# Do not distribute course material

You may not and may not allow others to reproduce or distribute lecture notes and course materials publicly whether or not a fee is charged.

# Lecture 3
# Multiple Linear Regression
# Discriminative Supervised Learning

# Thanks to:

- Some of the material is from Prof. Sundeep Rangan
  - This includes some slides and the motivating examples
- Some approaches to introducing the material have been taken from Hsuan'Tien Lin's lecture slides (He is one of the authors of the book *Learning from Data*)

# Notation

- We will follow the notation
- Input (input **features**): $\mathbf{x}$ ($\mathbf{x}^{(i)}$ for the i[th] example)
- Output (**target**): $y$
- example (training example): $(\mathbf{x}, y)$,
- Target function: $f(\mathcal{X}) \rightarrow \mathcal{Y}$
- Data (**training set, training examples**): $(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \ldots, (\mathbf{x}^{(N)}, y^{(N)}),$
- The number of training examples: N
- **Hypothesis**: $g(\mathcal{X}) \rightarrow \mathcal{Y}$

# Outline

- ❑ Motivating example 1 and cost function
- ❑ Motivating example 2
- ❑ Multiple variable linear models
- ❑ Least squares solutions
  - Normal Equations
  - Gradient descent
  - Feature scaling
- ❑ Evaluating our hypothesis
- ❑Extensions

# Example 1

# Predicting Trends

Out[126]:

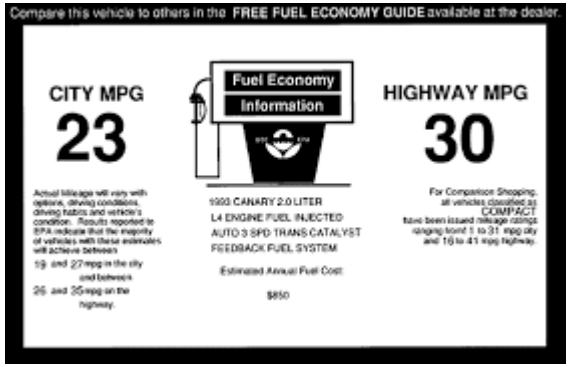
| | mpg | cylinders | displacement | horsepower | weight | acceleration | model year | origin | car name |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 18 | 8 | 307 | 130 | 3504 | 12.0 | 70 | 1 | chevrolet chevelle malibu |
| **1** | 15 | 8 | 350 | 165 | 3693 | 11.5 | 70 | 1 | buick skylark 320 |
| **2** | 18 | 8 | 318 | 150 | 3436 | 11.0 | 70 | 1 | plymouth satellite |
| **3** | 16 | 8 | 304 | 150 | 3433 | 12.0 | 70 | 1 | amc rebel sst |
| **4** | 17 | 8 | 302 | 140 | 3449 | 10.5 | 70 | 1 | ford torino |
| **5** | 15 | 8 | 429 | 198 | 4341 | 10.0 | 70 | 1 | ford galaxie 500 |

- Example: Predicting mpg for a car


| mpg | cylinders | displacement | horsepower | weight | acceleration | model year | origin | car name |
|---|---|---|---|---|---|---|---|---|
| ? | 8 | 383.0 | 170.0 | 3563.0 | 10.0 | 70 | 1 | dodge challenger se |

# Example:  What Determines mpg in a Car?

- What engine characteristics determine fuel efficiency?
- Why would a data scientist be hired to answer this question?
  - Not to help purchasing a specific car
    - The mpg for a currently available car is already known
    - (If the car company isn't lying?)
  - To guide building new cars
    - Understand what is reasonably achievable before full design
  - To find cars that are outside the trend
    - Example:  What cars give great mpg for the cost or size?



# Choose the average MPG?

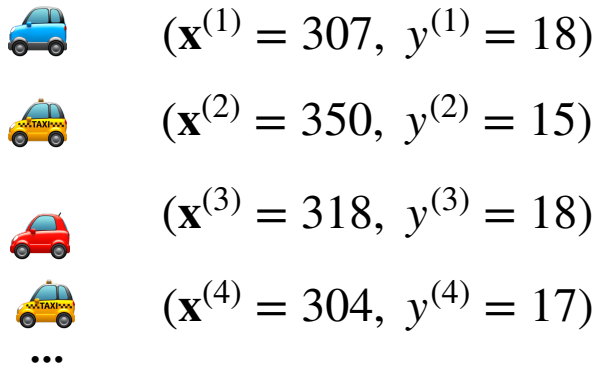
# Keeping it simple

- Lets assume that *one* of the features can approximately predict the MPG

# Lets plot the data

# Visualizing the Data using a scatter plot

- We will plot data in Python using Matplotlib
- A nice tutorial: https://matplotlib.org/users/pyplot_tutorial.html
- How could you predict the mpg of a car not in the data as a function of the displacement?



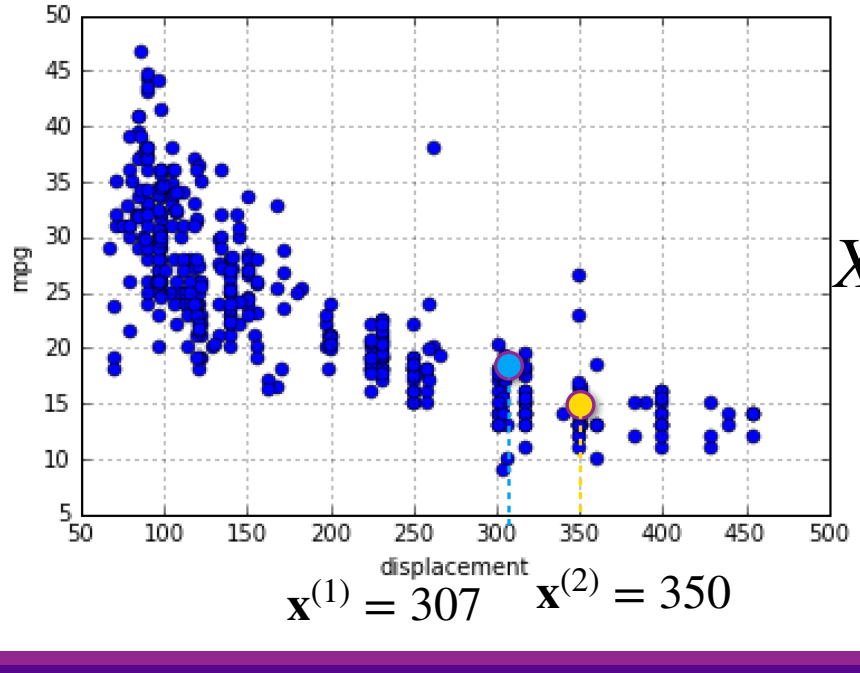
$(\mathbf{x}^{(1)} = 307,\ y^{(1)} = 18)$

$(\mathbf{x}^{(2)} = 350,\ y^{(2)} = 15)$

$(\mathbf{x}^{(3)} = 318,\ y^{(3)} = 18)$

$(\mathbf{x}^{(4)} = 304,\ y^{(4)} = 17)$

...




$\mathbf{x}^{(1)} = 307 \quad \mathbf{x}^{(2)} = 350$

$$X = \begin{bmatrix} 307.0 \\ 350.0 \\ 318.0 \\ 304.0 \\ 302.0 \\ 429.0 \\ 454.0 \\ 440.0 \\ .. \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} 18.0 \\ 15.0 \\ 18.0 \\ 16.0 \\ 17.0 \\ 15.0 \\ 14.0 \\ 14.0 \\ ... \end{bmatrix}$$

# Lets plot some of the other features using a scatter pot

# Postulating a Model

- What relationships do you see?
- Is there a mathematical model relating the variables?
- How well can you predict mpg from these variables?

If a car's horsepower is 170 ($\mathbf{x}$ = 170), can we guess it's mpg (y)?

$(\mathbf{x}^{(1)} = 180,\ y^{(1)} = 18)$

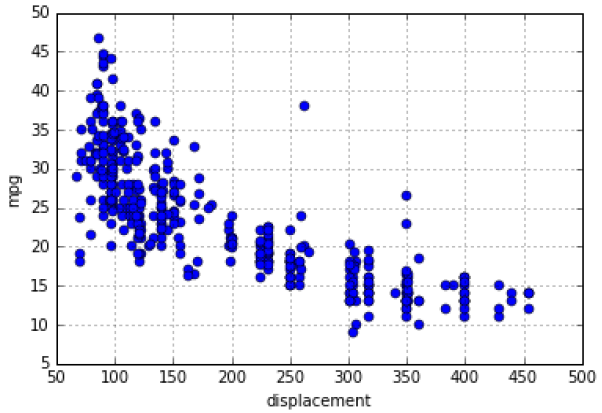
$(\mathbf{x}^{(2)} = 165,\ y^{(2)} = 15)$

$(\mathbf{x}^{(3)} = 150,\ y^{(3)} = 18)$

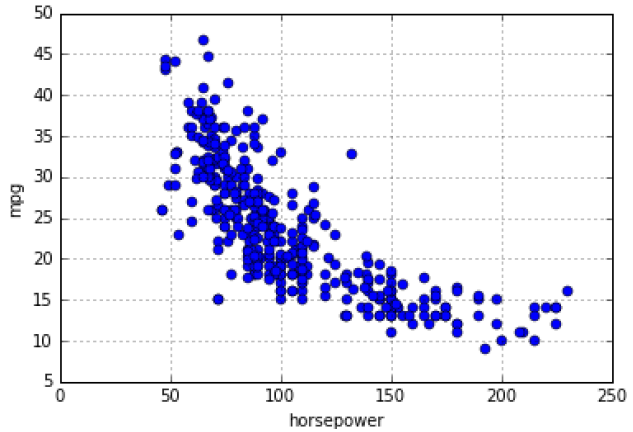
$(\mathbf{x}^{(4)} = 140,\ y^{(4)} = 17)$

