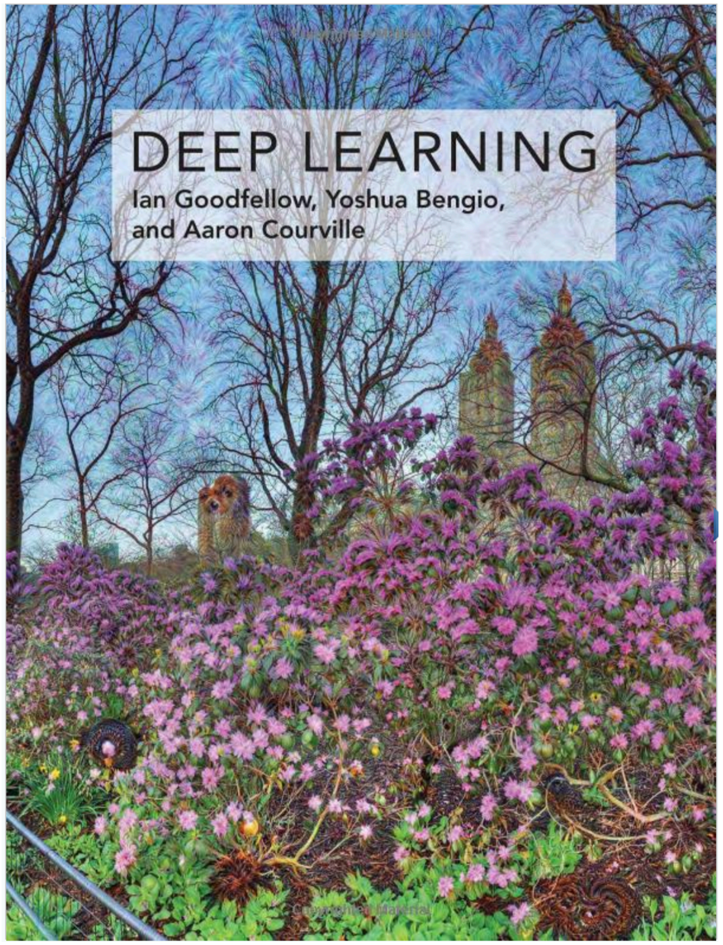
Lecture 2: Reviewing the basics of machine learning

Announcements:

- We sent out an Announcement on Bruin Learn on how to sign up for Gradescope and Piazza, as well as a poll on discussion section scheduling.
- We will set discussion sections and OH based on the polls. OH and discussions will commence in Week 2.
- We uploaded the formal notes of the class and exams dating back to 2019.
- **The midterm is TENTATIVELY scheduled for** Wednesday, February 21, 7-9pm. It is tentative because the registrar will not allow us to schedule classrooms until week 3 of the quarter.
- Any questions on the syllabus or logistics?

Rough class schedule

Date (2024)	Lecture	Content
08 Jan	1	Overview to deep learning
10 Jan	2	Machine learning refresher I
15 Jan	-	No class (MLK holiday)
		HW #1 released
17 Jan	3	Supervised Classification & Gradient descent principles
22 Jan	4	Fully connected neural networks
		HW #2 released
24 Jan	5	Backpropagation
29 Jan	6	Regularizations for training neural networks
		HW #3 released
31 Jan	7	Optimization for training neural networks
05 Feb	8	Convolutional Neural Networks I
		HW #4 released
07 Feb	9	Convolutional Neural Networks II
12 Feb	10	Convolutional Networks III
		HW #5 released
14 Feb	11	Recurrent Neural Networks
19 Feb	-	President's day holiday
21 Feb	M	Midterm (may be in the evening; may be Saturday)
		Project released
26 Feb	12	Recurrent neural networks II
28 Feb	13	PyTorch and Tensorflow
04 Mar	14	Transformers
06 Mar	15	Adversarial Attacks
11 Mar	16	Survey of advanced topics in deep learning
13 Mar	17	Overview



http://www.deeplearningbook.org/

Grading

Grading

You will be graded on three components:

1. **Homework (20%)**. Homework will contain both written components as well as `Python` components.
   - Assignments are due (i.e., submitted to Gradescope) by 11:59pm on the day they are due.
   - To accomodate for unexpected or unforeseen circumstances, we will give *three late days* to every student. These late days should only be used in extenuating circumstances. We will not grant additional late days beyond these.
   - You may use **at most** 2 late days on any given assignment.
   - Any assignment more than two days late receives a grade of **zero**.
2. **Midterm exam (50%)**, in class.
3. **Final project (30%)**, details to be released.

Midterm exam

- The midterm is in week 7, and is tentatively scheduled on Wednesday, Feb 21, 7-9p.
- We will still cancel class on Feb 21.
- You are allowed to bring 4 cheat sheets (each an 8.5 x 11 inch paper) to the exam. You can fill both sides (8 sides total). You can put whatever materials you want on it. It can be handwritten, typed, or a combination of the two. The exam is otherwise closed book, closed computer (except for a calculator), and closed internet.
- You may use pen or pencil on this exam. We will scan your exams after you take them.
- Other miscellaneous details students have asked: (1) you may be asked to write code on the exam, but this will not be used to test syntax; (2) while you should put commonly used gradients on your cheat sheets (including those we derived in class, or backpropagation rules), we will provide any gradients you would typically look up in the Matrix Cookbook.

On academic integrity

Academic integrity

UCLA embraces the core values of integrity, excellence, accountability, respect, and service through the True Bruin program

`http://www.truebruin.ucla.edu`

I take academic integrity very seriously; students caught cheating or violating these principles will face disciplinary action. Please refer to the UCLA student conduct code:

`https://deanofstudents.ucla.edu/student-conduct-code`

In this class, unacceptable behavior includes plagiarizing the work of others, plagiarizing code, and copying another person's exam. In accordance with UCLA policy, any instance of suspected academic dishonesty will be immediately reported to the Dean of Students Office and zero credit will be given for any work determined to be dishonest.

What I assume you know

Pre-requisites

This class requires a solid understanding in probability (131A) and linear algebra (133A or 205A), as well as prior exposure to machine learning (M146). This class will be very difficult (but not impossible) if you do not have prior machine learning background. We will spend two lectures doing machine learning review to ensure we are familiar with concepts we will expand upon in machine learning.

It also requires coding experience. The class will be taught entirely in `Python`. If you have only had exposure to `MATLAB`, there will be ramp up time to familiarize yourself with `Python`. You should factor this into your course load.

