UCLA Samueli Computer Science

# CS145 Discussion: Week 2
# Linear Regression & Logistic Regression

Junheng Hao

Friday, 10/16/2020

## Roadmap

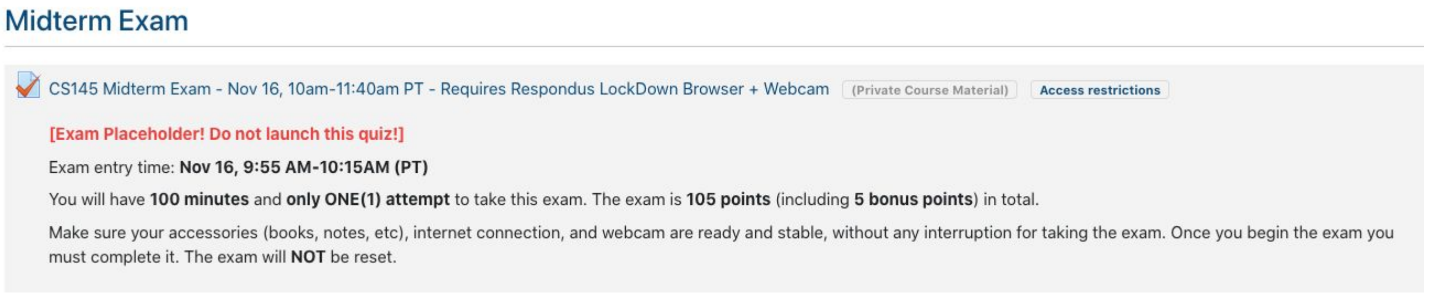
- Announcements
- Linear Regression
- Logistic Regression
- Project Tips & Programming Resource
- Q & A (Office hour)

## Announcements

- Register the team by the end of **Week 2 (Fri 16 Oct, 2020)**, including Name, UID, email and the group leader.
  - Team ID will be assigned later (early Week 3)
  - Signup Sheet: [Link]
- Homework 1 due on **Oct 19, 2020 11:59 PT**
  - Submit through GradeScope of 1 PDF (2 python file and 1 jupyter notebook into 1 PDF file)
- Homework 2 will be expected to release on Wednesday, Oct 21 (Week 3)
  - Tasks: Decision tree and SVM

# Time Zone Accomodation of Exams

- Requires Respondus LockDown Browser + Webcam
  - No external internet access, no exit to other browser window, no screenshot during the exam
  - No reviews and scores immediately after the exam
  - Once you begin the exam you must complete it. The exam will NOT be reset.
  - Submissions that fails to use LockDown Browser will not be credits
- Practice quiz is available for browser setup
- Two exam time slots for midterm and final exam: Enter either one of them within the entry time window



Midterm Exam

CS145 Midterm Exam - Nov 16, 10am-11:40am PT - Requires Respondus LockDown Browser + Webcam (Private Course Material) Access restrictions

[Exam Placeholder! Do not launch this quiz!]

Exam entry time: **Nov 16, 9:55 AM-10:15AM (PT)**

You will have **100 minutes** and **only ONE(1) attempt** to take this exam. The exam is **105 points** (including **5 bonus points**) in total.

Make sure your accessories (books, notes, etc), internet connection, and webcam are ready and stable, without any interruption for taking the exam. Once you begin the exam you must complete it. The exam will **NOT** be reset.

## Linear Regression: Model

- Linear model to predict value of a variable *y* using features ***x***

$$y = \boldsymbol{x}^T \boldsymbol{\beta} = x_1\beta_1 + x_2\beta_2 + \cdots + x_p\beta_p$$

- Least Square Estimation

$$J(\boldsymbol{\beta}) = \frac{1}{2n}(X\boldsymbol{\beta} - y)^T (X\boldsymbol{\beta} - y)$$

- Closed form solution

$$\hat{\boldsymbol{\beta}} = (X^T X)^{-1} X^T y$$

## Linear Regression: Example

- A ball is rolled down a hallway and its position is recorded at five different times. Use the table shown below to calculate
  - Weights
  - Predicted position at each given time and at time 12 seconds

| Time (s) | Position (m) |
|---|---|
| 1 | 9 |
| 2 | 12 |
| 4 | 17 |
| 6 | 21 |
| 8 | 26 |

## Linear Regression: Example

**Step 1: Question**

- What are X and Y variables?
- What are the parameters for our problem?
- Calculating parameters

| **Time (s)** | **Position (m)** |
|---|---|
| 1 | 9 |
| 2 | 12 |
| 4 | 17 |
| 6 | 21 |
| 8 | 26 |

UCLA
Engineer Change.

## Linear Regression: Example

### Step 1: Calculate Weights

- What are X and Y variables?
  - Time (X) and Position(Y)
- What are the parameters for our problem?
  - $\hat{\beta}_1$ :Time $\hat{\beta}_0$ :Intercept
- Calculating parameters
  - $\hat{\boldsymbol{\beta}} = (X^T X)^{-1} X^T y$

| Time (s) | Position (m) |
|---|---|
| 1 | 9 |
| 2 | 12 |
| 4 | 17 |
| 6 | 21 |
| 8 | 26 |

## Linear Regression: Example

***Let's calculate on BOARD!***

$$X = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 4 \\ 1 & 6 \\ 1 & 8 \end{bmatrix} \qquad y = \begin{bmatrix} 9 \\ 12 \\ 17 \\ 21 \\ 26 \end{bmatrix}$$

$$X^T X = ? \qquad (X^T X)^{-1} = ?$$

$$X^T y = ? \qquad \hat{\beta} = (X^T X)^{-1} X^T y = ?$$

| Time (s) | Position (m) |
|---|---|
| 1 | 9 |
| 2 | 12 |
| 4 | 17 |
| 6 | 21 |
| 8 | 26 |

# Linear Regression: Example

## Step 2: Apply your model and predict

- Plug time values into linear regression equation

$$\hat{y} = 2.378x + 7.012$$

- Predicted value at time = 12 secs

$$\hat{y}(x = 12) = 2.378 \times 12 + 7.012 = 35.548$$

- Matrix form to predict all other positions

$$\hat{\boldsymbol{y}} = \boldsymbol{X}\hat{\beta}$$

| Time (s) | Position (m) |
|---|---|
| 1 | 9 |
| 2 | 12 |
| 4 | 17 |
| 6 | 21 |
| 8 | 26 |
| 12 | 35.55 |

## Linear Regression: Example

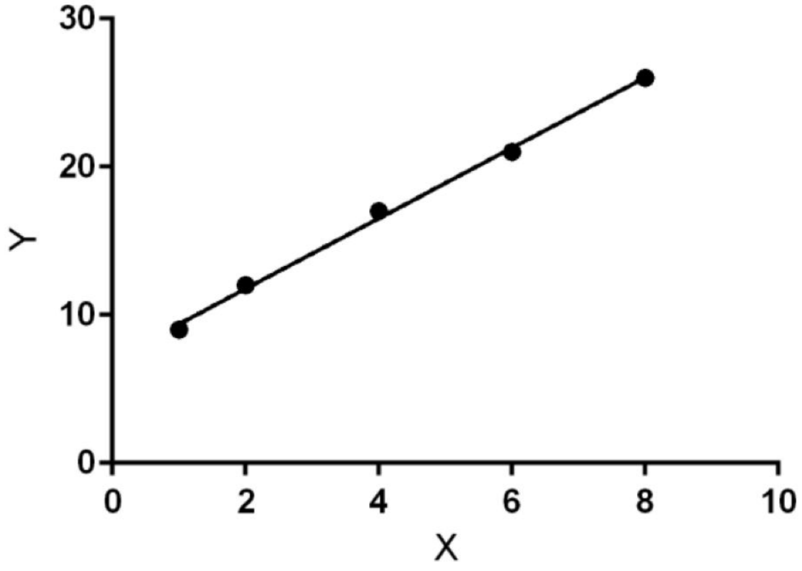
**Plot: Check your model**

$$\hat{y} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 4 \\ 1 & 6 \\ 1 & 8 \end{bmatrix} \begin{bmatrix} 7.012 \\ 2.378 \end{bmatrix} = \begin{bmatrix} 9.390 \\ 11.768 \\ 16.524 \\ 21.280 \\ 26.036 \end{bmatrix}$$

| Time (s) | Position (m) |
|---|---|
| 1 | 9 |
| 2 | 12 |
| 4 | 17 |
| 6 | 21 |
| 8 | 26 |