...

mpg

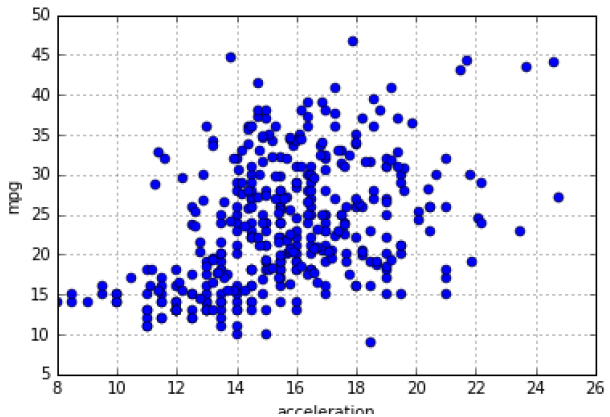



Displacement




$\mathbf{x} = 170$

Horsepower




Acceleration

"Remember that all models are wrong; the practical question is how wrong do they have to be to not be useful." George Box

# Linear Model

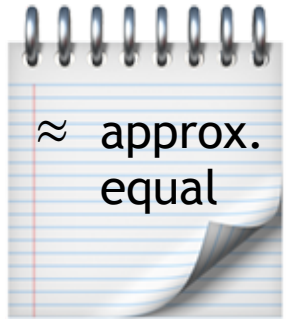
- Many natural phenomenon have a linear relationship
- Easy to interpret results
- Easy to compute

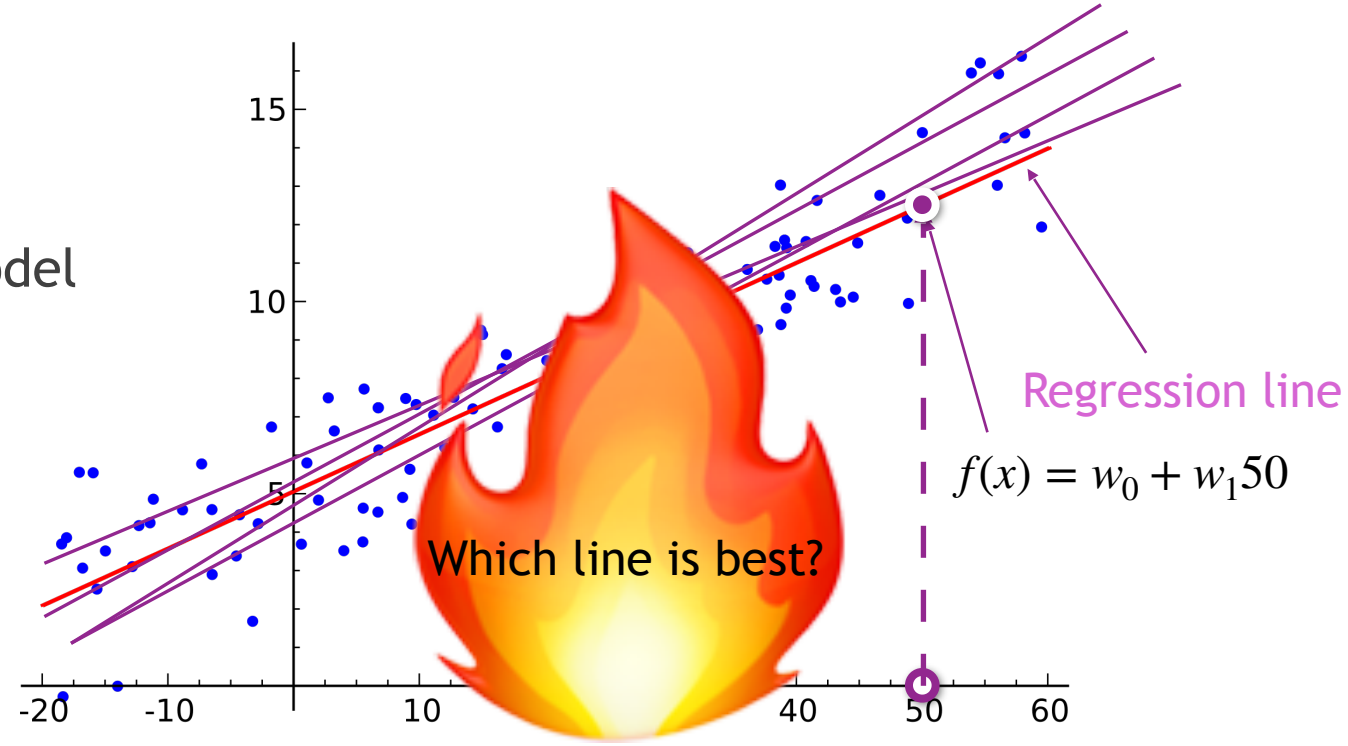
We will use the ideas presented in the linear model in other models

- Assume a linear relation

$$y \approx w_0 + w_1 \mathbf{x}$$

  - $w_0$ = intercept
  - $w_1$ = slope
- $w_0, w_1$ are the parameters of the model
- What are the units of $w_0, w_1$ ?
- When is this model good?




How do we predict the value of y?

$$\hat{y} = f(x) = w_0 + w_1 50$$

# How do we decide which line is 'best'?

*What does it mean for one line to be better than another line?*

HOW MUCH DOES A POINT NOT ON THE LINE **COST**?

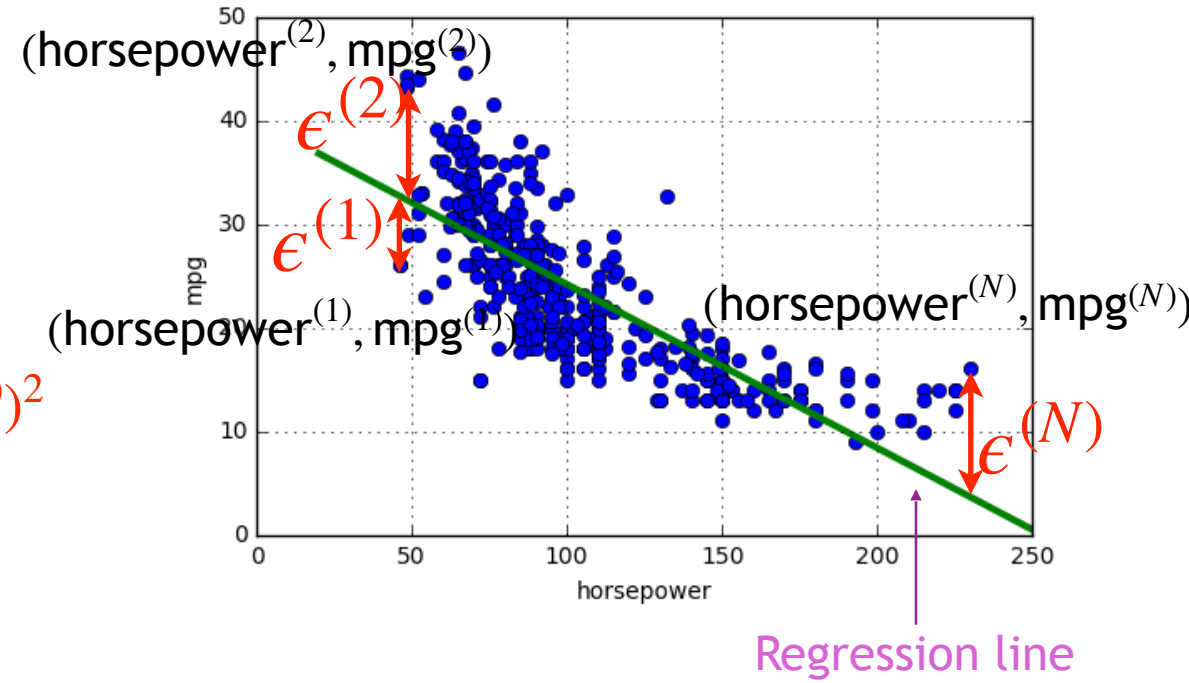
WE COULD CREATE A **COST FUNCTION**. THE BEST LINE HAS THE LOWEST COST WHEN SUMMED OVER ALL THE EXAMPLES

WHAT **COST FUNCTION** SHOULD WE USE?

# Least Squares Model Fitting

- Model relationship between horsepower and mpg as a line $\hat{y} = w_0 + w_1 \mathbf{x}$
- Find parameters $w_0$, $w_1$ to minimize cost $\epsilon^2 = (y - \hat{y})^2$

$$RSS(w) = [\text{mpg}^{(1)} - (w_0 + w_1 \text{horsepower}^{(1)})]^2 \quad (\epsilon^{(1)})^2$$
$$+[\text{mpg}^{(2)} - (w_0 + w_1 \text{horsepower}^{(2)})]^2 \quad (\epsilon^{(2)})^2$$
$$\ldots \quad \ldots$$
$$+[\text{mpg}^{(N)} - (w_0 + w_1 \text{horsepower}^{(N)})]^2 \quad (\epsilon^{(N)})^2$$
$$= \sum_{i=1}^{N} [\text{mpg}^{(i)} - (w_0 + w_1 \text{horsepower}^{(i)})]^2 \quad \sum_{i=1}^{N} (\epsilon^{(i)})^2$$




- Define residual sum of squares: $RSS(w) = \sum_{i=1}^{N} (y^{(i)} - \hat{y}^{(i)})^2$
- Least squares solution: Find $w_0$, $w_1$ to minimize RSS.
  - Geometrically, minimizes squared distances of samples to regression line
- $E_{in}$: ***Empirical error***, the average error the data makes on the data used to ***fit*** the model: $E_{in} = \frac{1}{N} RSS(w)$