Pre-requisite topics I will **assume** you know.

- Probability: independence, conditional probability, Bayes rule, multivariate Gaussian distribution, marginalization, expectation, variance
- Linear algebra: basic matrix operations, span, rank, range and null space, eigenvalue decomposition, singular value decomposition, pseudoinverse

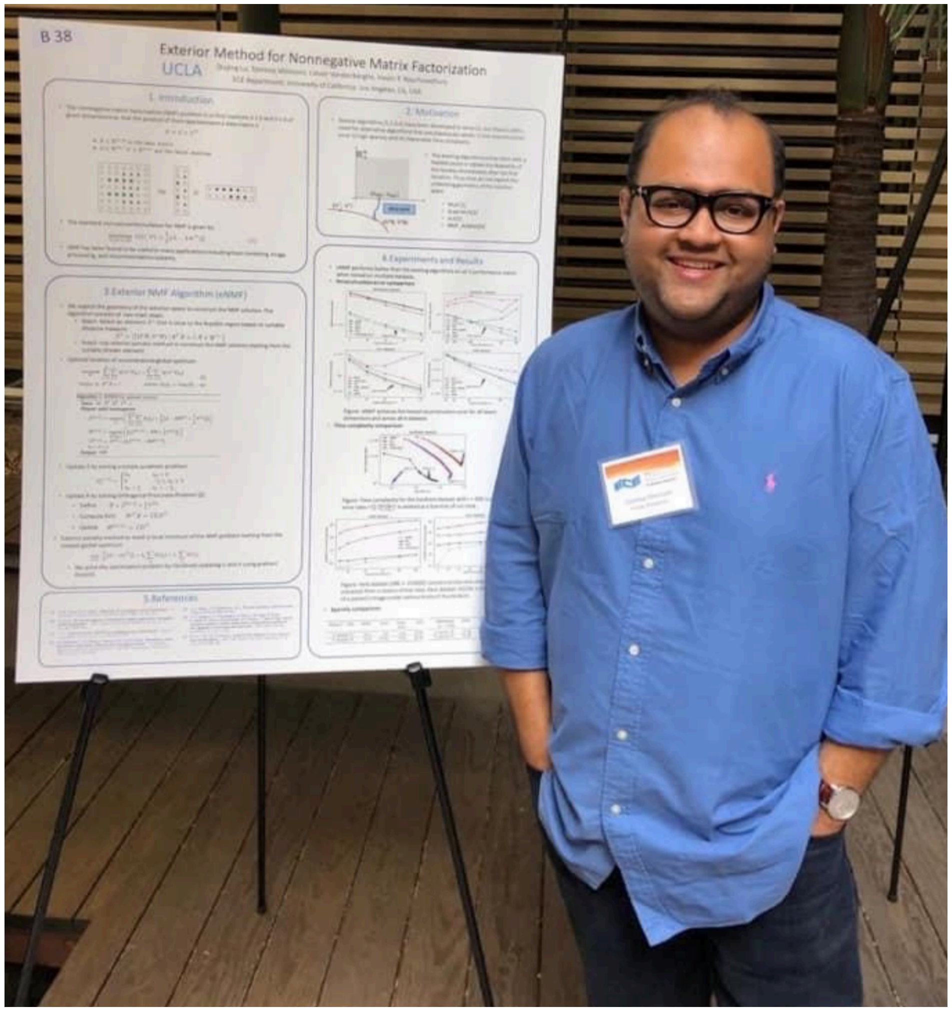
A few last notes about this class

- Common student feedback is that, even if they were familiar with `MATLAB`, it was still time-consuming to transition to `Python`. Please consider this seriously as you plan your schedule for assignments. `Python` is the standard language for machine learning research today, and the best deep learning packages are specifically designed for `Python`.
- We know, and consistently receive feedback, that this class is a lot of work and is time-consuming. I want to state this up front so you can plan accordingly. We will aim to keep the stated HW schedule, following the assignments and schedule used last year.

About me

- This is my seventh time teaching this class at UCLA.
- Though this class focuses on deep learning for engineering purposes, I'm interested in deep learning because of its ties to the brain.
  - I am primarily a computational neuroscientist and neural engineer.
  - My work includes brain-machine interfaces and applying machine learning to understand how neurons in the brain communicate.
  - We use deep learning as a model to understand the brain and build brain-machine interfaces.
- I normally teach ECE C143A/C243A in the Spring quarter, but this year I am developing a sequel to this course, "Neural Networks and Deep Learning 2" tentatively scheduled for Spring 2024 as ECE 239AS.

Tonmoy Monsoor



- Graduate student researcher at Big data and Complex Networks Group
- Fourth year Ph.D. student advised by Professor Vwani Roychowdhury
- Research interests
  - Reinforcement learning
  - Distributed optimization
  - Pattern extraction in dynamic networks
- Favorite classes at UCLA
  - Convex optimization, ECE 236B
  - Neural signal processing, ECE 243A

Yang Liu

- Third-year graduate student in electrical and computer engineering
- Aharoni lab in Neurology
- Tennis, climbing, and snowboarding

Shreyas Rajesh

- I'm a MS/PhD student in the ECE department advised by Prof. Vwani Roychowdhury. My research interests are in Natural Language Processing and Deep Learning.
- This is my 5th quarter TA'ing at UCLA and I have throughly enjoyed teaching thus far. I'm looking forward to a lot of learning and interesting discussions with all of you, please feel free to reach out with any questions you may have.
- Outside of work, I enjoy reading (my favorite writer is Mohsin Hamid) and watching and playing soccer (I'm a huge fan of Arsenal Football Club!)

Lahari Julakanti

- I am a second year MS student in the ECE department specializing in Signals and Systems. My field of interest is Digital and Wireless Communications.
- In my leisure time, I enjoy reading novels and comics.
- My favorite classes at UCLA have been 247 Neural Networks and Deep Learning, 231E Channel Coding Theory.

Lecture 2: Machine learning refresher

This lecture gives a refresher on key concepts from machine learning.