## Linear Regression: Example

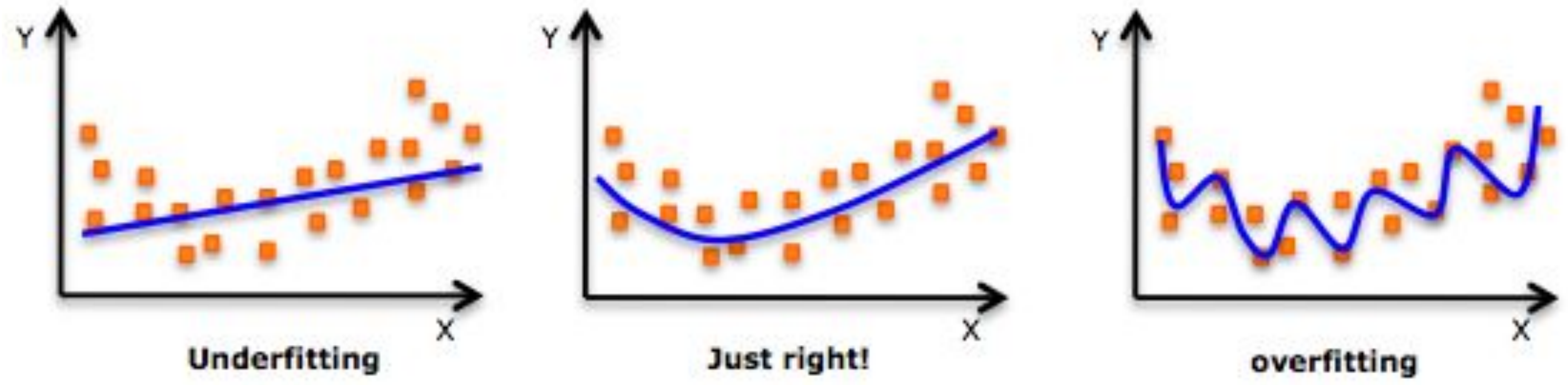
**Plot: Check your model**



| Time (s) | Position (m) |
|---|---|
| 1 | 9 |
| 2 | 12 |
| 4 | 17 |
| 6 | 21 |
| 8 | 26 |

# Linear Regression + Regularization

- Why do we need regularization?
  - Constraint the parameter values → Avoid overfitting phenomenon



# Bias vs Variance: Review

True predictor $f(x): x^T\boldsymbol{\beta}$

Estimated predictor $\hat{f}(x): x^T\widehat{\boldsymbol{\beta}}$

- Bias: $E(\hat{f}(x)) - f(x)$
  - How far away is the expectation of the estimator to the true value? The smaller the better.
- Variance: $Var\left(\hat{f}(x)\right) = E[\left(\hat{f}(x) - E\left(\hat{f}(x)\right)\right)^2]$
  - How variant is the estimator? The smaller the better.
- Reconsider mean square error
  - $J(\widehat{\boldsymbol{\beta}})/n = \sum_i \left(\boldsymbol{x}_i^T\widehat{\boldsymbol{\beta}} - y_i\right)^2/n$
  - Can be considered as
    - $E[\left(\hat{f}(x) - f(x) - \varepsilon\right)^2] = bias^2 + variance + noise$

Note $E(\varepsilon) = 0, Var(\varepsilon) = \sigma^2$

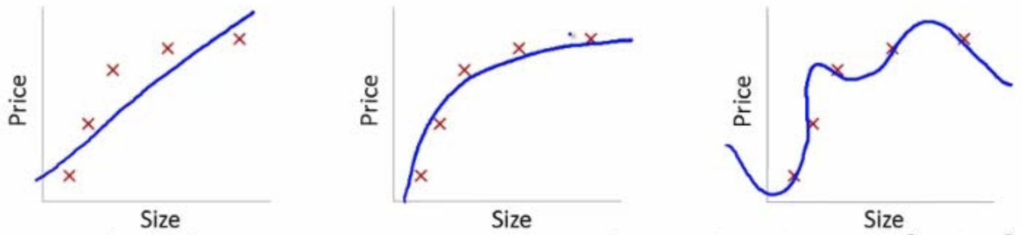
Reference: https://en.wikipedia.org/wiki/Bias%E2%80%93variance_tradeoff

# Bias vs Variance: Example

1. $h_\theta(x) = \theta_0 + \theta_1 x$
2. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$
3. $h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_3 x^3$

$\vdots$

10. $h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_{10} x^{10}$



$\theta_0 + \theta_1 x$ | $\theta_0 + \theta_1 x + \theta_2 x^2$ | $\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

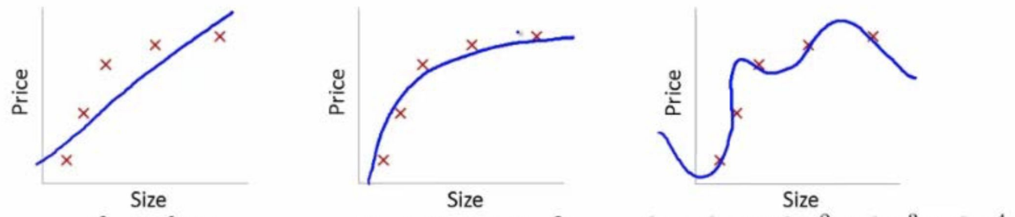
High bias (underfit) | "Just right" | High variance (overfit)

# Bias vs Variance: Example

1. $h_\theta(x) = \theta_0 + \theta_1 x$
2. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$
3. $h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_3 x^3$

$\vdots$

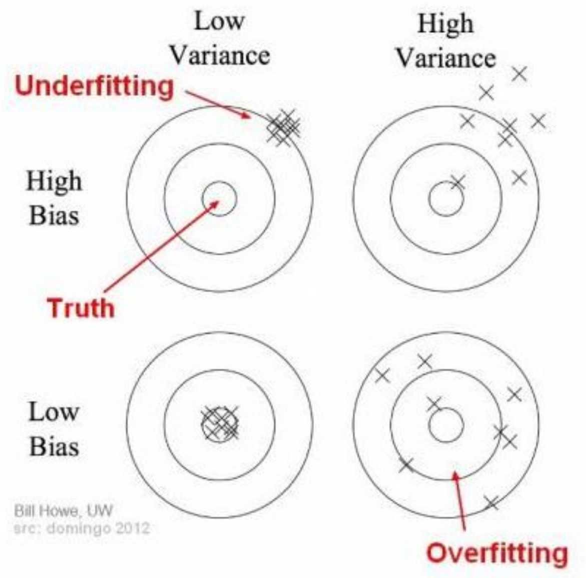
10. $h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_{10} x^{10}$



$\theta_0 + \theta_1 x$ — High bias (underfit)

$\theta_0 + \theta_1 x + \theta_2 x^2$ — "Just right"

$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$ — High variance (overfit)



# Closed form: LR + Regularization

- Model

$$\hat{y} = \boldsymbol{x}^T \boldsymbol{\beta} = x_1\beta_1 + x_2\beta_2 + \cdots + x_p\beta_p$$

- Original Objective

$$\min_{\boldsymbol{\beta}} \; J(\boldsymbol{\beta}) = \frac{1}{2}\sum_{i=1}^{N}(\boldsymbol{x}^T\boldsymbol{\beta} - y)^2$$

- L2-Regularized Objective

$$\min_{\boldsymbol{\beta}} \; J(\boldsymbol{\beta}) = \frac{1}{2}\sum_{i=1}^{N}(\boldsymbol{x}^T\boldsymbol{\beta} - y)^2 + \frac{\lambda}{2}||\boldsymbol{\beta}||^2$$

## Closed form: LR + Regularization

$$\boxed{\min_{\boldsymbol{\beta}} \; J(\boldsymbol{\beta}) = \frac{1}{2}\sum_{i=1}^{N}(\boldsymbol{x}^T\boldsymbol{\beta} - y)^2 + \frac{\lambda}{2}||\boldsymbol{\beta}||^2} \longrightarrow$$

$$\frac{\partial J(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = \sum_{i=1}^{N}\boldsymbol{x}(\boldsymbol{x}^T\boldsymbol{\beta} - y) + \frac{\partial \lambda \; ||\boldsymbol{\beta}||^2}{\partial \boldsymbol{\beta}}$$

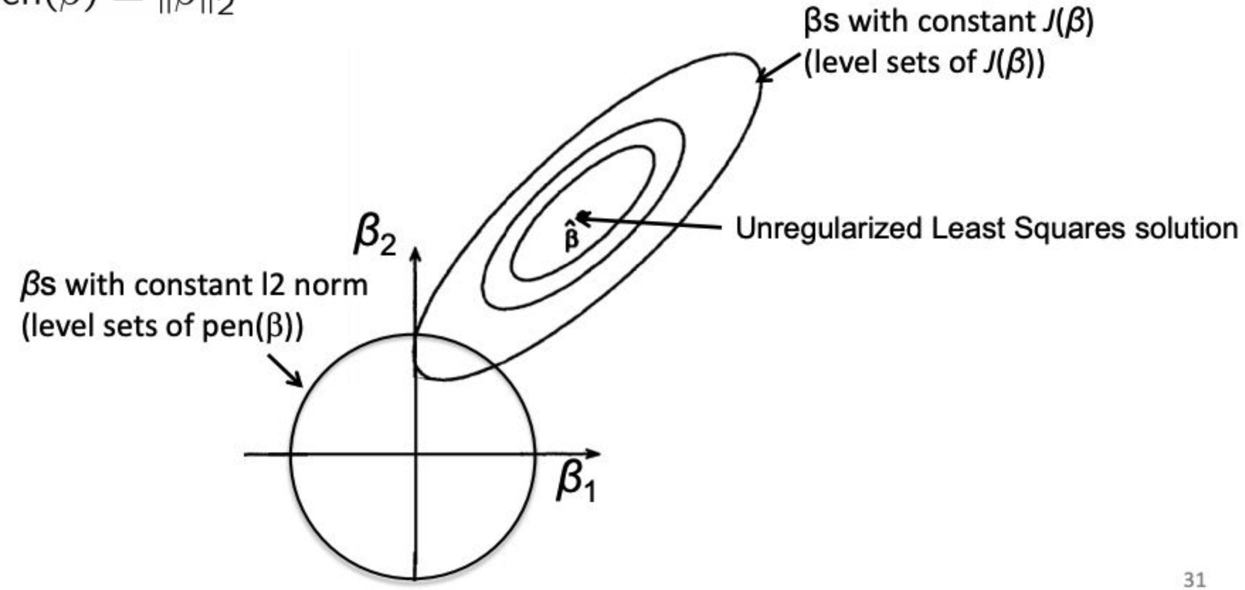
$$\frac{\partial J(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = \sum_{i=1}^{N}\boldsymbol{x}(\boldsymbol{x}^T\boldsymbol{\beta} - y) + \lambda\boldsymbol{\beta}$$