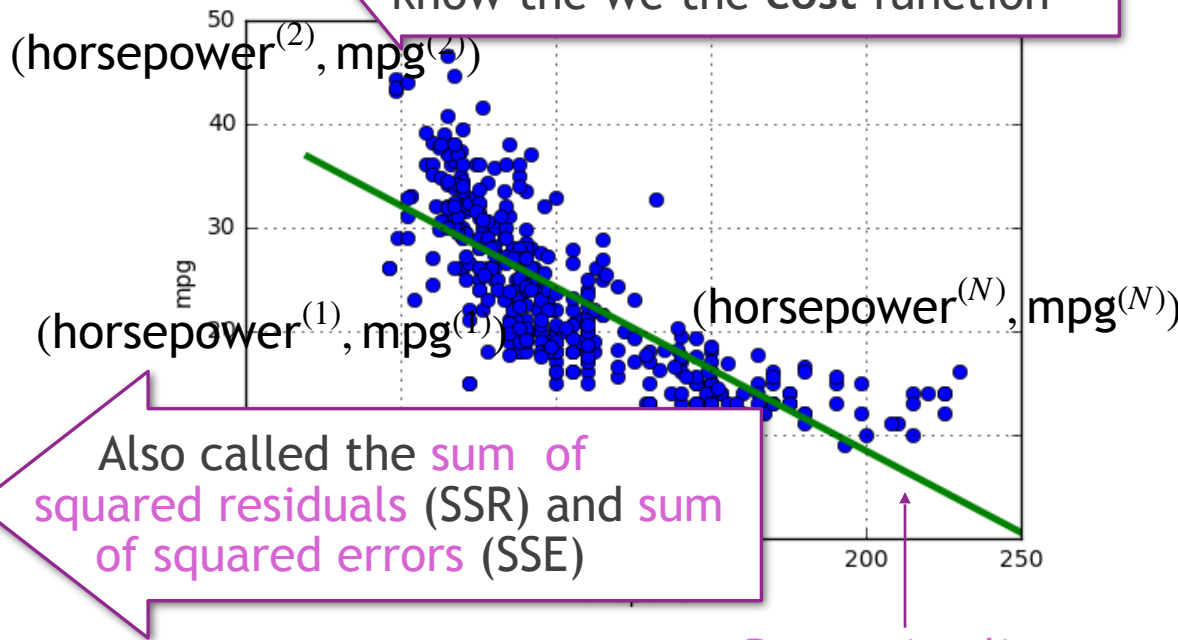
# Least Squares Model Fitting

We chose a hypothesis set, *H*, now we need to ***fit a function*** from *H* (ie find the parameters)

- Model relationship between horsepower and mpg as a line $\hat{y} = w_0 + w_1 \mathbf{x}$
- Find parameters $w_0$, $w_1$ to minimize cost $\epsilon^2 = (y - \hat{y})^2$

To fit the model, we need to know the we the **cost** function

$$RSS(w) = [\text{mpg}^{(1)} - (w_0 + w_1 \text{horsepower}^{(1)})]^2$$
$$+[\text{mpg}^{(2)} - (w_0 + w_1 \text{horsepower}^{(2)})]^2$$
$$\ldots$$
$$+[\text{mpg}^{(N)} - (w_0 + w_1 \text{horsepower}^{(N)})]^2$$
$$= \sum_{i=1}^{N} [\text{mpg}^{(i)} - (w_0 + w_1 \text{horsepower}^{(i)})]^2$$

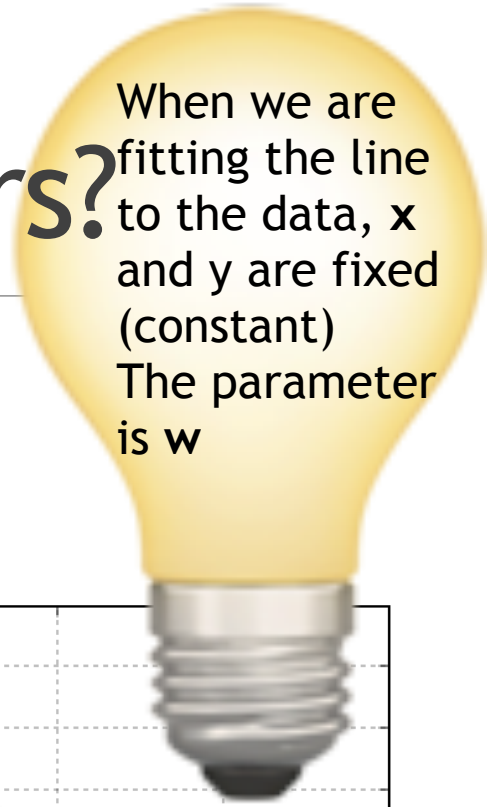



- Define residual sum of squares: $RSS(w) = \sum_{i=1}^{N} (y^{(i)} - \hat{y}^{(i)})^2$

Also called the sum of squared residuals (SSR) and sum of squared errors (SSE)

- Least squares solution: Find $w_0$, $w_1$ to minimize RSS.
  - Geometrically, minimizes squared distances of samples to regression line
- $E_{in}$: ***Empirical error***, the average error the data makes on the data used to ***fit*** the model: $E_{in} = \frac{1}{N} RSS(w)$

# How do we fit/train the parameters?




- Goal: minimize RSS (Residual Sum of Squares)

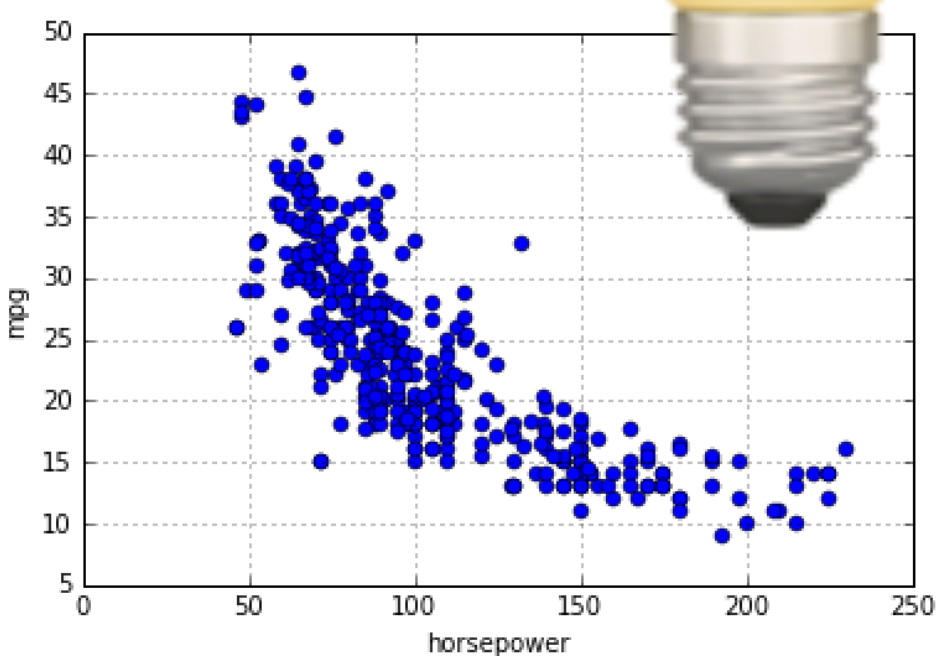
$$RSS(\mathbf{w}) = (y^{(1)} - (w_0 + w_1 x^{(1)}))^2 + (y^{(2)} - (w_0 + w_1 x^{(2)}))^2 + \cdots + (y^{(N)} - (w_0 + w_1 x^{(N)}))^2$$

$$= \sum_{i=1}^{N} (y^{(i)} - (w_0 + w_1 x^{(i)}))^2$$

$$E_{in}(\mathbf{w}) = \frac{1}{N} RSS(\mathbf{w})$$

- We reduced the problem to finding the function that minimizes the RSS!
- *Of course…. the model we may have chosen might not match the true unknown function f

  Most models can only produce a small number of shapes to estimate f



# Probabilisitic Model

# When minimizing the residuals why minimize $\left(y_i - (w_0 + w_1\mathbf{x}^{(i)})\right)^2$?

The target might not be a deterministic function (measurement error, missing features)

$$y^{(i)} = w_0 + w_1\mathbf{x}^{(i)} + \epsilon^{(i)}$$

Assume the noise is independently and identically distributed (IID) Gaussian distribution (normal distribution) with a mean of 0 and variance $\sigma^2$

$$p(\epsilon^{(i)}) = \frac{1}{\sqrt{2\pi\sigma^2}} exp \frac{-(\epsilon^{(i)} - 0)^2}{2\sigma^2} = \frac{1}{\sqrt{2\pi\sigma^2}} exp \frac{-(\epsilon^{(i)})^2}{2\sigma^2}$$ notice that $$p(y^{(i)} \mid \mathbf{x}^{(i)}; \mathbf{w}) = p(\epsilon^{(i)}) = \frac{1}{\sqrt{2\pi\sigma^2}} exp \frac{-\left(y^{(i)} - (w_0 + w_1\mathbf{x}^{(i)})\right)^2}{2\sigma^2}$$

What is the most likely choice of the parameters $\mathbf{w}$?

What is the likelihood of having seen the data if the parameter had been $\mathbf{w}$?