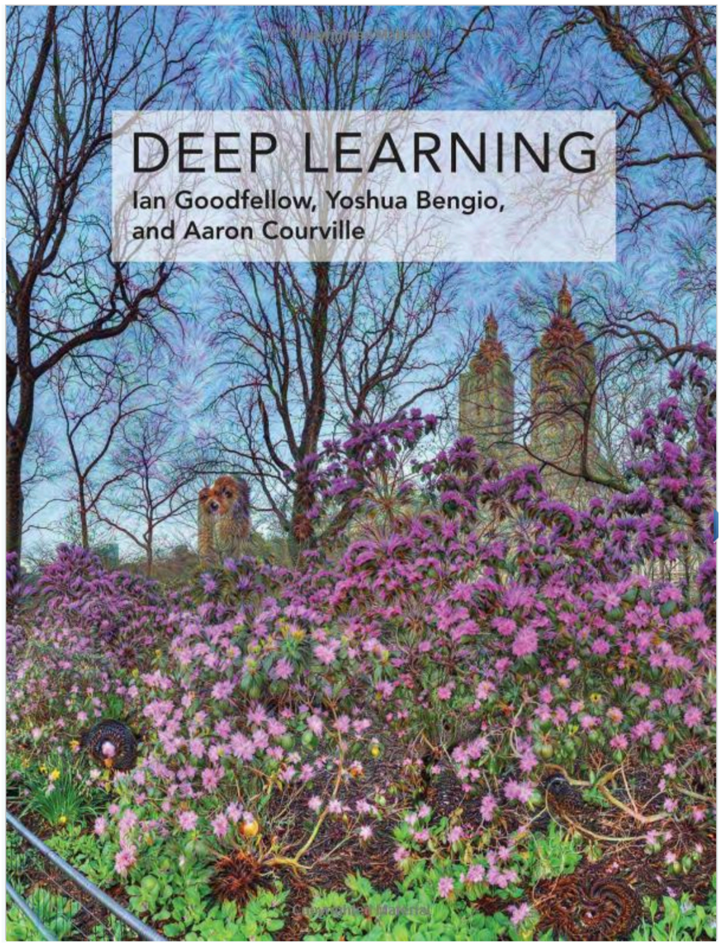
- Introduction to concepts in machine learning
- Cost functions
- Example: linear and polynomial regression
- Model complexity and overfitting
- Training set, validation set, test set
- Dealing with probabilistic cost functions and models
- Example: maximum-likelihood classification

Lecture 2: Basics of machine learning

Reading:

Deep Learning, chapter 5 (up to and including section 5.5).

To refresh your linear algebra and probability, look at chapters 2 and 3 respectively.



Focus of this class:

$x^{(i)}, y^{(i)}$

- This class will focus on **supervised** learning problems.

- This class will also focus on **classification** largely (e.g., when considering convolutional neural networks) as well as some instances of **regression** (e.g., when considering recurrent neural networks)