# Closed form: LR + Regularization

$$\min_{\beta}(\mathbf{A}\beta - \mathbf{Y})^T(\mathbf{A}\beta - \mathbf{Y}) + \lambda \text{pen}(\beta) = \min_{\beta} J(\beta) + \lambda \text{pen}(\beta)$$

**Ridge Regression:**

$\text{pen}(\beta) = \|\beta\|_2^2$



# Regularization

**Regularization methods**

- L2 normalization (Ridge)

$$\min_{\beta} \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij})^2 + \lambda \sum_{j=1}^{p} \beta_j^2$$

- L1 normalization (Lasso)

$$\min_{\beta} \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij})^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

- Dropout *(we will learn this in neural networks)*
- Early Stopping

## Z-Score Normalization

- Why normalize features？
  - Different feature ranges such as [-1, 1] and [-100, 100] may negatively affect algorithm performance.
  - Thus, small change in bigger range can affect more than huge change in smaller range.
- **Calculate Z-Score (Standard Score)**

$$z_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}$$

➢ $z_{ij}$ → the standard score for feature *j* of data point *i*

➢ $x_{ij}$ → the value of feature *j* of data point *i*

➢ $\mu_j, \sigma_j$ → the mean and standard deviation of feature *j*

# Z-Score Normalization: Example

| Galaxy | Distance (Mpc) | Velocity (km/sec) |
|---|---|---|
| Virgo | 15 | 1,600 |
| Ursa Minor | 200 | 15,000 |
| Corona Borealis | 290 | 24,000 |
| Bootes | 520 | 40,000 |

- How to Normalize feature “Distance” ?

# Z-Score Normalization: Example

| Galaxy | Distance (Mpc) | Velocity (km/sec) |
|---|---|---|
| Virgo | 15 | 1,600 |
| Ursa Minor | 200 | 15,000 |
| Corona Borealis | 290 | 24,000 |
| Bootes | 520 | 40,000 |

- How to Normalize feature “Distance”?
  - Computer mean and deviation first

$$\mu_{\text{dist}} = ? \quad \sigma_{\text{dist}} = ?$$

# Z-Score Normalization: Example

| Galaxy | Distance (Mpc) | Velocity (km/sec) |
|---|---|---|
| Virgo | 15 | 1,600 |
| Ursa Minor | 200 | 15,000 |
| Corona Borealis | 290 | 24,000 |
| Bootes | 520 | 40,000 |

- How to Normalize feature “Distance”?
  - Computer mean and deviation $\boxed{\mu_{\text{dist}} = 256.25, \sigma_{\text{dist}} = 181.71}$
  - Computer Z-Score for each sample

$$z_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}$$

# Z-Score Normalization: Example

| Galaxy | Distance (Mpc) | Velocity (km/sec) |
|---|---|---|
| Virgo | 15 | 1,600 |
| Ursa Minor | 200 | 15,000 |
| Corona Borealis | 290 | 24,000 |
| Bootes | 520 | 40,000 |

- How to Normalize feature “Distance”?
  - Computer mean and deviation $\mu_{\text{dist}} = 256.25, \sigma_{\text{dist}} = 181.71$
  - Computer Z-Score for each sample (for example “Virgo”)

$$\boxed{z_{\text{virgo,dist}} = \frac{x_{\text{virgo,dist}} - \mu_{\text{dist}}}{\sigma_{\text{dist}}} = \frac{15 - 256.25}{181.71} = -1.328}$$

## Newton’s Method

From lectures:

$$\beta^{\text{new}} = \beta^{\text{old}} - \left(\frac{\partial^2 L(\beta)}{\partial\beta\partial\beta^T}\right)^{-1} \frac{\partial L(\beta)}{\partial\beta}$$

Apply Newton’s methods on single variable to find minima:

$$x_{n+1} = x_n - \frac{f'(x_n)}{f''(x_n)}$$

From single variable to Multivariate Newton-Raphson Method

# Newton's Method: Steps

1. Initialize $x^{(0)}$
2. Calculate $\nabla f(x)$
3. Calculate $F(x)$
4. Initialize step $n = 0$ and start loops
   a. Calculate $\nabla f(x^{(n)})$
   b. Calculate $F(x^{(n)})$
   c. Calculate $[F(x^{(n)})]^{-1}$
   d. Update: $x^{(n+1)} = x^{(n)} - [F(x^{(n)})]^{-1} \cdot \nabla f(x^{(n)})$
   e. Update: $n = n + 1$
5. Exit Loop

## Newton's Method: Example

$$x^{(0)} = [3, -1, 0]$$

$$f(x_1, x_2, x_3) = (x_1 + 10x_2)^2 + 5(x_1 - x_3)^2 + (x_2 - 2x_3)^4$$

***Again, let's calculate on BOARD!***

## Newton's Method: Solution

$$x^{(0)} = [3, -1, 0]$$

$$f(x_1, x_2, x_3) = (x_1 + 10x_2)^2 + 5(x_1 - x_3)^2 + (x_2 - 2x_3)^4$$

***Newton's Method Example in one step:***

- ➔ Calculate $\nabla f(x^{(n)})$
- ➔ Calculate $F(x^{(n)})$
- ➔ Calculate $[F(x^{(n)})]^{-1}$
- ➔ Update: $x^{(n+1)}$
- ➔ Update: $n = n + 1$

$$\nabla f(x^{(0)}) = \left[\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \frac{\partial f}{\partial x_3}\right] = [16, -144, 22]$$

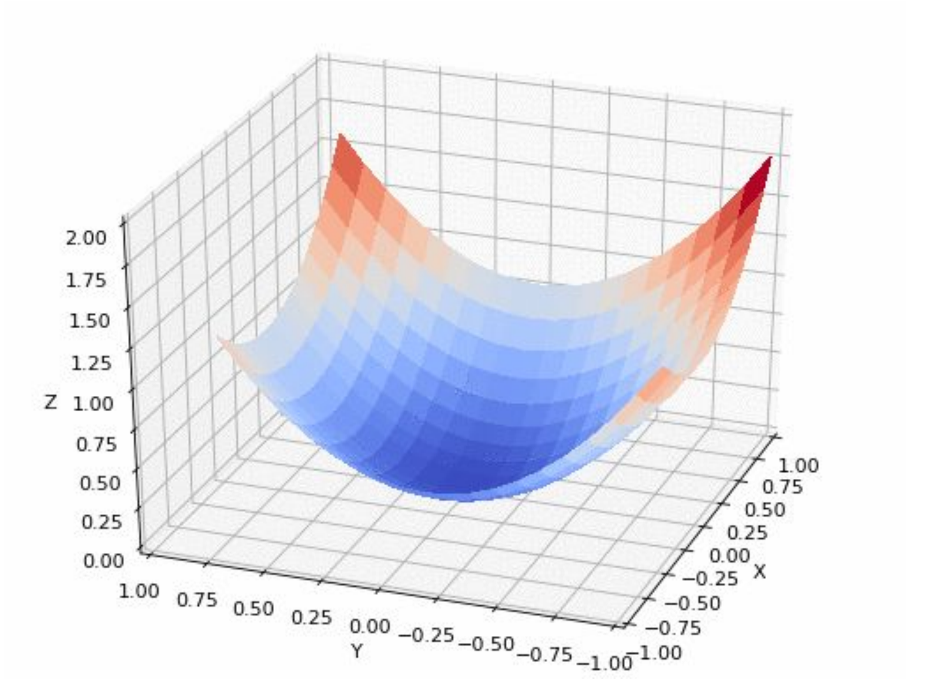
$$F(x^{(0)}) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \frac{\partial^2 f}{\partial x_1 \partial x_3} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \frac{\partial^2 f}{\partial x_2 \partial x_3} \\ \frac{\partial^2 f}{\partial x_3 \partial x_1} & \frac{\partial^2 f}{\partial x_3 \partial x_2} & \frac{\partial^2 f}{\partial x_3^2} \end{bmatrix} = \begin{bmatrix} 12 & 20 & -10 \\ 20 & 22 & -24 \\ -10 & -24 & 48 \end{bmatrix}$$