Define $L(\mathbf{w}) = L(\mathbf{w}; X, \mathbf{y})$ to be the likelihood function

We are using L for the likelihood function

$$L(\mathbf{w}) = \prod_{i=1}^{N} p(y^{(i)} \mid \mathbf{x}^{(i)}; \mathbf{w}) = \prod_{i=1}^{N} p(\epsilon^{(i)}) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} exp \frac{-\left(y^{(i)} - (w_0 + w_1\mathbf{x}^{(i)})\right)^2}{2\sigma^2}$$



From: http://cs229.stanford.edu/notes/cs229-notes1.pdf



For more information look at: https://medium.com/quick-code/maximum-likelihood-estimation-for-regression-65f9c99f815d

# When minimizing the residuals why minimize $\left(y^{(i)} - (w_0 + w_1\mathbf{x}^{(i)})\right)^2$?

$(\epsilon^{(i)})^2 = (y^{(i)} - \hat{y}^{(i)})^2 = (y^{(i)} - (w_0 + w_1\mathbf{x}^{(i)}))^2$

The target might not be a deterministic function (measurement error, missing features)

$$y^{(i)} = w_0 + w_1\mathbf{x}^{(i)} + \epsilon^{(i)}$$

Assume the noise is independent identically distributed (IID) Gaussian distribution (normal distribution) with a mean of 0 and variance $\sigma^2$

$$p(\epsilon^{(i)}) = \frac{1}{\sqrt{2\pi\sigma^2}} exp \frac{-(\epsilon^{(i)} - 0)^2}{2\sigma^2} = \frac{1}{\sqrt{2\pi\sigma^2}} exp \frac{-(\epsilon^{(i)})^2}{2\sigma^2}$$ notice that $$p(y^{(i)} \mid \mathbf{x}^{(i)}; \mathbf{w}) = p(\epsilon^{(i)}) = \frac{1}{\sqrt{2\pi\sigma^2}} exp \frac{-\left(y^{(i)} - (w_0 + w_1\mathbf{x}^{(i)})\right)^2}{2\sigma^2}$$

What is the most likely choice of the parameters $\mathbf{w}$?

What is the likelihood of having seen the data if the parameter had been $\mathbf{w}$?

Define $L(\mathbf{w}) = L(\mathbf{w}; X, \mathbf{y})$ to be the likelihood function

We are using L for the likelihood function

$$L(\mathbf{w}) = \prod_{i=1}^{N} p(y^{(i)} \mid \mathbf{x}^{(i)}; \mathbf{w}) = \prod_{i=1}^{N} p(\epsilon^{(i)}) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} exp \frac{-\left(y^{(i)} - (w_0 + w_1\mathbf{x}^{(i)})\right)^2}{2\sigma^2}$$


From: http://cs229.stanford.edu/notes/cs229-notes1.pdf

For more information look at: https://medium.com/quick-code/maximum-likelihood-estimation-for-regression-65f9c99f815d

From: http://cs229.stanford.edu/notes/cs229-notes1.pdf

# What cost makes sense? (cont.)

Which parameter w is best?

The one that is most likely is the one that maximizes $L(\mathbf{w}) = L(\mathbf{w}; X, \mathbf{y})$

$$L(\mathbf{w}) = \prod_{i=1}^{N} p(y^{(i)} \mid \mathbf{x}^{(i)}; \mathbf{w}) = \prod_{i=1}^{N} P(\epsilon^{(i)}) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} exp \frac{-\left(y^{(i)} - (w_0 + w_1 \mathbf{x}^{(i)})\right)^2}{2\sigma^2}$$

Note that maximizing this value is the same as maximizing $\ell(\mathbf{w}) = \log L(\mathbf{w})$

We are using $\ell$ for the log likelihood function

$$\ell(\mathbf{w}) = \log \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} exp \frac{-\left(y^{(i)} - (w_0 + w_1 \mathbf{x}^{(i)})\right)^2}{2\sigma^2} = \sum_{i=1}^{N} \log \frac{1}{\sqrt{2\pi\sigma^2}} exp \frac{-\left(y^{(i)} - (w_0 + w_1 \mathbf{x}^{(i)})\right)^2}{2\sigma^2}$$

$$= N \log \frac{1}{\sqrt{2\pi\sigma^2}} + \frac{1}{2\sigma^2} \sum_{i=1}^{N} - \left(y^{(i)} - (w_0 + w_1 \mathbf{x}^{(i)})\right)^2$$

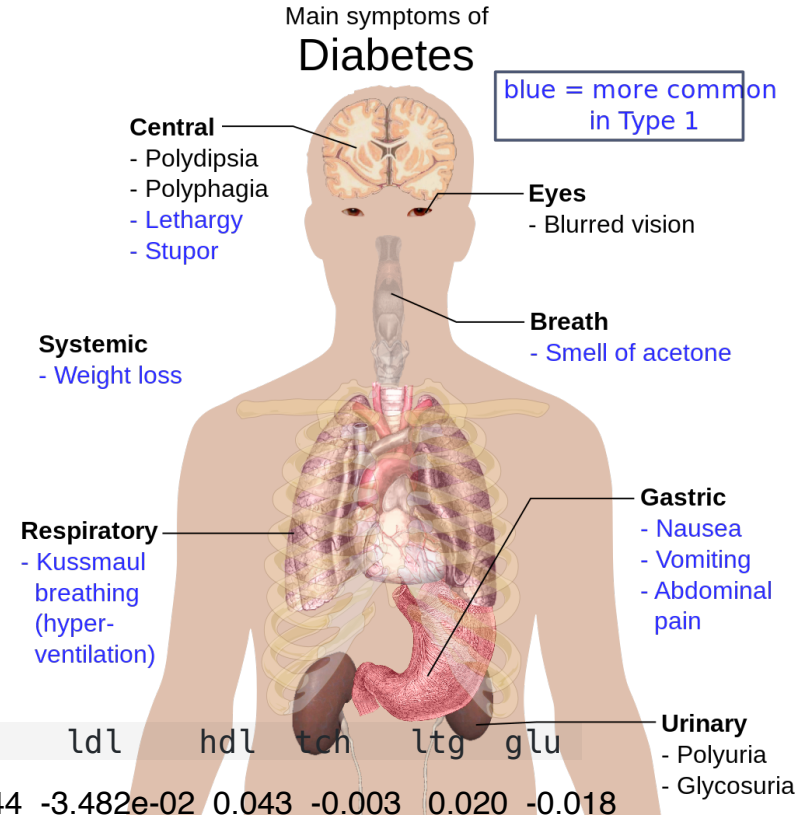
This … is the same as minimizing $\sum_{i=1}^{N} \left(y^{(i)} - (w_0 + w_1 \mathbf{x}^{(i)})\right)^2$

Additional information can be found at: https://www.stat.cmu.edu/~cshalizi/mreg/15/lectures/06/lecture-06.pdf
https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-867-machine-learning-fall-2006/lecture-notes/lec5.pdf

# Example 2

# Example 2: Diabetes Patients Progression

- Can we predict diabetes patients' condition a year after taking 10 baseline measurements?
- The 10 baseline measurements are: age, sex, body mass index. average blood pressure, and 6 blood serum measurements
- Many factors affect diabetes
- are difficult to obtain
  - Hard to derive from first principles
  - Difficult to model physiological process precisely
- Can machine learning help?




By Mikael Häggström - See above. All used images are in public domain.) [Public domain], via Wikimedia Commons

| age | sex | bmi | map | tc | ldl | hdl | tch | ltg | glu |
|---|---|---|---|---|---|---|---|---|---|
| 59 | 2 | 32.1 | 101 | 157 | 93.2 | 38 | 4 | 4.8598 | 87 |
| 48 | 1 | 21.6 | 87 | 183 | 103.2 | 70 | 3 | 3.8918 | 69 |
| 72 | 2 | 30.5 | 93 | 156 | 93.6 | 41 | 4 | 4.6728 | 85 |
| 24 | 1 | 25.3 | 84 | 198 | 131.4 | 40 | 5 | 4.8903 | 89 |
| 50 | 1 | 23 | 101 | 192 | 125.4 | 52 | 4 | 4.2905 | 80 |
| 23 | 1 | 22.6 | 89 | 139 | 64.8 | 61 | 2 | 4.1897 | 68 |
| 36 | 2 | 22 | 90 | 160 | 99.6 | 50 | 3 | 3.9512 | 82 |

| age | sex | bmi | map | tc | ldl | hdl | tch | ltg | glu |
|---|---|---|---|---|---|---|---|---|---|
| 0.038 | 0.051 | 0.062 | 2.187e-02 | -0.044 | -3.482e-02 | 0.043 | -0.003 | 0.020 | -0.018 |
| -0.002 | -0.045 | -0.051 | -2.633e-02 | -0.008 | -1.916e-02 | -0.074 | -0.039 | -0.068 | -0.092 |
| 0.085 | 0.051 | 0.044 | -5.670e-03 | -0.046 | -3.419e-02 | 0.032 | 0.003 | 0.009 | -0.026 |
| -0.089 | -0.045 | -0.012 | -3.666e-02 | 0.012 | 2.499e-02 | 0.036 | 0.034 | 0.023 | -0.009 |
| 0.005 | -0.045 | -0.036 | 2.187e-02 | 0.004 | 1.560e-02 | -0.008 | -0.003 | -0.032 | -0.047 |
| -0.093 | -0.045 | -0.041 | -1.944e-02 | -0.069 | -7.929e-02 | -0.041 | -0.076 | -0.041 | -0.096 |
| -0.046 | 0.051 | -0.047 | -1.600e-02 | -0.040 | -2.480e-02 | -0.001 | -0.039 | -0.063 | -0.038 |

# Matrix Representation of Data

- Data matrix/ Design matrix
- N samples:
  - One sample per row
- d features / attributes /predictors:
  - One feature per column

Features/Attributes

Target vector

$$X = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \dots & x_d^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \dots & x_d^{(2)} \\ \vdots & \vdots & & \vdots \\ x_1^{(N)} & x_2^{(N)} & \dots & x_d^{(N)} \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}$$

Examples/ Samples/ Training Data

- This example:
  - $y^{(i)}$ : quantitative measure of disease progression one year after baseline of i-th sample
  - $x^{(i)}_j$ : j-th feature of i-th sample
  - $(\mathbf{x}^{(i)})^T = [x_1^{(i)}, x_2^{(i)}, \dots, x_N^{(i)}]$ : feature or predictor vector
  - i-th sample contains $\mathbf{x}^{(i)}, y^{(i)}$
- Note: typically we will use uppercase letters for matrices, e.g. X, and lower case letters for scalers and vectors, e.g. **y**