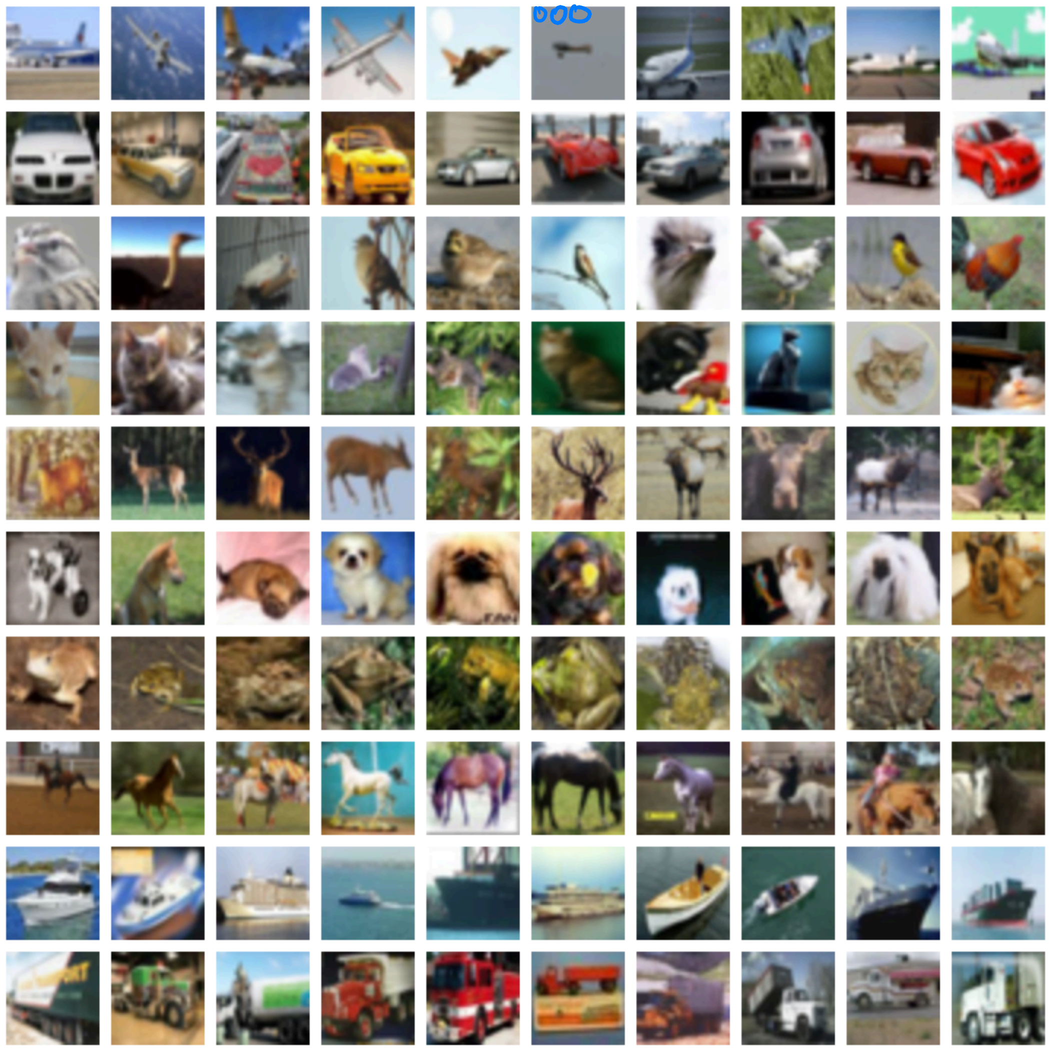
$y^{(i)}$

Supervised **Classification** Color : r, g, b

CIFAR-10

10 classes

airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck



$\vec{x}$

$32 \times 32 \times 3$

$\vec{x} \in \mathbb{R}^{3072}$

$\Rightarrow$ classification

class $\leftarrow f(\vec{x})$

$\downarrow$

what class the image belongs to

1 out of 10

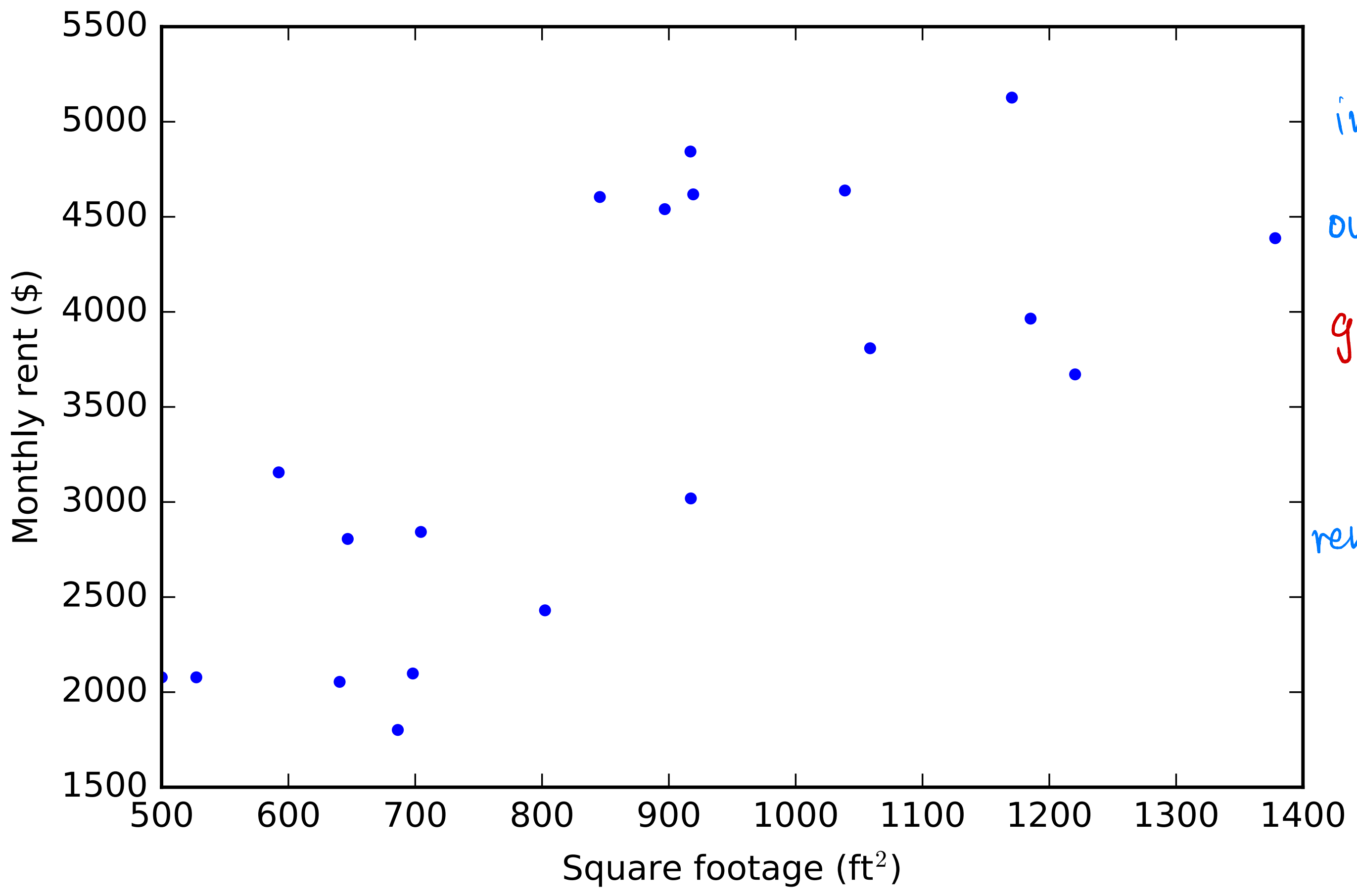
CIFAR-10 dataset, https://www.cs.toronto.edu/~kriz/cifar.html

softmax classifier

Regression

An example of supervised learning

Let's say we want to rent a home in Westwood, and we wanted to know if we were getting a good deal. **(Warning: this data is synthetic! Scrape the real data if you're curious.)**



input: sq. ft.