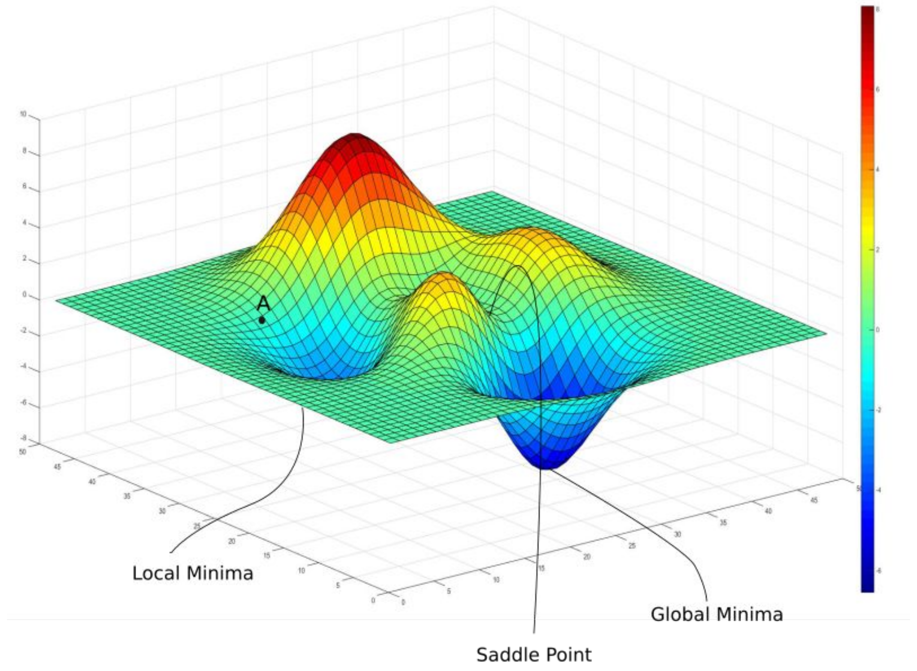
$$[F(x^{(0)})]^{-1} = \begin{bmatrix} -0.079 & 0.119 & 0.043 \\ 0.119 & -0.079 & -0.015 \\ 0.043 & -0.015 & 0.023 \end{bmatrix}$$

$$x^{(1)} = x^{(0)} - [F(x^{(0)})]^{-1} \cdot \nabla f(x^{(0)})$$

$$n = 1$$

# Gradient Descent

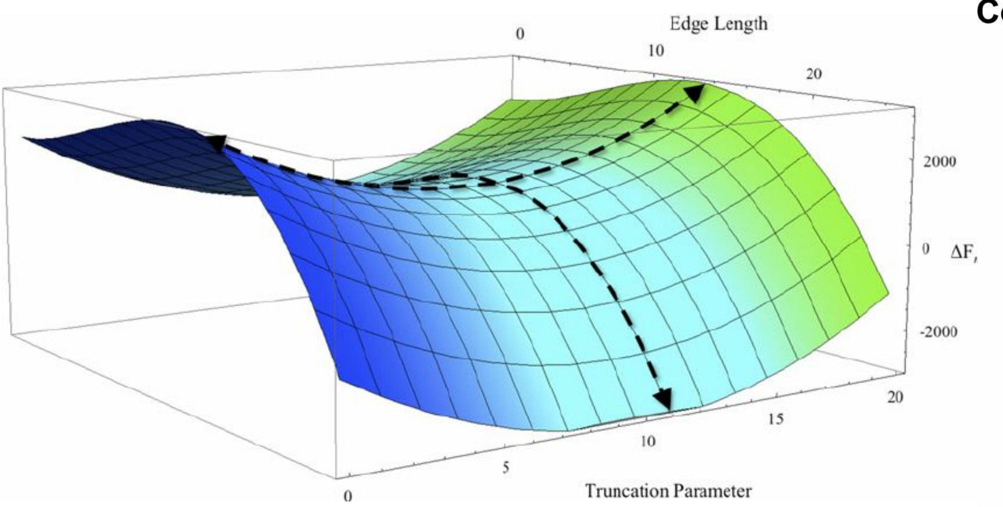


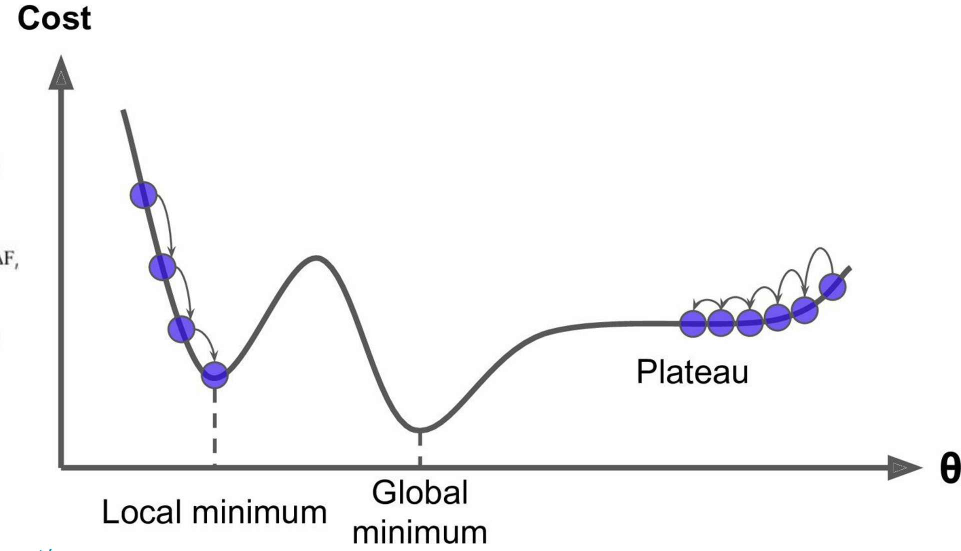


## Gradient Descent: Batch vs Stochastic

Why do we need Stochastic GD besides the efficiency/scalability reason?

Think about saddle points and local minima.



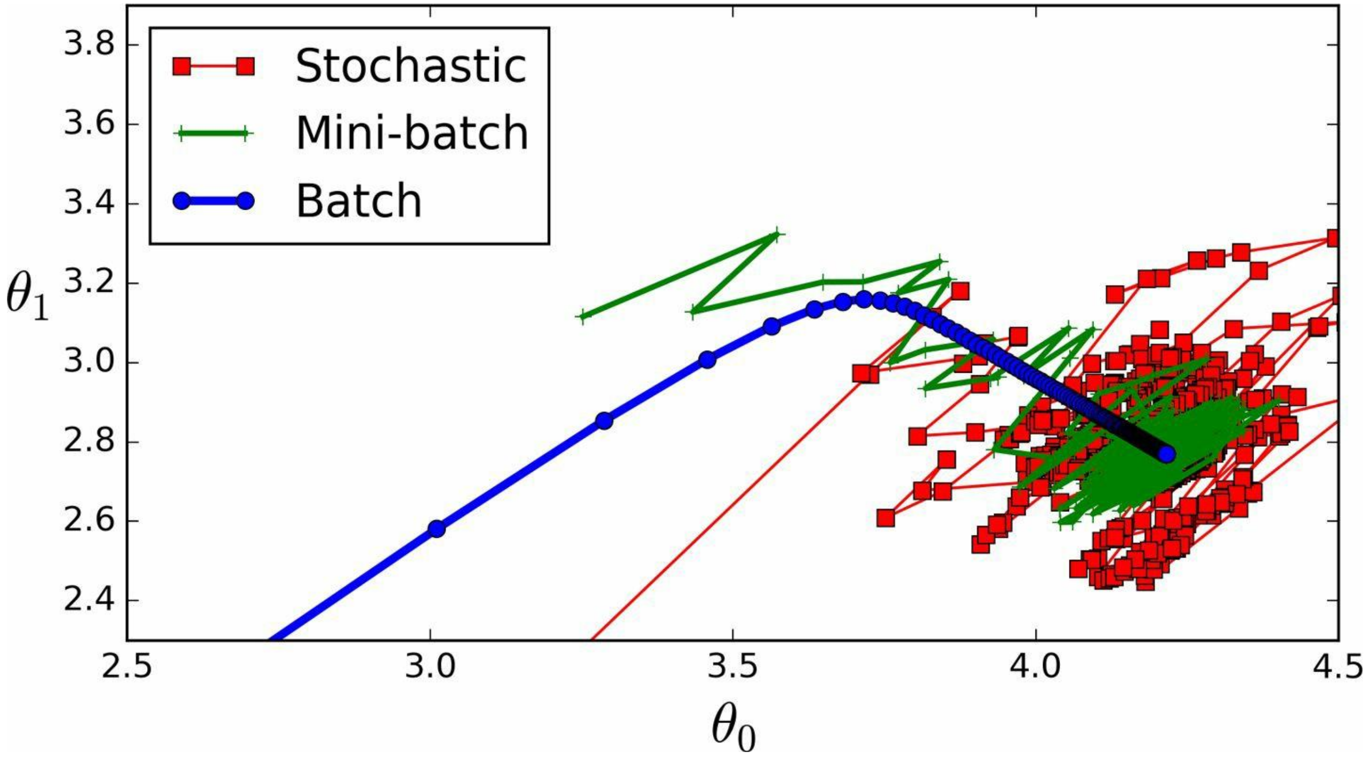


https://blog.paperspace.com/intro-to-optimization-in-deep-learning-gradient-descent/