1-indexed example in book and notes (common in math)
0-indexed in python

# e (using y 7 of the 442 trai ng examples)

7th feature is in the 7th

7 x 10 matrix

4th example is in the 4th row

y, a column vector is an N x 1 matrix. E.g. Y is a 7 x 1 matrix. It is also called a 7 dimensional vector

$X =$

| age | sex | bmi | map | tc | ldl | hdl | tch | ltg | glu |
|---|---|---|---|---|---|---|---|---|---|
| 59 | 2 | 32.1 | 101 | 157 | 93.2 | 38 | 4 | 4.8598 | 87 |
| 48 | 1 | 21.6 | 87 | 183 | 103.2 | 70 | 3 | 3.8918 | 69 |
| 72 | 2 | 30.5 | 93 | 156 | 93.6 | 41 | 4 | 4.6728 | 85 |
| 24 | 1 | 25.3 | 84 | 198 | 131.4 | 40 | 5 | 4.8903 | 89 |
| 50 | 1 | 23 | 101 | 192 | 125.4 | 52 | 4 | 4.2905 | 80 |
| 23 | 1 | 22.6 | 89 | 139 | 64.8 | 61 | 2 | 4.1897 | 68 |
| 36 | 2 | 22 | 90 | 160 | 99.6 | 50 | 3 | 3.9512 | 82 |

$y$

| |
|---|
| 151 |
| 75 |
| 141 |
| 206 |
| 135 |
| 97 |
| 138 |

$y^{(4)}$

Where is the 4th example?

Where is the 7th feature for all the examples?

What is the 7th feature of the 4th example $x_7^{(4)}$? $x_7^{(4)} = 40$

What is the measure of disease progression one year after baseline level of the 4th example $y^{(4)}$?

$y^{(4)} = 206$



Data from https://web.stanford.edu/~hastie/Papers/LARS/diabetes.data

# Standardized Example

$X =$

| age | sex | bmi | map | tc | ldl | hdl | tch | ltg | glu |
|---|---|---|---|---|---|---|---|---|---|
| 0.038 | 0.051 | 0.062 | 2.187e-02 | -0.044 | -3.482e-02 | 0.043 | -0.003 | 0.020 | -0.018 |
| -0.002 | -0.045 | -0.051 | -2.633e-02 | -0.008 | -1.916e-02 | -0.074 | -0.039 | -0.068 | -0.092 |
| 0.085 | 0.051 | 0.044 | -5.670e-03 | -0.046 | -3.419e-02 | 0.032 | -0.003 | 0.003 | -0.026 |
| -0.089 | -0.045 | -0.012 | -3.666e-02 | 0.012 | 2.499e-02 | 0.036 | 0.034 | 0.023 | -0.009 |
| 0.005 | -0.045 | -0.036 | 2.187e-02 | 0.004 | 1.560e-02 | -0.008 | -0.003 | -0.032 | -0.047 |
| -0.093 | -0.045 | -0.041 | -1.944e-02 | -0.069 | -7.929e-02 | -0.041 | -0.076 | -0.041 | -0.096 |
| -0.046 | 0.051 | -0.047 | -1.600e-02 | -0.040 | -2.480e-02 | -0.001 | -0.039 | -0.063 | -0.038 |

$y =$

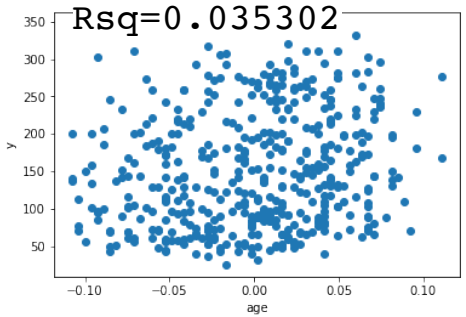
| |
|---|
| -1.133 |
| -77.133 |
| -11.133 |
| 53.866 |
| -17.133 |
| -55.133 |
| -14.133 |

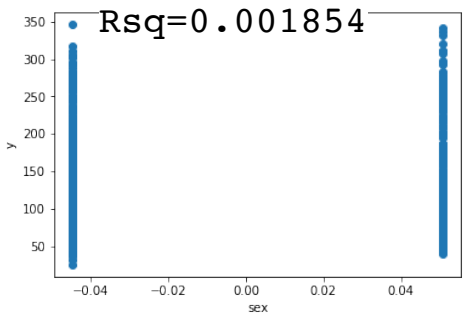
"data are first standardized to have zero mean and unit L2 norm"

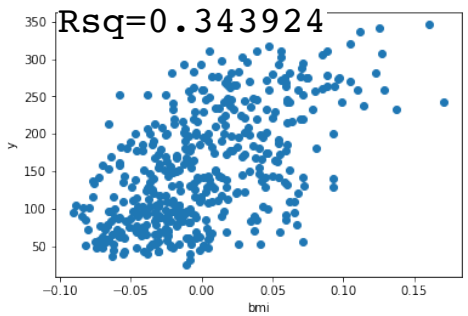
# Outline

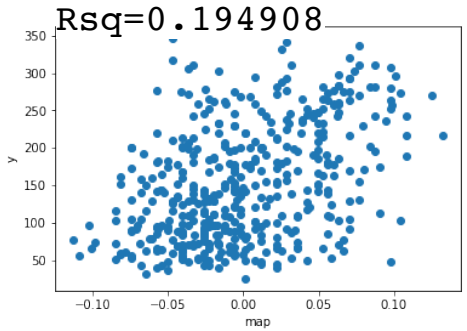
- ❑Motivating Example:  Understanding glucose levels in diabetes patients
- ❑Multiple variable linear models
- ❑Least squares solutions
  - •Normal Equations
  - •Gradient descent
  - •Feature scaling
- ❑Evaluating our hypothesis
- ❑Extensions