output: rent

goal: learn f

rent $\leftarrow f(\text{sq. ft.})$

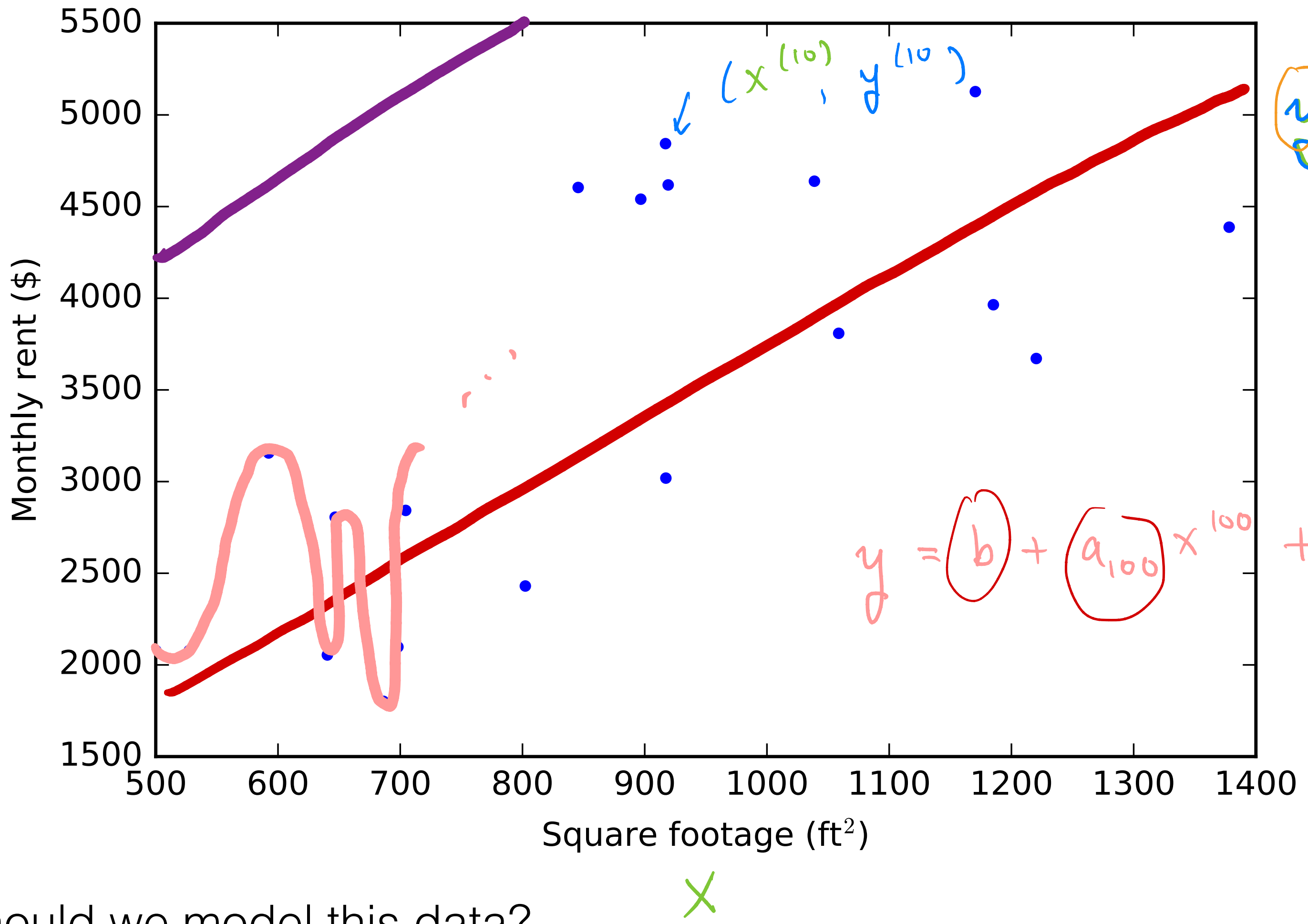
An example of supervised learning

$f(x) = ax + b$

parameters

$\boxed{y} = \boxed{a}\,\boxed{x} + \boxed{b}$

data given to me



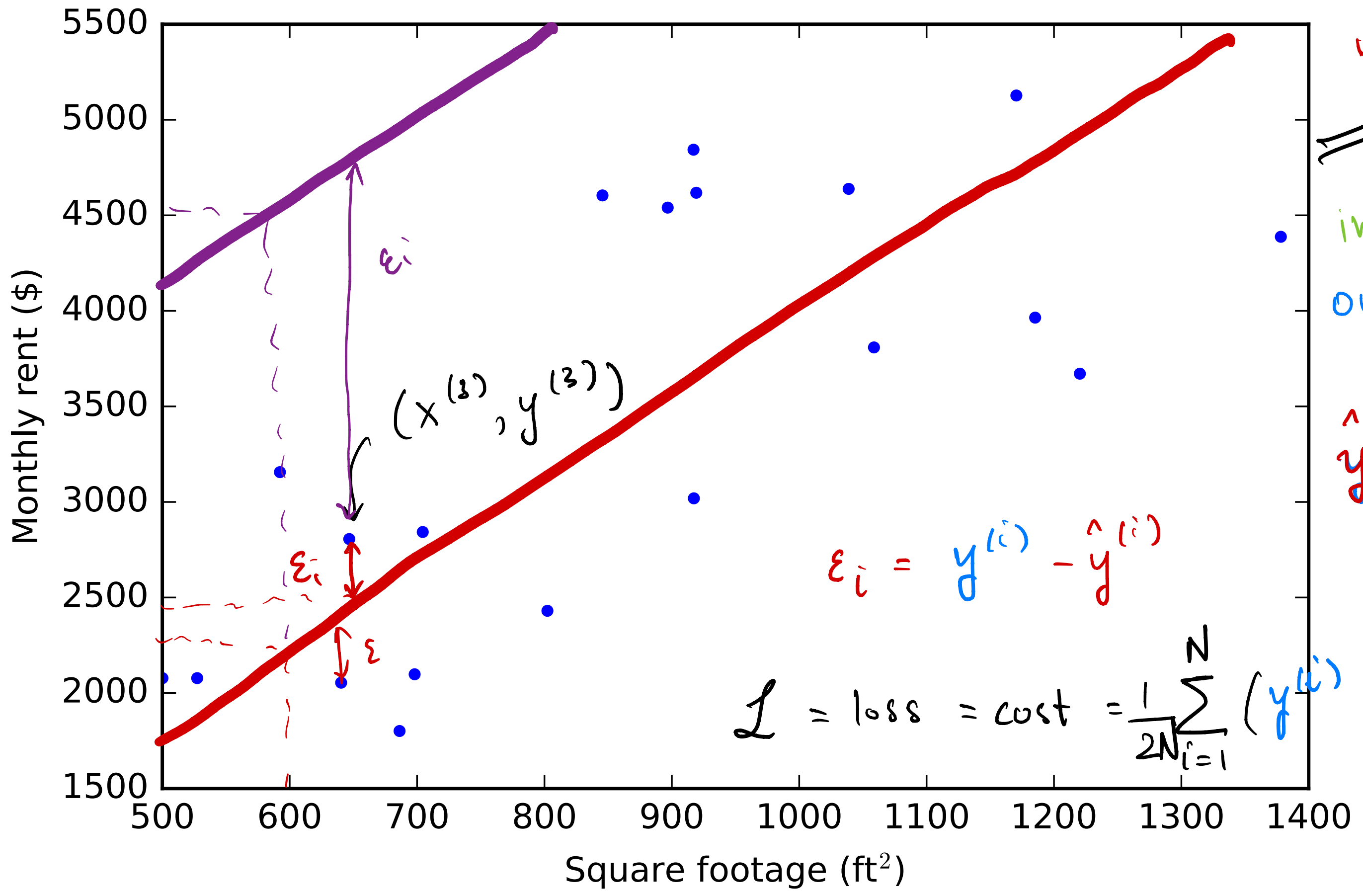
$y = \boxed{b} + \boxed{a_{100}}x^{100} + \boxed{a_{99}}x^{99} + \ldots + \boxed{a_1}x$

How should we model this data?

- Inputs, **x**? Outputs, **y**?
- What model should we use? f will be NN.
- How do we assess how good our model is?