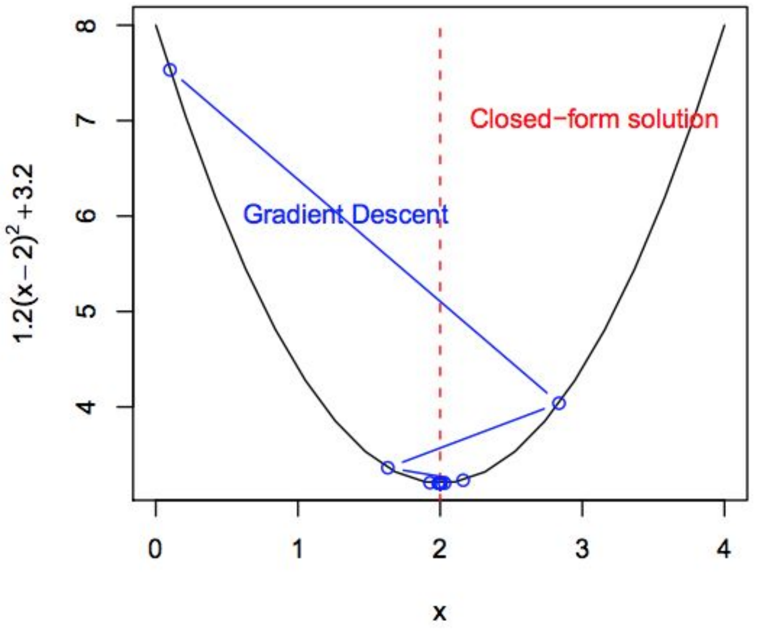
Géron, Aurélien. *Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems*. " O'Reilly Media, Inc.", 2017.

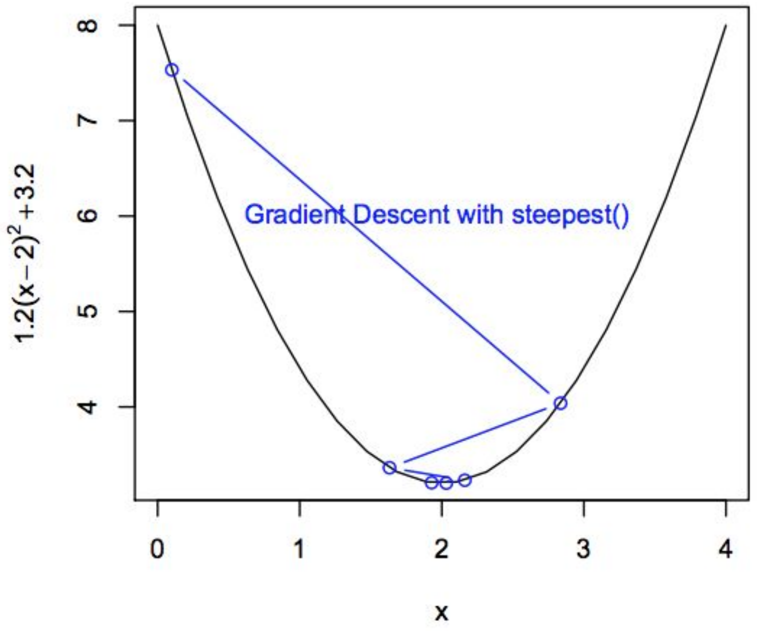
# Comparison

Batch vs Stochastic vs Mini-batch



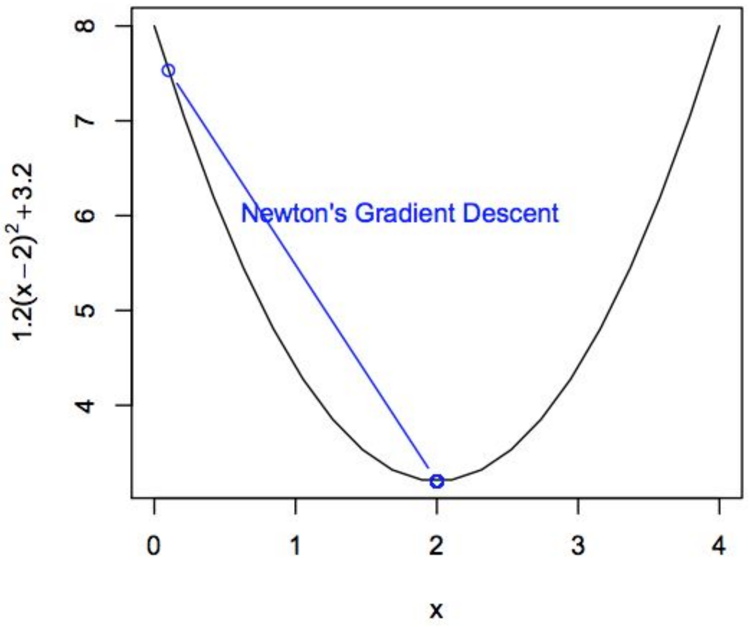
# Gradient Descent or Newton's Method?

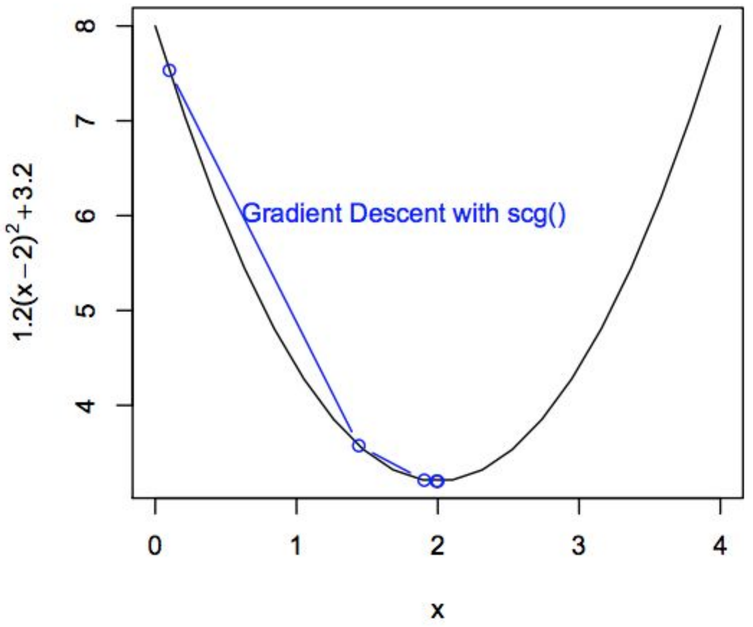




https://stats.stackexchange.com/questions/253632/why-is-newtons-method-not-widely-used-in-machine-learning
https://math.stackexchange.com/questions/609680/newtons-method-intuition
https://thelaziestprogrammer.com/sharrington/math-of-machine-learning/solving-logreg-newtons-method
https://web.archive.org/web/20151122203025/http://www.cs.colostate.edu/~anderson/cs545/Lectures/week6day2/week6day2.pdf

# Gradient Descent or Newton's Method?





https://stats.stackexchange.com/questions/253632/why-is-newtons-method-not-widely-used-in-machine-learning
https://math.stackexchange.com/questions/609680/newtons-method-intuition
https://thelaziestprogrammer.com/sharrington/math-of-machine-learning/solving-logreg-newtons-method
https://web.archive.org/web/20151122203025/http://www.cs.colostate.edu/~anderson/cs545/Lectures/week6day2/week6day2.pdf

## Logistic Regression: Example Question

We are given a data set consisting of the following experiment. Well, the dataset is a little bit small. (O_o)

The height and weight of 3 people were recorded at the beginning of each person's 65th birthday. At exactly one year after each person's 65th birthday the vital status was recorded to be either alive or deceased.

Our end goal is to use logistic regression to predict the probability that a person's life expectancy is at least 66 years given their age of 65, initial vital status of alive, height, and weight (but we won't go that far here).