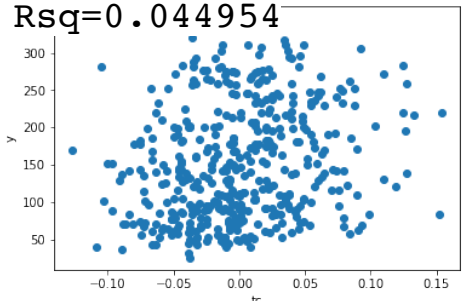
# As before, lets plot at the features

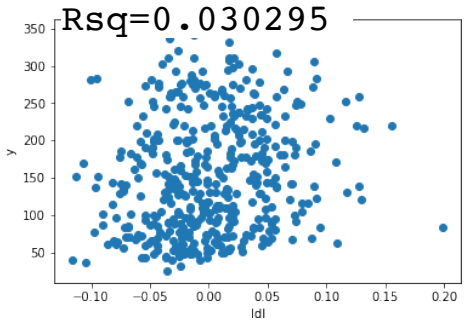


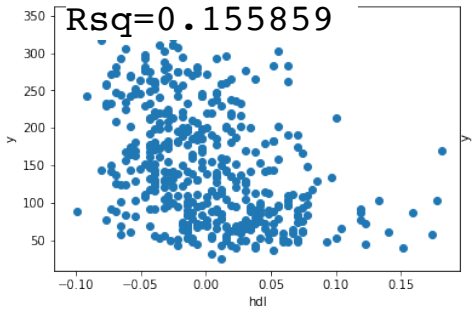



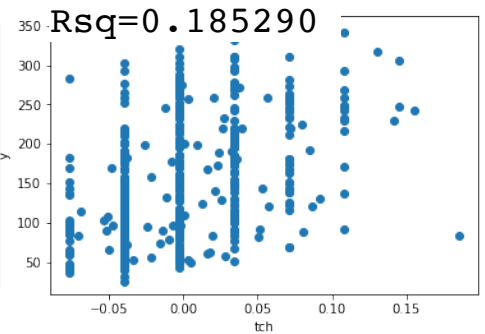



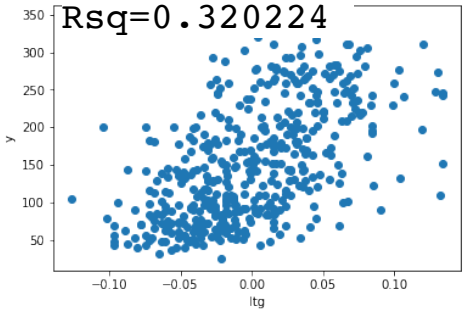



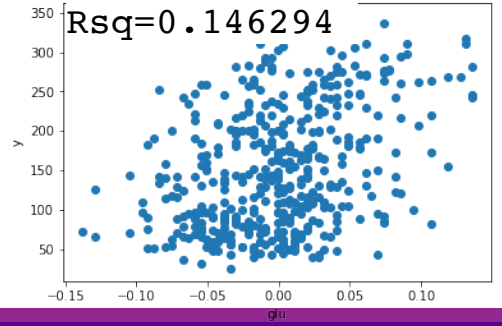



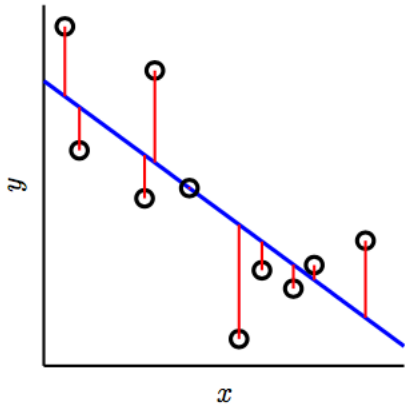



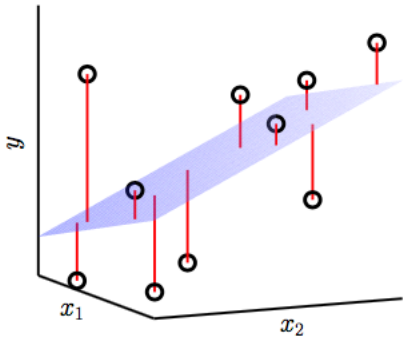



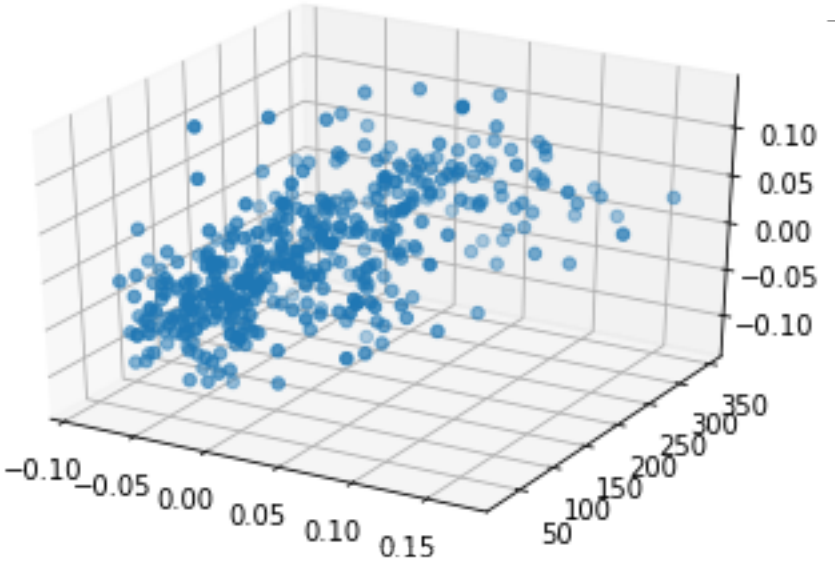

Could we do a better job of predicting if we used more than one feature to make the prediction?

# Least squares Linear Regression

## Finding the best line/hyperplane with the smallest residuals




$$y = f(\mathbf{x}) + \epsilon \qquad \longleftarrow \text{ noisy target } P(y|\mathbf{x})$$

$$y^{(i)} \approx \hat{y}^{(i)} = w_0 + w_1 x_1^{(i)} + w_2 x_2^{(i)} + \cdots + w_d x_d^{(i)}$$

The in sample error $\frac{1}{N} RSS(\mathbf{w}) = E_{\text{in}}(\mathbf{w}) = \frac{1}{N} \sum_{i=1}^{N} (\hat{y}^{(i)} - y^{(i)})^2$



This picture is from the Malik Magdon-Ismail and Hsuan-Tien Lin's lecture slides

# Why Use a Linear Model?

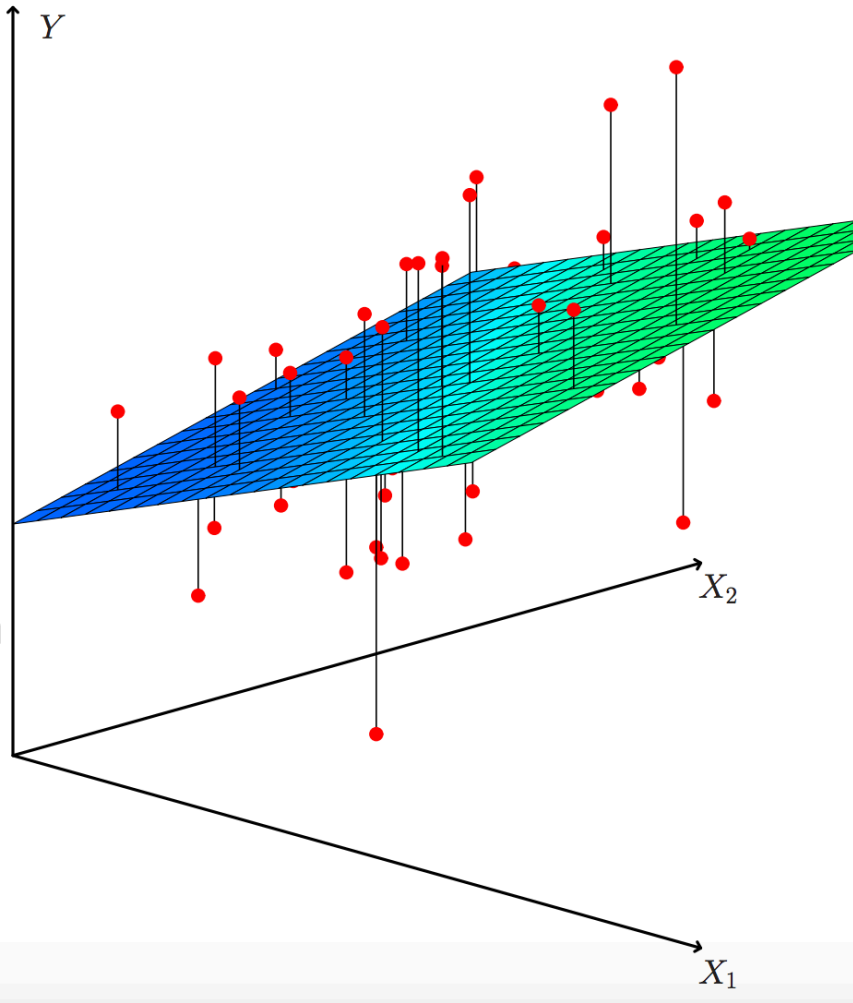
- Many natural phenomena have linear relationship
- Predictor a *small* variation of the linear function

$$y^{(i)} = w_0 + \sum_{j=1}^{d} w_j \cdot x_j^{(i)} + \epsilon^{(i)} \qquad \epsilon^{(i)} \text{ is the residual}$$

- The residual $\epsilon$ is often assumed to be a Gaussian random variable

$$\epsilon \sim \mathcal{N}\left(0, \sigma^2\right)$$

- Think of the examples in d+1 space. If there are two features, then y is a two dimensional plane. For k features, then y is a hyperplane
- Advantages:
  - Simple to compute
  - Easy to interpret relation
- By minimizing $RSS(\mathbf{w}) = \sum_{i=1}^{N} (\hat{y}^{(i)} - y^{(i)})^2$ we minimize our residual error



"Remember that all models are wrong; the practical question is how wrong do they have to be to not be useful." George Box

Linear regression is predicting a continuous value given some input

# Matrix and Vector Review

- Consider

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}, \qquad B = \begin{bmatrix} 2 & 0 \\ 3 & 2 \end{bmatrix}, \qquad x = \begin{bmatrix} 2 \\ 3 \end{bmatrix},$$

- Compute (computations on the board):
  - Matrix vector multiply: $Ax$
  - Transpose: $A^T$
  - Matrix multiply: $AB$

- Compute the dot product of two vectors x and w
- Predict y given **w** and **x**

# Matrix Form of Linear Regression

- Our predicted value for the i-th sample: $\hat{y}^{(i)} = w_0 + w_1 x_1^{(i)} + w_2 x_2^{(i)} + \cdots + w_d x_d^{(i)} = w_0 + \sum_{j=1}^{d} w_j \cdot x_j^{(i)} = w_0 + \mathbf{w}_{1:d}^T \mathbf{x}^{(i)}$
- Notice that we can *simplify* our notation if we add a one in the front of **x**

  $\mathbf{x} = [x_0, x_1, x_2, \ldots, x_d]^T = [1, x_1, x_2, \ldots, x_d]^T$ where the added coordinate $x_0$ is set to 1 $\quad \hat{y}^{(i)} = \mathbf{w}^T \mathbf{x}^{(i)}$

- The matrix X and vector **w** are *now*

$$X = \begin{bmatrix} 1 & x_1^{(1)} & \ldots & x_d^{(1)} \\ 1 & x_1^{(2)} & \ldots & x_d^{(2)} \\ \vdots & \vdots & & \vdots \\ 1 & x_1^{(N)} & \ldots & x_d^{(N)} \end{bmatrix} \qquad \mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_d \end{bmatrix}$$

*d + 1 coefficients*
*we merged the bias/intercept with the other weights into one vector*

  - *Feature matrix/Design matrix/Data matrix*
  - **w** is our regression vector/*coefficient vector*

- Now we can compute our predicted values in a simpler form: $\hat{y}^{(i)} = \mathbf{w}^T \mathbf{x}^{(i)} = (\mathbf{x}^{(i)})^T \mathbf{w}$ $\quad \hat{\mathbf{y}} = X\mathbf{w}$
  - Note that $\hat{\mathbf{y}}$ is a N-dimensional column vector where $\hat{y}_i$ is the predicted value of the i-th example

# Example (using only some of the 4 ini

| | age | sex | bmi | map | tc | ldl | hdl | tch | ltg | glu |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.038 | 0.051 | 0.062 | 2.187e-02 | -0.044 | -3.482e-02 | 0.043 | -0.003 | 0.020 | -0.018 |
| 1 | -0.002 | -0.045 | -0.051 | -2.633e-02 | -0.008 | -1.916e-02 | -0.074 | -0.039 | -0.068 | -0.092 |
| 1 | 0.085 | 0.051 | 0.044 | -5.670e-03 | -0.046 | -3.419e-02 | 0.032 | -0.003 | 0.003 | -0.026 |
| 1 | -0.089 | -0.045 | -0.012 | -3.666e-02 | 0.012 | 2.499e-02 | 0.036 | 0.034 | 0.023 | -0.009 |
| 1 | 0.005 | -0.045 | -0.036 | 2.187e-02 | 0.004 | 1.560e-02 | -0.008 | -0.003 | -0.032 | -0.047 |
| 1 | -0.093 | -0.045 | -0.041 | -1.944e-02 | -0.069 | -7.929e-02 | -0.041 | -0.076 | -0.041 | -0.096 |
| 1 | -0.046 | 0.051 | -0.047 | -1.600e-02 | -0.040 | -2.480e-02 | -0.001 | -0.039 | -0.063 | -0.038 |

$$Xw = \begin{bmatrix} \cdots \\ \cdots \\ \cdots \\ \cdots \\ \cdots \\ \cdots \\ \cdots \end{bmatrix} \begin{bmatrix} 152.34786452 \\ -16.57607993 \\ -254.66532396 \\ 560.98630022 \\ 278.91811152 \\ -393.41357305 \\ 97.05460405 \\ -19.0023093 \\ 169.46450327 \\ 632.95050374 \\ 114.21638941 \end{bmatrix} = \begin{bmatrix} 204.51116637 \\ 67.10485972 \\ 175.02956894 \\ 165.88615565 \\ 123.11207835 \\ 105.64709238 \\ 71.73293158 \end{bmatrix} = \hat{y}$$

Think of this as points in d+1 dimensional space

Our goal is to find this

So we can predict!