An example of supervised learning

$y = NN(x)$



$y = b + ax$

inputs: $x^{(i)}$

output: $y^{(i)}$

$\hat{y}^{(i)} = b + a x^{(i)}$

$\varepsilon_i = y^{(i)} - \hat{y}^{(i)}$

$$\mathcal{L} = \text{loss} = \text{cost} = \frac{1}{2N} \sum_{i=1}^{N} \left(y^{(i)} - \hat{y}^{(i)} \right)^2$$

How should we model this data?

- Inputs, **x**? Outputs, **y**?
- What model should we use?
- How do we assess how good our model is?

An example of supervised learning

data given to us

A simple linear example:

output input

parameters: I get to choose the values of a and b to make the model as good as poss.

- Model:

$$\underbrace{\overset{y}{\hat{y}} = ax + b}$$
$$= \theta^T \hat{\mathbf{x}}$$

$\theta = \begin{bmatrix} a \\ b \end{bmatrix}$ $\hat{x} = \begin{bmatrix} x \\ 1 \end{bmatrix}$

$\downarrow$

to make $\mathcal{L}(a,b)$ as small as poss.

- Cost function:

$$\mathcal{L}(\theta) = \frac{1}{2}\sum_{i=1}^{N}(y^{(i)} - \hat{y}^{(i)})^2$$
$$= \frac{1}{2}\sum_{i=1}^{N}\left(y^{(i)} - \theta^T\hat{\mathbf{x}}^{(i)}\right)^2$$

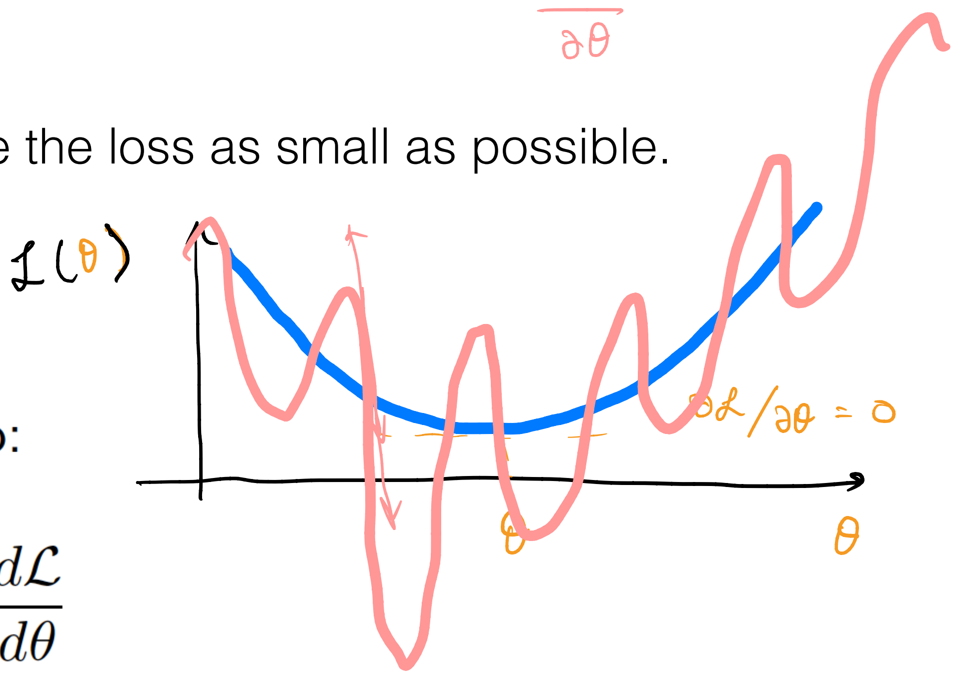
How do we learn the parameters of this model?

An example of supervised learning

Construct it as an optimization problem.

$\frac{\partial \mathcal{L}}{\partial \theta}$

Our goal is to choose the parameters to make the loss as small as possible.



Thus our strategy to find the best θ is to:

- Calculate

$$\frac{d\mathcal{L}}{d\theta}$$

- Solve for θ such that

$$\frac{\partial \mathcal{L}}{\partial \theta} = 0$$

However, θ is a vector, so how do we take derivatives with respect to it?

Aside: vector and matrix derivatives

In machine learning, we often take derivatives with respect to vectors and matrices.

These are typically called *gradients*, and in this class we'll use the following notation to denote derivatives with respect to vectors and matrices.

Differentiate a scaler y

w.r.t. a vector x

In this class, we will use both the differentiation operator ∂ and ∇ to denote derivatives. If y is a scalar, and $\mathbf{x}$ is a vector, then the gradient of y with respect to $\mathbf{x}$ is denoted as both:

$$\frac{\partial y}{\partial \mathbf{x}} \text{ and } \nabla_{\mathbf{x}} y$$

The gradient of y with respect to $\mathbf{x}$ is itself a vector with the same dimensionality as $\mathbf{x}$.

Aside: vector and matrix derivatives

y scalar

x vector $\in \mathbb{R}^n$

$\frac{\partial y}{\partial x}$ or $\nabla_x y \in \mathbb{R}^n$

The gradient

The gradient generalizes the scalar derivative to multiple dimensions. Suppose $f : \mathbb{R}^n \to \mathbb{R}$ transforms a vector $\mathbf{x} \in \mathbb{R}^n$ to a scalar. If $y = f(\mathbf{x})$, then the gradient is:

$$\nabla_{\mathbf{x}} y = \begin{bmatrix} \frac{\partial y}{\partial x_1} \\ \frac{\partial y}{\partial x_2} \\ \vdots \\ \frac{\partial y}{\partial x_n} \end{bmatrix}$$

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \qquad \frac{\partial y}{\partial x} = \nabla_x y = \begin{bmatrix} 1 \\ 0.5 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$\frac{\partial y}{\partial x_1} = 1$$

$$\frac{\partial y}{\partial x_2} = 0.5$$

Δy due to Δx_1, is $\approx \frac{\partial y}{\partial x_1} \Delta x_1$

Aside: vector and matrix derivatives

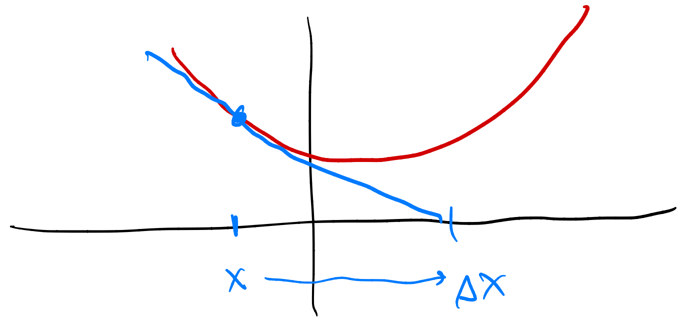
In other words, the gradient is:

- A *vector* that is the same size as $\mathbf{x}$, i.e., if $\mathbf{x} \in \mathbb{R}^n$ then $\nabla_{\mathbf{x}} y \in \mathbb{R}^n$.
- Each dimension of $\nabla_{\mathbf{x}} y$ tells us how small changes in $\mathbf{x}$ in that dimension affect y. i.e., changing the ith dimension of $\mathbf{x}$ by a small amount, Δx_i, will change y by

$$\frac{\partial y}{\partial x_i} \Delta x_i$$

We may also denote this as:

$$(\nabla_{\mathbf{x}} y)_i \, \Delta x_i$$



$$\Delta x = \begin{bmatrix} 0.05 \\ 0.01 \\ 0.02 \\ \vdots \end{bmatrix} \rightarrow \text{how much does } y \text{ change? } (\Delta y)$$

$$\Delta y \approx \sum_{i=1}^{n} \frac{\partial y}{\partial x_i} \Delta x_i$$

$$\Delta y \approx (\nabla_x y)^T \Delta x$$

Aside: vector and matrix derivatives

$\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix}$ $x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$

Example: derivative with respect to a vector

Let $f(\mathbf{x}) = \theta^T \mathbf{x}$. What is $\underline{\nabla_{\mathbf{x}} f(\mathbf{x})}$?

$\theta, x \in \mathbb{R}^n$

$y = \theta^T x$

$y \in \mathbb{R}$

$\nabla_x y \in \mathbb{R}^n$

$$y = \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

$$\nabla_x y = \begin{bmatrix} \partial y / \partial x_1 \\ \partial y / \partial x_2 \\ \vdots \\ \partial y / \partial x_n \end{bmatrix} = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix} = \theta$$

$$\boxed{\nabla_x y = \theta} \qquad \boxed{\nabla_\theta y = x}$$

Aside: vector and matrix derivatives

Example: derivative with respect to a vector

Let $f(\mathbf{x}) = \mathbf{x}^T \mathbf{A}\mathbf{x}$. What is $\nabla_{\mathbf{x}} f(\mathbf{x})$?

$(1\times n)\ (n\times n)\ (n \times 1)$

$x \in \mathbb{R}^n, \quad A \in \mathbb{R}^{n\times n}$

$f(x) \in \mathbb{R}, \quad \nabla_x f(x) \in \mathbb{R}^n$

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{n1} & \cdots & & a_{nn} \end{bmatrix}$$

$$f(x) = x^T A x = \sum_{i=1}^{n} \sum_{j=1}^{n} x_i \cdot a_{ij} \cdot x_j$$

$$\frac{\partial f(x)}{\partial x_1} = 2a_{11}x_1 + \sum_{j=2}^{n} a_{1j}x_j + \sum_{i=2}^{n} a_{i1}\cdot x_i$$

$$= \sum_{j=1}^{n} a_{1j}x_j + \sum_{i=1}^{n} a_{i1}x_i$$

$i=1,\ j=1$

$$\frac{\partial\, a_{11}x_1^2}{\partial x_1} = 2a_{11}x_1$$

$i=1,\ j\neq 1$

$$\frac{\partial\left(\sum_{j=2}^{n} x_1 \cdot a_{1j}x_j\right)}{\partial x_1} =$$

$i\neq 1,\ j=1$

$$\frac{\partial\left(\sum_{i=2}^{n} x_i \cdot a_{i1}x_1\right)}{\partial x_1}$$

Aside: vector and matrix derivatives

$$\frac{\partial f(x)}{\partial x_1} = \sum_{j=1}^{n} a_{1j} x_j + \sum_{i=1}^{n} a_{i1} x_i$$

$$\underset{z}{\begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix}} = \underset{A}{\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{n1} & \cdots & \cdots & a_{nn} \end{bmatrix}} \underset{x}{\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}}$$

$$(Ax)_1 + (A^T x)_1$$

$$\boxed{D_x f(x) = \begin{bmatrix} \frac{\partial f(x)}{\partial x_1} \\ \frac{\partial f(x)}{\partial x_2} \\ \vdots \\ \frac{\partial f(x)}{\partial x_n} \end{bmatrix} = \begin{bmatrix} (Ax)_1 + (A^T x)_1 \\ (Ax)_2 + (A^T x)_2 \\ \vdots \\ (Ax)_n + (A^T x)_n \end{bmatrix}}$$

$\frac{\partial f(x)}{\partial x_2}, \frac{\partial f(x)}{\partial x_3}, \ldots,$ all follow the same pattern.

$$\Longrightarrow \quad D_x f(x) = Ax + A^T x$$

$$= (A + A^T) x$$

if A is symmetric

$$= \underset{(n\times n)(n\times 1)}{2Ax}$$

when $n=1$

$$f(x) = x \cdot a \cdot x = ax^2$$

$$\frac{\partial f(x)}{\partial x} = 2ax$$

Aside: vector and matrix derivatives

First, we note that $\mathbf{x}^T\mathbf{A}\mathbf{x} = \sum_i \sum_j a_{ij}x_ix_j$. Then, we have

$$\begin{aligned}\nabla_{\mathbf{x}} f(\mathbf{x}) &= \begin{bmatrix} 2a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n + a_{21}x_2 + \cdots + a_{n1}x_n \\ 2a_{22}x_2 + a_{21}x_1 + \cdots + a_{2n}x_n + a_{12}x_1 + \cdots + a_{n2}x_n \\ \vdots \\ 2a_{nn}x_n + a_{n1}x_1 + \cdots + a_{n,n-1}x_{n-1} + a_{1n}x_1 + \cdots + a_{n-1,n}x_{n-1} \end{bmatrix} \\ &= \mathbf{A}\mathbf{x} + \mathbf{A}^T\mathbf{x}\end{aligned}$$

Matrix derivatives

$y = z^T A x$ $\quad z \in \mathbb{R}^m$ $\quad A \in \mathbb{R}^{m \times n}$

$x \in \mathbb{R}^n$

$$\frac{\partial y}{\partial A} = D_A y = \begin{bmatrix} \frac{\partial y}{\partial a_{11}} & \frac{\partial y}{\partial a_{12}} & \cdots & \frac{\partial y}{\partial a_{1n}} \\ \vdots & & \ddots & \vdots \\ \frac{\partial y}{\partial a_{m1}} & & \cdots & \frac{\partial y}{\partial a_{mn}} \end{bmatrix}$$