The data is given in the following table on the right.

| **Height (inches)** | **Weight (lbs)** | **Vital Status** |
|---|---|---|
| 60 | 155 | Deceased |
| 64 | 135 | Alive |
| 73 | 170 | Alive |

## Logistic Regression: Example Question

**Step 1: State the log-likelihood function.**

| Height (inches) | Weight (lbs) | Vital Status |
|---|---|---|
| 60 | 155 | Deceased |
| 64 | 135 | Alive |
| 73 | 170 | Alive |

# Logistic Regression: Example Question

**Step 1: State the log-likelihood function.**

Answer:

$$\alpha_1 = -b - 155w_1 - 60w_2$$

$$\alpha_2 = -b - 135w_1 - 64w_2$$

$$\alpha_3 = -b - 170w_1 - 73w_2$$

| Height (inches) | Weight (lbs) | Vital Status |
|---|---|---|
| 60 | 155 | Deceased |
| 64 | 135 | Alive |
| 73 | 170 | Alive |

$$L = log\left(1 - \frac{1}{1+e^{\alpha_1}}\right) + log\left(\frac{1}{1+e^{\alpha_2}}\right) + log\left(\frac{1}{1+e^{\alpha_3}}\right)$$

# Logistic Regression: Example Question

**Step 2: State the gradients for each parameter.**

| Height (inches) | Weight (lbs) | Vital Status |
|---|---|---|
| 60 | 155 | Deceased |
| 64 | 135 | Alive |
| 73 | 170 | Alive |

# Logistic Regression: Example Question

**Step 2: State the gradients for each parameter.**

Answer:

$$L = log\left(1 - \frac{1}{1+e^{\alpha_1}}\right) + log\left(\frac{1}{1+e^{\alpha_2}}\right) + log\left(\frac{1}{1+e^{\alpha_3}}\right)$$

$$\nabla_b = -1.0 \cdot \frac{1}{1+e^{\alpha_1}} + -1.0 \cdot -\frac{e^{\alpha_2}}{1+e^{\alpha_2}} + -1.0 \cdot -\frac{e^{\alpha_3}}{1+e^{\alpha_3}}$$

$$\nabla_{w_1} = -155.0 \cdot \frac{1}{1+e^{\alpha_1}} + -135.0 \cdot -\frac{e^{\alpha_2}}{1+e^{\alpha_2}} + -170.0 \cdot -\frac{e^{\alpha_3}}{1+e^{\alpha_3}}$$

$$\nabla_{w_2} = -60 \cdot \frac{1}{1+e^{\alpha_1}} + -64.0 \cdot -\frac{e^{\alpha_2}}{1+e^{\alpha_2}} + -73.0 \cdot -\frac{e^{\alpha_3}}{1+e^{\alpha_3}}$$

| Height (inches) | Weight (lbs) | Vital Status |
|---|---|---|
| 60 | 155 | Deceased |
| 64 | 135 | Alive |
| 73 | 170 | Alive |

$$\alpha_1 = -b - 155w_1 - 60w_2$$

$$\alpha_2 = -b - 135w_1 - 64w_2$$

$$\alpha_3 = -b - 170w_1 - 73w_2$$

## Logistic Regression: Example Question

**Step 3: Give the Hessian Matrix**

| Height (inches) | Weight (lbs) | Vital Status |
|---|---|---|
| 60 | 155 | Deceased |
| 64 | 135 | Alive |
| 73 | 170 | Alive |

## Logistic Regression: Example Question

**Step 3: Give the Hessian Matrix**

$$\nabla_b = -1.0 \cdot \frac{1}{1+\mathrm{e}^{\alpha_1}} + -1.0 \cdot -\frac{\mathrm{e}^{\alpha_2}}{1+\mathrm{e}^{\alpha_2}} + -1.0 \cdot -\frac{\mathrm{e}^{\alpha_3}}{1+\mathrm{e}^{\alpha_3}}$$

$$\nabla_{w_1} = -155.0 \cdot \frac{1}{1+\mathrm{e}^{\alpha_1}} + -135.0 \cdot -\frac{\mathrm{e}^{\alpha_2}}{1+\mathrm{e}^{\alpha_2}} + -170.0 \cdot -\frac{\mathrm{e}^{\alpha_3}}{1+\mathrm{e}^{\alpha_3}}$$

$$\nabla_{w_2} = -60 \cdot \frac{1}{1+\mathrm{e}^{\alpha_1}} + -64.0 \cdot -\frac{\mathrm{e}^{\alpha_2}}{1+\mathrm{e}^{\alpha_2}} + -73.0 \cdot -\frac{\mathrm{e}^{\alpha_3}}{1+\mathrm{e}^{\alpha_3}}$$

$$H_b^T = \begin{bmatrix} -1.0 \cdot -1.0 \cdot -\frac{\mathrm{e}^{\alpha_1}}{(1+\mathrm{e}^{\alpha_1})^2} + -1.0 \cdot -1.0 \cdot -\frac{\mathrm{e}^{\alpha_2}}{(1+\mathrm{e}^{\alpha_2})^2} + -1.0 \cdot -1.0 \cdot -\frac{\mathrm{e}^{\alpha_3}}{(1+\mathrm{e}^{\alpha_3})^2} \\ -155.0 \cdot -1.0 \cdot -\frac{\mathrm{e}^{\alpha_1}}{(1+\mathrm{e}^{\alpha_1})^2} + -135.0 \cdot -1.0 \cdot -\frac{\mathrm{e}^{\alpha_2}}{(1+\mathrm{e}^{\alpha_2})^2} + -170.0 \cdot -1.0 \cdot -\frac{\mathrm{e}^{\alpha_3}}{(1+\mathrm{e}^{\alpha_3})^2} \\ -60 \cdot -1.0 \cdot -\frac{\mathrm{e}^{\alpha_1}}{(1+\mathrm{e}^{\alpha_1})^2} + -64.0 \cdot -1.0 \cdot -\frac{\mathrm{e}^{\alpha_2}}{(1+\mathrm{e}^{\alpha_2})^2} + -73.0 \cdot -1.0 \cdot -\frac{\mathrm{e}^{\alpha_3}}{(1+\mathrm{e}^{\alpha_3})^2} \end{bmatrix}$$

$$H_{w_1}^T = \begin{bmatrix} -1.0 \cdot -155.0 \cdot -\frac{\mathrm{e}^{\alpha_1}}{(1+\mathrm{e}^{\alpha_1})^2} + -1.0 \cdot -135.0 \cdot -\frac{\mathrm{e}^{\alpha_2}}{(1+\mathrm{e}^{\alpha_2})^2} + -1.0 \cdot -170.0 \cdot -\frac{\mathrm{e}^{\alpha_3}}{(1+\mathrm{e}^{\alpha_3})^2} \\ -155.0 \cdot -155.0 \cdot -\frac{\mathrm{e}^{\alpha_1}}{(1+\mathrm{e}^{\alpha_1})^2} + -135.0 \cdot -135.0 \cdot -\frac{\mathrm{e}^{\alpha_2}}{(1+\mathrm{e}^{\alpha_2})^2} + -170.0 \cdot -170.0 \cdot -\frac{\mathrm{e}^{\alpha_3}}{(1+\mathrm{e}^{\alpha_3})^2} \\ -60 \cdot -155.0 \cdot -\frac{\mathrm{e}^{\alpha_1}}{(1+\mathrm{e}^{\alpha_1})^2} + -64.0 \cdot -135.0 \cdot -\frac{\mathrm{e}^{\alpha_2}}{(1+\mathrm{e}^{\alpha_2})^2} + -73.0 \cdot -170.0 \cdot -\frac{\mathrm{e}^{\alpha_3}}{(1+\mathrm{e}^{\alpha_3})^2} \end{bmatrix}$$

$$H_{w_2}^T = \begin{bmatrix} -1.0 \cdot -60 \cdot -\frac{\mathrm{e}^{\alpha_1}}{(1+\mathrm{e}^{\alpha_1})^2} + -1.0 \cdot -64.0 \cdot -\frac{\mathrm{e}^{\alpha_2}}{(1+\mathrm{e}^{\alpha_2})^2} + -1.0 \cdot -73.0 \cdot -\frac{\mathrm{e}^{\alpha_3}}{(1+\mathrm{e}^{\alpha_3})^2} \\ -155.0 \cdot -60 \cdot -\frac{\mathrm{e}^{\alpha_1}}{(1+\mathrm{e}^{\alpha_1})^2} + -135.0 \cdot -64.0 \cdot -\frac{\mathrm{e}^{\alpha_2}}{(1+\mathrm{e}^{\alpha_2})^2} + -170.0 \cdot -73.0 \cdot -\frac{\mathrm{e}^{\alpha_3}}{(1+\mathrm{e}^{\alpha_3})^2} \\ -60 \cdot -60 \cdot -\frac{\mathrm{e}^{\alpha_1}}{(1+\mathrm{e}^{\alpha_1})^2} + -64.0 \cdot -64.0 \cdot -\frac{\mathrm{e}^{\alpha_2}}{(1+\mathrm{e}^{\alpha_2})^2} + -73.0 \cdot -73.0 \cdot -\frac{\mathrm{e}^{\alpha_3}}{(1+\mathrm{e}^{\alpha_3})^2} \end{bmatrix}$$

# Logistic Regression: Example Question

**Step 4: Assuming an initial guess of 0.25 for each parameter, write python code for finding the values of the parameters after 2 iterations using the Newton Raphson method.**

$$b = 1.1346728128592689$$

$$w_1 = -2.4878423877892759$$

$$w_2 = 3.8192554544178936$$

| Height (inches) | Weight (lbs) | Vital Status |
|---|---|---|
| 60 | 155 | Deceased |
| 64 | 135 | Alive |
| 73 | 170 | Alive |