Assigning a new feature whose value is always one

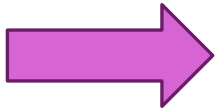
w is a (d+1)-dimensional vector

$\hat{y}$ is a N-dimensional

In numpy this is X.dot(w)

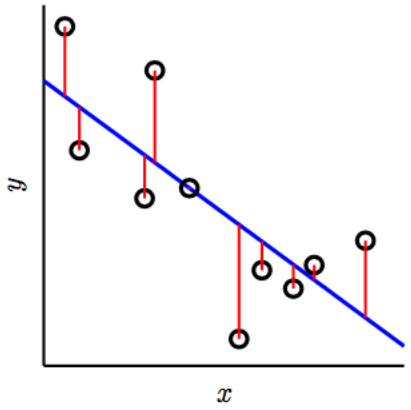
# Outline

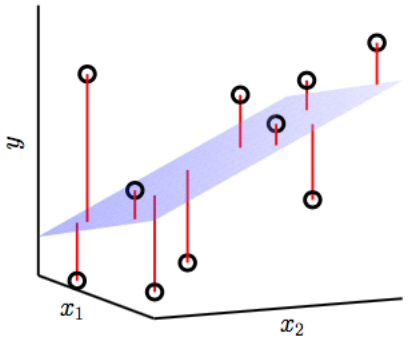
- Motivating Example:  Understanding glucose levels in diabetes patients
- Multiple variable linear models
- Least squares solutions
  - Normal Equations
  - Gradient descent
  - Feature scaling
- Evaluating our hypothesis
- Special case:  Simple linear regression
- Extensions



# Least squares Linear Regression

## Finding the best line/hyperplane with the smallest residuals




$$y = f(\mathbf{x}) + \epsilon \qquad \longleftarrow \text{ noisy target } P(y|\mathbf{x})$$

$$\hat{y}^{(i)} = w_0 + w_1 x_1^{(i)} + w_2 x_2^{(i)} + \cdots + w_d x_d^{(i)} = (\mathbf{x}^{(i)})^T \mathbf{w}$$

The in sample error

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N}\sum_{i=1}^{N} (\hat{y}^{(i)} - y^{(i)})^2 = \frac{1}{N}\sum_{i=1}^{N} (\mathbf{w}^T\mathbf{x}^{(i)} - y^{(i)})^2$$

$$RSS(\mathbf{w}) = \sum_{i=1}^{N} (\hat{y}^{(i)} - y^{(i)})^2$$

# Small example

$$(\hat{\mathbf{y}}-\mathbf{y})^2 = \left( \begin{bmatrix} 204.51116637 \\ 67.10485972 \\ 175.02956894 \\ 165.88615565 \\ 123.11207835 \\ 105.64709238 \\ 71.73293158 \end{bmatrix} - \begin{bmatrix} 151 \\ 75 \\ 141 \\ 206 \\ 135 \\ 97 \\ 138 \end{bmatrix} \right)^2 = \begin{bmatrix} 53.51116637 \\ -7.89514028 \\ 34.02956894 \\ -40.11384435 \\ -11.88792165 \\ 8.64709238 \\ -66.26706842 \end{bmatrix}^2 = \begin{bmatrix} 2863.44492665 \\ 62.33324 \\ 1158.01156228 \\ 1609.12050832 \\ 141.32268118 \\ 74.77220665 \\ 4391.32435646 \end{bmatrix}$$

$$RSS(\mathbf{w}) = \sum_{i=1}^{n} (y - \hat{y})^2 = 10300.329$$

$$RMSE(\mathbf{w}) = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (y - \hat{y})^2} = \quad 38.360$$

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N} RSS(\mathbf{w}) \quad = 1471.476$$

# Matrix form of $E_{in}$

$$\hat{y}^{(i)} = w_0 + w_1 x_1^{(i)} + w_2 x_2^{(i)} + \cdots + w_d x_d^{(i)} = (\mathbf{x}^{(i)})^T \mathbf{w}$$

$$E_{in}(\mathbf{w}) = \frac{1}{N}\sum_{i=1}^{N} (\hat{y}^{(i)} - y^{(i)})^2 = \frac{1}{N}\sum_{i=1}^{N} \left((\mathbf{x}^{(i)})^T \mathbf{w} - y^{(i)}\right)^2$$

$$= \frac{1}{N}\left\| \begin{bmatrix} (\mathbf{x}^{(1)})^T \mathbf{w} - \mathbf{y}^{(1)} \\ (\mathbf{x}^{(2)})^T \mathbf{w} - \mathbf{y}^{(2)} \\ \vdots \\ (\mathbf{x}^{(N)})^T \mathbf{w} - \mathbf{y}^{(N)} \end{bmatrix} \right\|_2^2 = \frac{1}{N}\left\| \begin{bmatrix} -- (\mathbf{x}^{(1)})^T -- \\ -- (\mathbf{x}^{(2)})^T -- \\ \vdots \\ -- (\mathbf{x}^{(N)})^T -- \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_d \end{bmatrix} - \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \right\|_2^2 = \frac{1}{N}\|X\mathbf{w} - \mathbf{y}\|_2^2$$

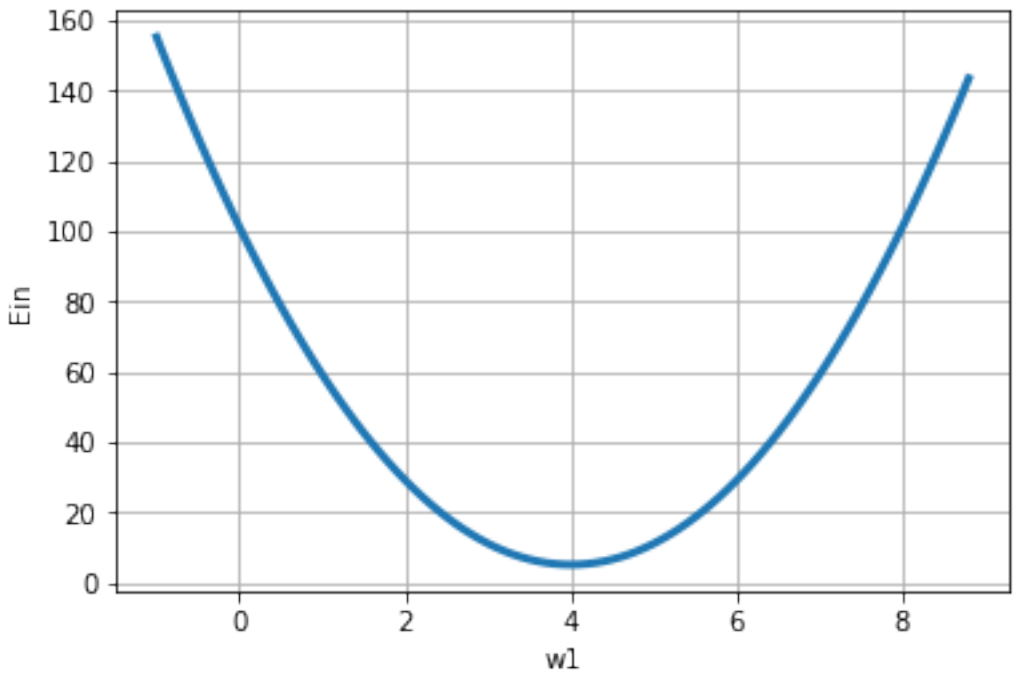
The size of a vector is referred to as the norm o the vector. What is the "size". It depends. The L2 norm of a vector is the square root of the sum of the squared vector values

$$\mathbf{v}^T = [1,2,3]$$

$$\|\mathbf{v}\|_2 = \sqrt{1^2 + 2^2 + 3^2}$$

# Finding **w** to minimize $E_{in}$

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2 = \frac{1}{N}||X\mathbf{w} - \mathbf{y}||_2^2$$




$$\nabla E_{\text{in}}(\mathbf{w}) = \begin{bmatrix} \frac{\partial E_{\text{in}}(\mathbf{w})}{\partial w_0} \\ \frac{\partial E_{\text{in}}(\mathbf{w})}{\partial w_1} \\ \vdots \\ \frac{\partial E_{\text{in}}(\mathbf{w})}{\partial w_d} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

The sum of a continuous functions is continuous
The sum of convex functions is convex
The sum of differentiable functions is differentiable
If we had 5 equations and 5 unknowns we could use linear algebra to find the hyperplane with no residuals

Goal find $\mathbf{w}_{\text{lin}}$ such that $\nabla E_{\text{in}}(\mathbf{w}) = \mathbf{0}$

# Finding **w** to minimize $E_{in}$

Goal find $\mathbf{w}_{\text{lin}}$ such that $\nabla E_{\text{in}}(\mathbf{w}) = \mathbf{0}$