$\frac{\partial y}{\partial x} \in \mathbb{R}^{n \times 1}$

$\frac{\partial y}{\partial A} \in \mathbb{R}^{m \times n}$

if $A \in \mathbb{R}^{m \times n}$

$D_A y \in \mathbb{R}^{m \times n}$

"Denominator layout"

$x \in \mathbb{R}^{n \times 1}$, $D_x y \in \mathbb{R}^{n \times 1}$

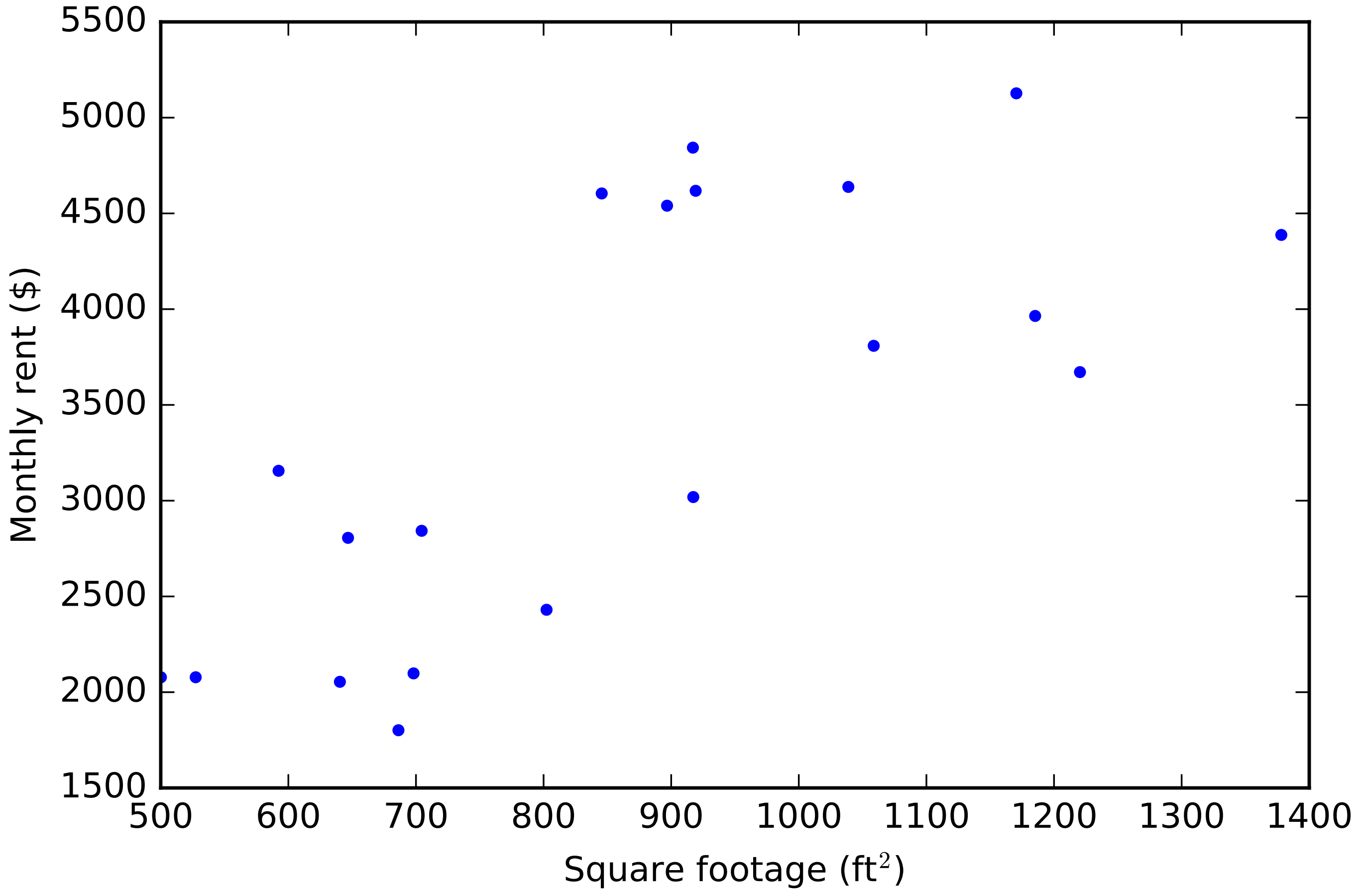
$A \in \mathbb{R}^{m \times n}$, $D_A y \in \mathbb{R}^{m \times n}$

"Numerator Layout"

$x \in \mathbb{R}^{n \times 1}$, $D_x y \in \mathbb{R}^{1 \times n}$

$A \in \mathbb{R}^{m \times n}$, $D_A y \in \mathbb{R}^{n \times m}$

Back to our supervised learning example



Thus our strategy to find the best θ is to:

- Calculate

$$\frac{d\mathcal{L}}{d\theta}$$

- Solve for θ such that

$$\frac{\partial\mathcal{L}}{\partial\theta} = 0$$

Back to our supervised learning example

Re-writing the cost function, we have:

$$\mathcal{L} = \frac{1}{2}\sum_{i=1}^{N}(y^{(i)} - \theta^T\hat{\mathbf{x}}^{(i)})^2$$

$5^T = 5$

$(ABC)^T = C^T B^T A^T$

$$= \frac{1}{2}\sum_{i=1}^{N}(y^{(i)} - \theta^T\hat{x}^{(i)})^T(y^{(i)} - \theta^T\hat{x}^{(i)})$$

$$= \frac{1}{2}\sum_{i=1}^{N}(y^{(i)} - \hat{x}^{(i)T}\theta)^T(y^{(i)} - \hat{x}^{(i)T}\theta)$$

" vectorization "

HW #2

$$= \frac{1}{2}\left(\underbrace{\begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}}_{Y} - \underbrace{\begin{bmatrix} \hat{x}^{(1)T} \\ \hat{x}^{(2)T} \\ \vdots \\ \hat{x}^{(N)T} \end{bmatrix}}_{\mathbb{X}} \underset{\theta}{\theta}\right)^T \left(\begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix} - \begin{bmatrix} \hat{x}^{(1)T} \\ \hat{x}^{(2)T} \\ \vdots \\ \hat{x}^{(N)T} \end{bmatrix}\theta\right)$$