# Closed form: LR + Regularization

- Model

$$y = \sigma(X) = \frac{1}{1 + e^{-X^T\beta}}$$

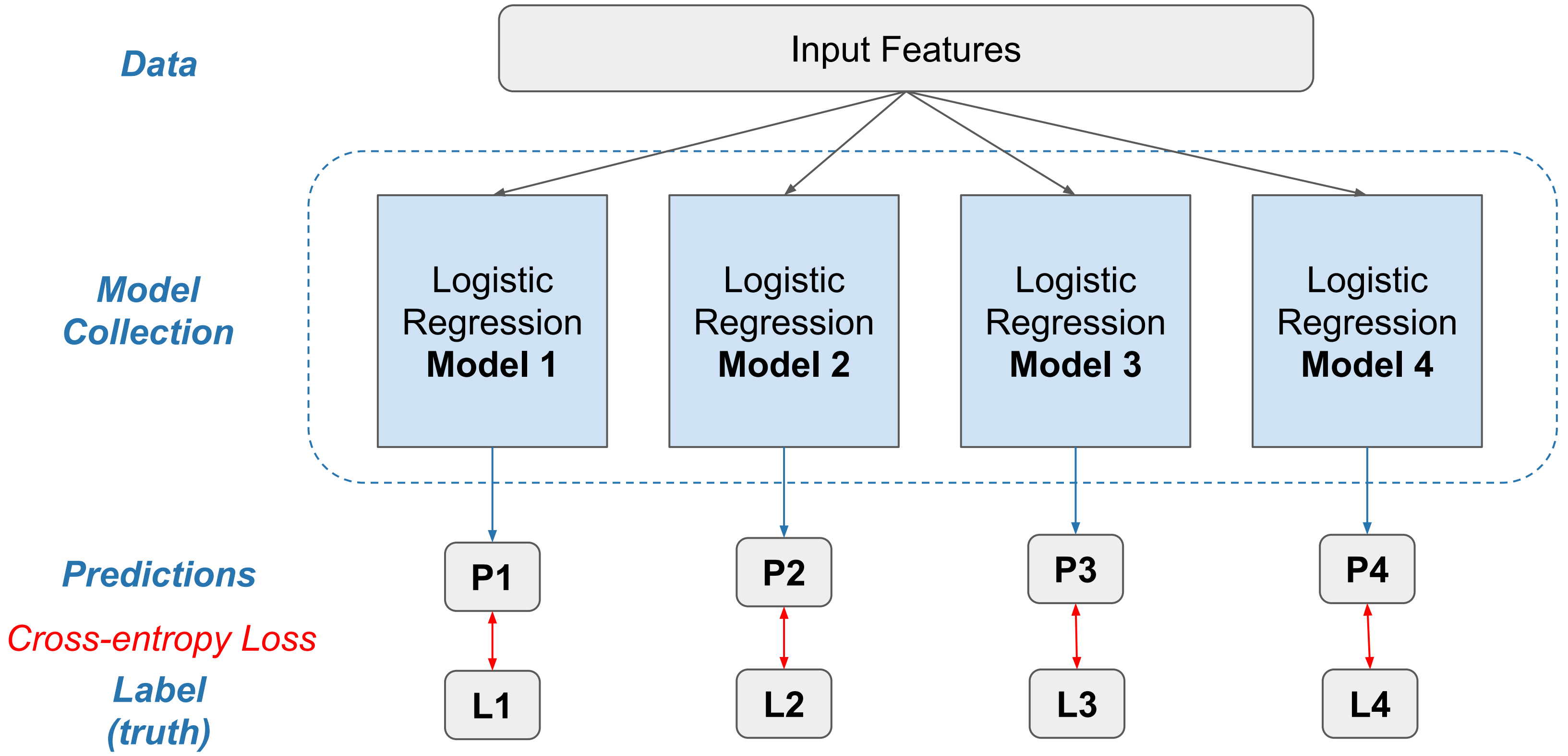
- Original Objective

$$J(\beta) = -\frac{1}{n}\sum_i \left(y_i x_i^T\beta - \log\left(1 + \exp\{x_i^T\beta\}\right)\right)$$

- L2-Regularized Objective

$$J(\beta) = -\frac{1}{n}\sum_i \left(y_i x_i^T\beta - \log\left(1 + \exp\{x_i^T\beta\}\right)\right) + \lambda\sum_j \beta_j^2$$

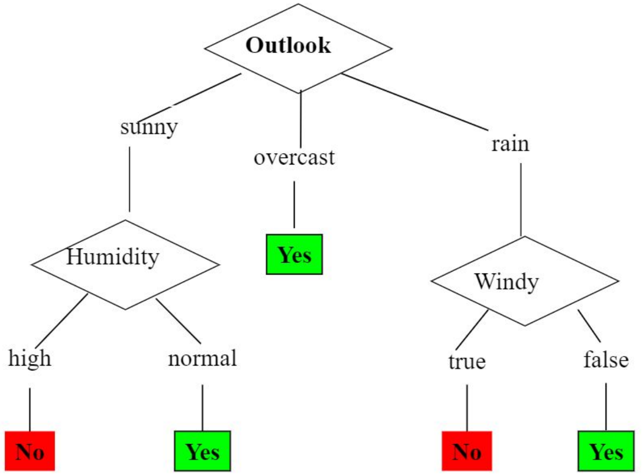
# Logistic Regression: Multiclass Case



Data
Input Features
Model Collection
Logistic Regression Model 1
Logistic Regression Model 2
Logistic Regression Model 3
Logistic Regression Model 4
Predictions
P1
P2
P3
P4
Cross-entropy Loss
Label (truth)
L1
L2
L3
L4

# Decision Tree for Classification

| Outlook | Temperature | Humidity | Windy | Play? |
|---|---|---|---|---|
| sunny | hot | high | false | No |
| sunny | hot | high | true | No |
| overcast | hot | high | false | Yes |
| rain | mild | high | false | Yes |
| rain | cool | normal | false | Yes |
| rain | cool | normal | true | No |
| overcast | cool | normal | true | Yes |
| sunny | mild | high | false | No |
| sunny | cool | normal | false | Yes |
| rain | mild | normal | false | Yes |
| sunny | mild | normal | true | Yes |
| overcast | mild | high | true | Yes |
| overcast | hot | normal | false | Yes |
| rain | mild | high | true | No |



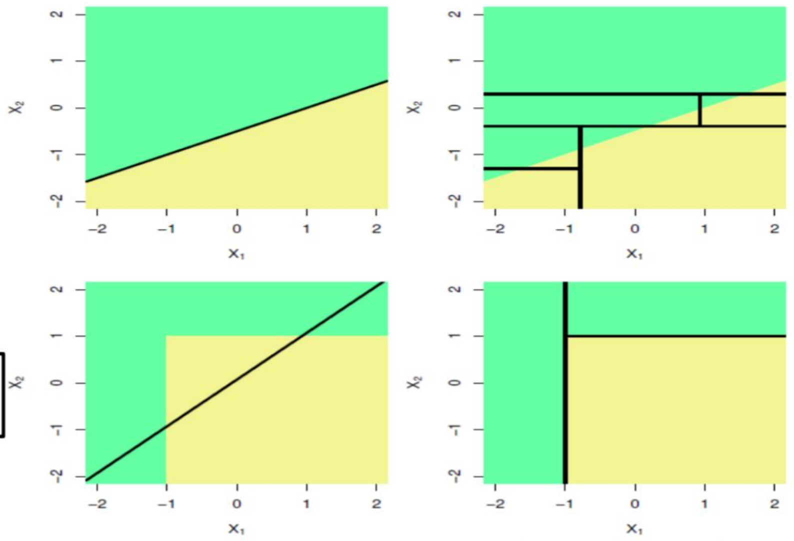
UCLA
Engineer Change.

## Decision Boundary

- Comparison: Logistic Regression vs Decision Tree

Ground Truth: Linear Boundary

Ground Truth: Non-Linear Boundary



Fitted Model: Linear Model

Fitted Model: Trees

# Project: Important Dates

- **Oct.18**: Group formation due (link)
- **Nov. 9**: Midterm project report due
- **Dec.10**: Kaggle Submission Due (release new data for Output2 around a week before)
- **Dec.18**: Final project report due (together with all codes)

*Note that the deadlines are subject to change according to the class schedule (avoid other deadlines of homework and exams).*

## Tips about COVID-19 project

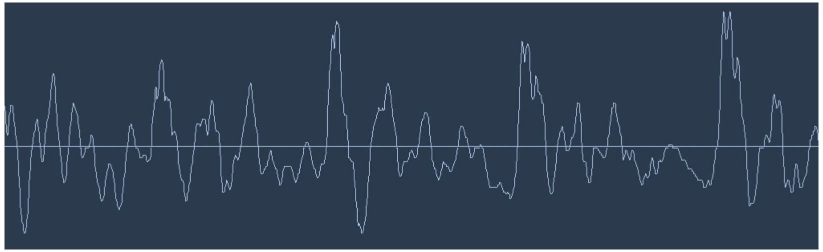
- **Start early**
- **Data:** Understand your input data and output objectives
    - Confirm train/test set (avoid information leak), data sources, feature type and properties, input dimensions, output dimensions.
    - Relatedness between input data sources and goal → More data do not guarantee better performance.
- **Modeling:** From simple to comprehensive, from intuitive to experiment support.
    - Summarize your assumptions that are used in the modeling and discuss whether they are reasonable or need to be further revised.
    - More complex models do not guarantee better performance.

