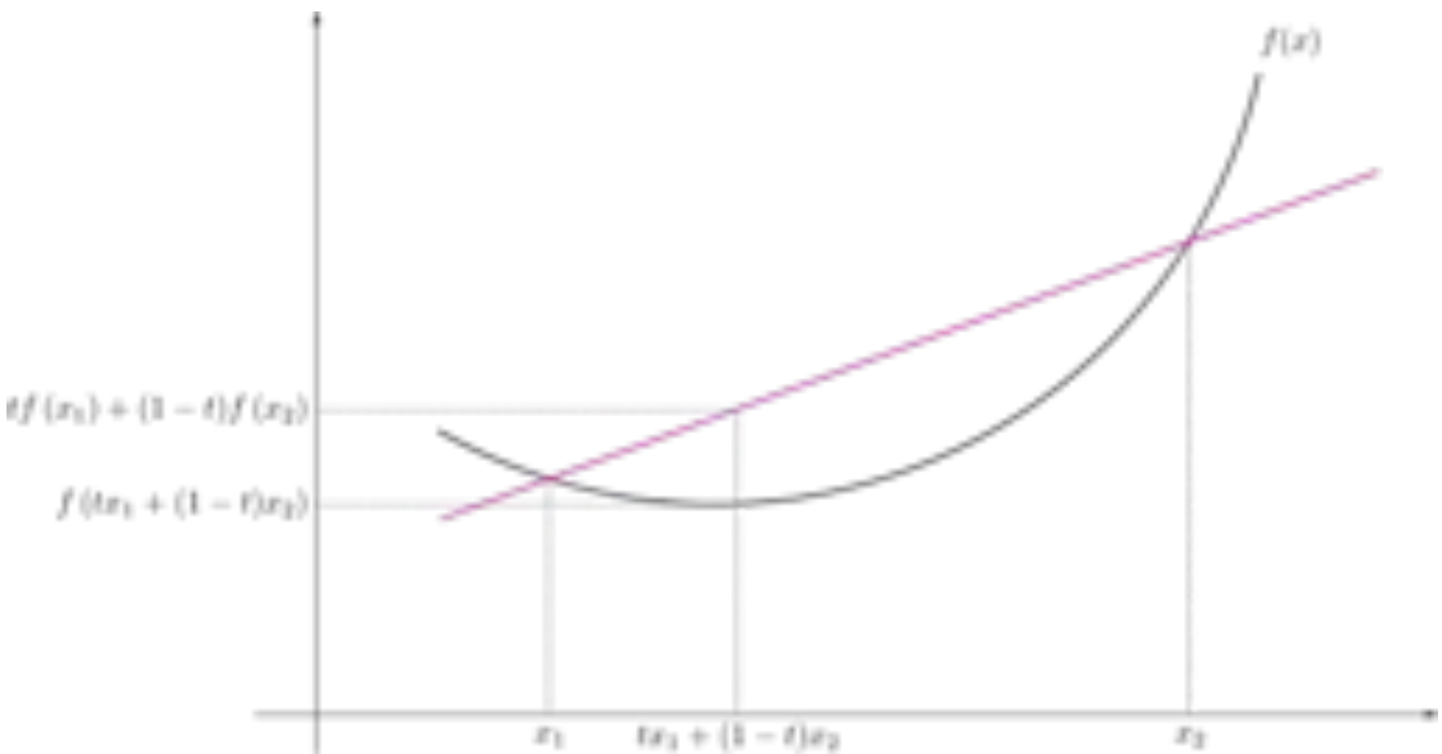
- Unconstrained optimization
- Convex, concave, nonconvex
- Derivatives
- Zero derivatives
- Gradient and Hessian

# Convexity

Function $f : \mathbb{R}^M \rightarrow \mathbb{R}$ is **convex**
if $\forall\ \mathbf{x}_1 \in \mathbb{R}^M, \mathbf{x}_2 \in \mathbb{R}^M, 0 \leq t \leq 1$:

$$f(t\mathbf{x}_1 + (1-t)\mathbf{x}_2) \leq tf(\mathbf{x}_1) + (1-t)f(\mathbf{x}_2)$$

There is only one local optimum if the function is *convex*



Slide adapted from William Cohen

# OPTIMIZATION FOR LINEAR REGRESSION

# Optimization for Linear Regression

*Whiteboard*

- Closed-form (Normal Equations)