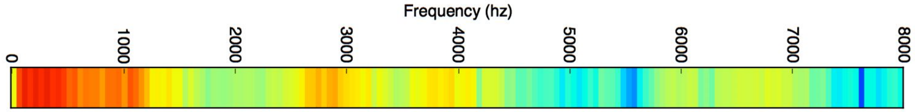
# Feature Engineering

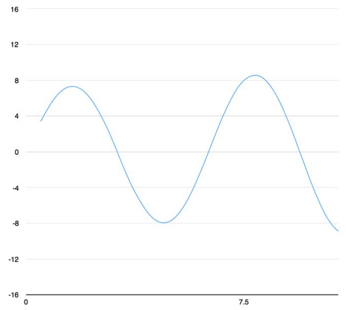
- What is feature engineering?
  - Wikipedia: The process of using **domain knowledge** of the data to **create features** that make machine learning algorithms work.
- Reason for feature engineering
  - Simpler models / Better results
- When to apply feature engineering
  - Data collection → Data cleaning → **Feature Engineering** → Model Training → Evaluation
- Examples
  - Speech recognition: Use Fourier transform to analyze spectrum instead of time series
  - Many other applications …
- Methods: **Selection / Construction / Representation**
  - Example: Indicator variables
  - More feature engineering methods
- Advanced topic: Automated feature engineering

Example of Feature Engineering

## Speech Recognition

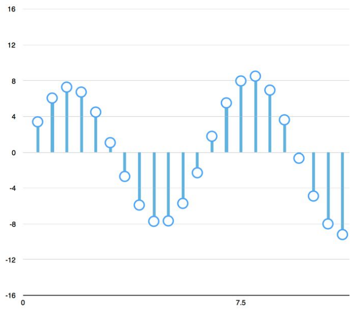




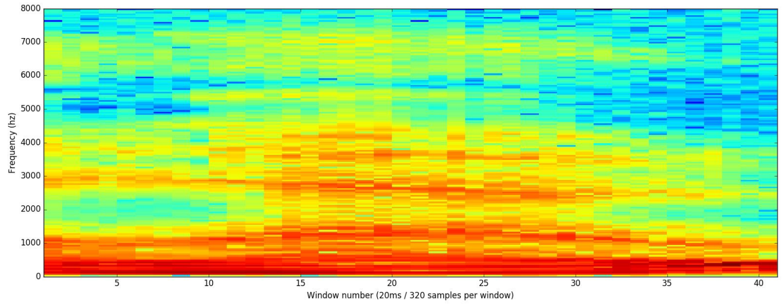


Original Analog Signal

$\overset{?}{==}$



Sampled Digital Signal



Check Blog: [https://medium.com/@ageitgey/machine-learning-is-fun-part-6-how-to-do-speech-recognition-with-deep-learning-28293c162f7a](https://medium.com/@ageitgey/machine-learning-is-fun-part-6-how-to-do-speech-recognition-with-deep-learning-28293c162f7a)

UCLA Engineer Change.

Programming Tips

## Numpy/Pandas

- Numpy: A fundamental package for scientific computing with Python. It has following functions
  - Operations of N-dimensional array object (matrix, tensor)
  - Useful linear algebra, mathematical transform, and random number capabilities
- Pandas: Provide high-performance, easy-to-use data structures and data analysis toolkit for the Python language
- Installation Guide: → Install `numpy` → Install `pandas`
- Tutorials: → `numpy` quickstart → `pandas` tutorial
- Cheatsheet: → `numpy` cheatsheet → `pandas` cheatsheet

## Numpy/Pandas

- What do you need from `numpy` and `pandas` in homework assignments or projects?

**`Numpy:`**

(1) How to create/initialize a numpy array/matrix object
(2) Dimensions of numpy array
(3) Basic operations of numpy array (add, subtract, product, etc)
(4) Indexing, slicing and stacking different arrays
(5) Broadcasting rule

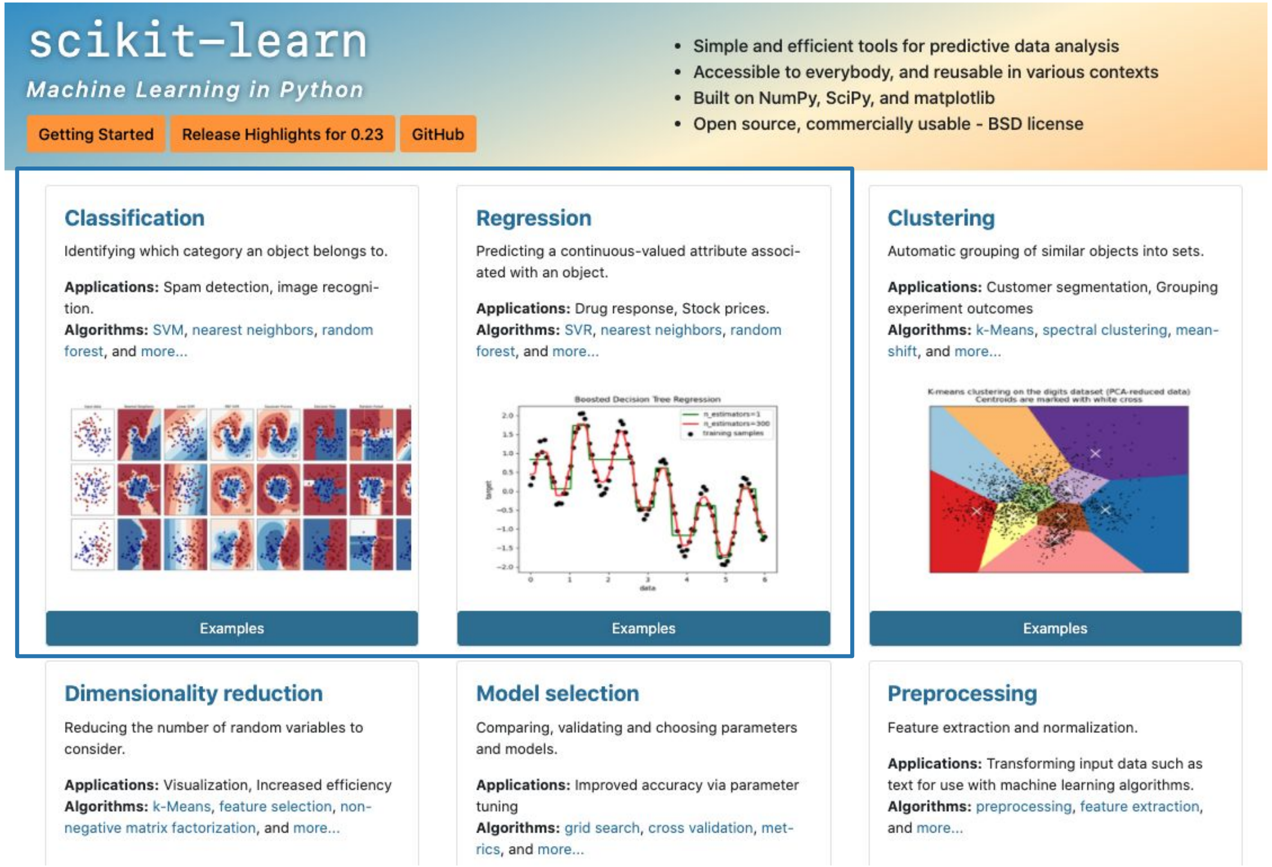
**`Pandas:`**

(1) How to load data from file
(2) How data is stored in DataFrame?
(3) How to read and write data by index in DataFrame?
(4) Iterating over a pandas DataFrame

# Programming Tips: `scikit-learn`

***Warning: Generally you cannot use `scikit-learn` in homework assignments!***

Getting Started:
[https://scikit-learn.org/stable/index.html](https://scikit-learn.org/stable/index.html)



scikit-learn

*Machine Learning in Python*

Getting Started | Release Highlights for 0.23 | GitHub

- Simple and efficient tools for predictive data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

**Classification**

Identifying which category an object belongs to.

**Applications:** Spam detection, image recognition.
**Algorithms:** SVM, nearest neighbors, random forest, and more...

Examples

**Regression**

Predicting a continuous-valued attribute associated with an object.

**Applications:** Drug response, Stock prices.
**Algorithms:** SVR, nearest neighbors, random forest, and more...

Examples

**Clustering**

Automatic grouping of similar objects into sets.

**Applications:** Customer segmentation, Grouping experiment outcomes
**Algorithms:** k-Means, spectral clustering, mean-shift, and more...

Examples

**Dimensionality reduction**

Reducing the number of random variables to consider.

**Applications:** Visualization, Increased efficiency
**Algorithms:** k-Means, feature selection, non-negative matrix factorization, and more...

**Model selection**

Comparing, validating and choosing parameters and models.

**Applications:** Improved accuracy via parameter tuning
**Algorithms:** grid search, cross validation, metrics, and more...

**Preprocessing**

Feature extraction and normalization.

**Applications:** Transforming input data such as text for use with machine learning algorithms.
**Algorithms:** preprocessing, feature extraction, and more...

Boosted Decision Tree Regression
n_estimators=1
n_estimators=300
training samples
target
data
K-means clustering on the digits dataset (PCA-reduced data)
Centroids are marked with white cross

## Tensorflow/PyTorch

- Good to use for **neural network based models**

Getting Started:
https://www.tensorflow.org/tutorials

Getting Started:
https://pytorch.org/get-started

UCLA Samueli Computer Science

# Thank you!

**Q & A**