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2 = \frac{1}{N}\|X\mathbf{w} - \mathbf{y}\|_2^2$$

$$= \frac{1}{N}(X\mathbf{w} - \mathbf{y})^T(X\mathbf{w} - \mathbf{y}) = \frac{1}{N}(\mathbf{w}^T \underbrace{X^T X}_{A}\mathbf{w} - 2\mathbf{w}^T \underbrace{X^T\mathbf{y}}_{b} + \underbrace{\mathbf{y}^T\mathbf{y}}_{c})$$

If w was a variable and not a vector:

$$E_{\text{in}}(w) = \frac{1}{N}(aw^2 - 2wb - c)$$

$$\nabla E_{\text{in}}(w) = \frac{1}{N}(2aw - 2b)$$

If **w** is a vector (this is very similar)

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N}(\mathbf{w}^T A\mathbf{w} - 2\mathbf{w}^T\mathbf{b} - c)$$

$$\nabla E_{\text{in}}(\mathbf{w}) = \frac{1}{N}(2A\mathbf{w} - 2\mathbf{b})$$

$$\nabla E_{\text{in}}(\mathbf{w}) = \frac{2}{N}(X^T X\mathbf{w} - X^T\mathbf{y})$$

1) Note that $\|\mathbf{v}\|^2 = \mathbf{v}^T\mathbf{v} = v_1^2 + v_2^2 + \cdots + v_d^2$

2) You can verify the following by writing out the matrices

$(A\mathbf{b} + \mathbf{c})^T = \mathbf{b}^T A^T + \mathbf{c}^T$

The following are identities from vector calculus

$\nabla_{\mathbf{w}} \mathbf{w}^T A\mathbf{w} = (A + A^T)\mathbf{w}$

$\nabla_{\mathbf{w}} \mathbf{w}^T\mathbf{b} = \mathbf{b}$

If A is symmetric this becomes $2A\mathbf{w}$



The calculus identities proved here: https://atmos.washington.edu/~dennis/MatrixCalculus.pdf

# Finding **w** to minimize $E_{in}$

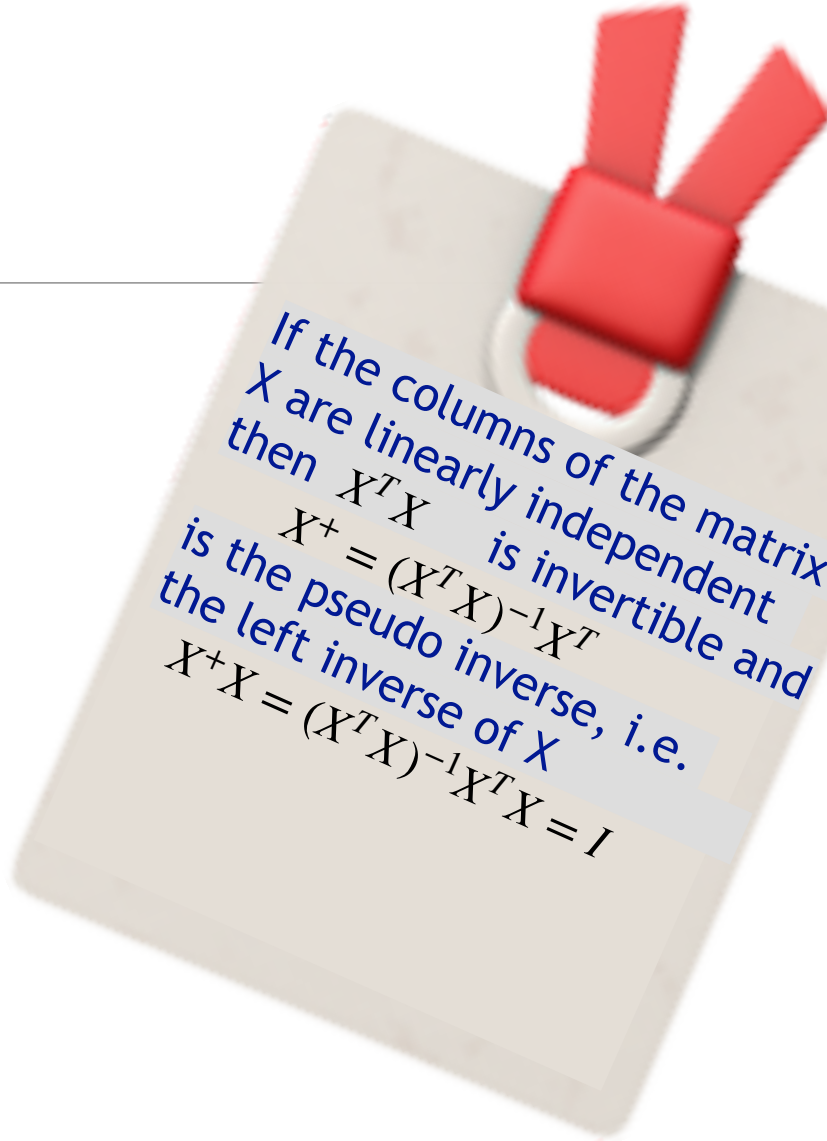
Goal find $\mathbf{w}_{\text{lin}}$ such that $\nabla E_{\text{in}}(\mathbf{w}) = \mathbf{0}$

Setting $\nabla E_{\text{in}}(\mathbf{w}) = \frac{2}{N}(X^T X\mathbf{w} - X^T\mathbf{y}) = \mathbf{0}$

Results in: $X^T X\mathbf{w} = X^T\mathbf{y}$

Thus $\mathbf{w}_{\text{lin}} = \underbrace{\left(X^T X\right)^{-1} X^T}\mathbf{y}$

pseudoinverse
left inverse



# Finding **w** to minimize $E_{in}$

**Linear Regression Algorithm:**

1. Construct the matrix X and the vector $\mathbf{y}$ from the data set $(\mathbf{x}^{(1)}, y^{(1)}), \ldots, (\mathbf{x}^{(N)}, y^{(N)})$, where each $\mathbf{x}$ includes the $x_0 = 1$ coordinate,

$$X = \underbrace{\begin{bmatrix} 1 & x_1^{(1)} & \ldots & x_d^{(1)} \\ 1 & x_1^{(2)} & \ldots & x_d^{(2)} \\ \vdots & \vdots & & \vdots \\ 1 & x_1^{(N)} & \ldots & x_d^{(N)} \end{bmatrix}}_{\text{data matrix}} \qquad \mathbf{y} = \underbrace{\begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}}_{\text{target vector}}$$

2. Compute the pseudo inverse $X^\dagger$ of the matrix X. If $X^TX$ is invertible,

$$X^\dagger = (X^TX)^{-1}X^T$$

3. Return $\mathbf{w}_{\text{lin}} = X^\dagger \mathbf{y}$.

Computing the pseudo inverse is faster! $O(n^2)$ instead of $O(n^{2.4})$ to $O(n^3)$



(Slightly modified) Algorithm from the book *Learning form Data*

# Finding Parameters via Optimization
## A general ML recipe

### General ML problem

- ❑Get data
- ❑Pick a model with parameters
- ❑Pick a loss function
  - Measures goodness of fit model to data
  - Function of the parameters
- ❑Find parameters that minimizes loss

### Multiple linear regression

Data: $\hat{y}^{(j)} = w_0 + w_1 x_1^{(j)} + w_2 x_2^{(j)} + \cdots + w_d x_d^{(j)}$

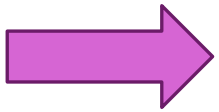
Linear model: $\text{RSS}(\mathbf{w}) = \sum_{i=1}^{N} (y^{(i)} - \hat{y}^{(i)})^2$

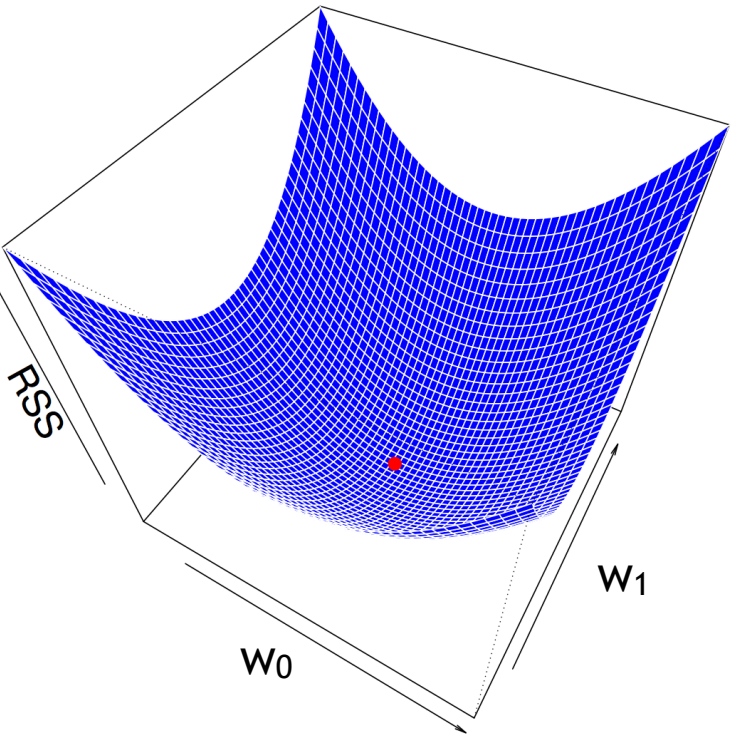
Loss function: $(\mathbf{x}^{(i)}, y^{(i)}), i = 1,2,\ldots, N$

Select $\mathbf{w} = [w_0, w_1, w_2, \ldots, w_d]^T$ to minimize $\text{RSS}(\mathbf{w})$

# Outline

- Motivating Example:  Understanding glucose levels in diabetes patients
- Multiple variable linear models
- Least squares solutions
  - Normal Equations
  - Gradient descent
  - Feature scaling
- Evaluating our hypothesis
- Special case:  Simple linear regression
- Extensions

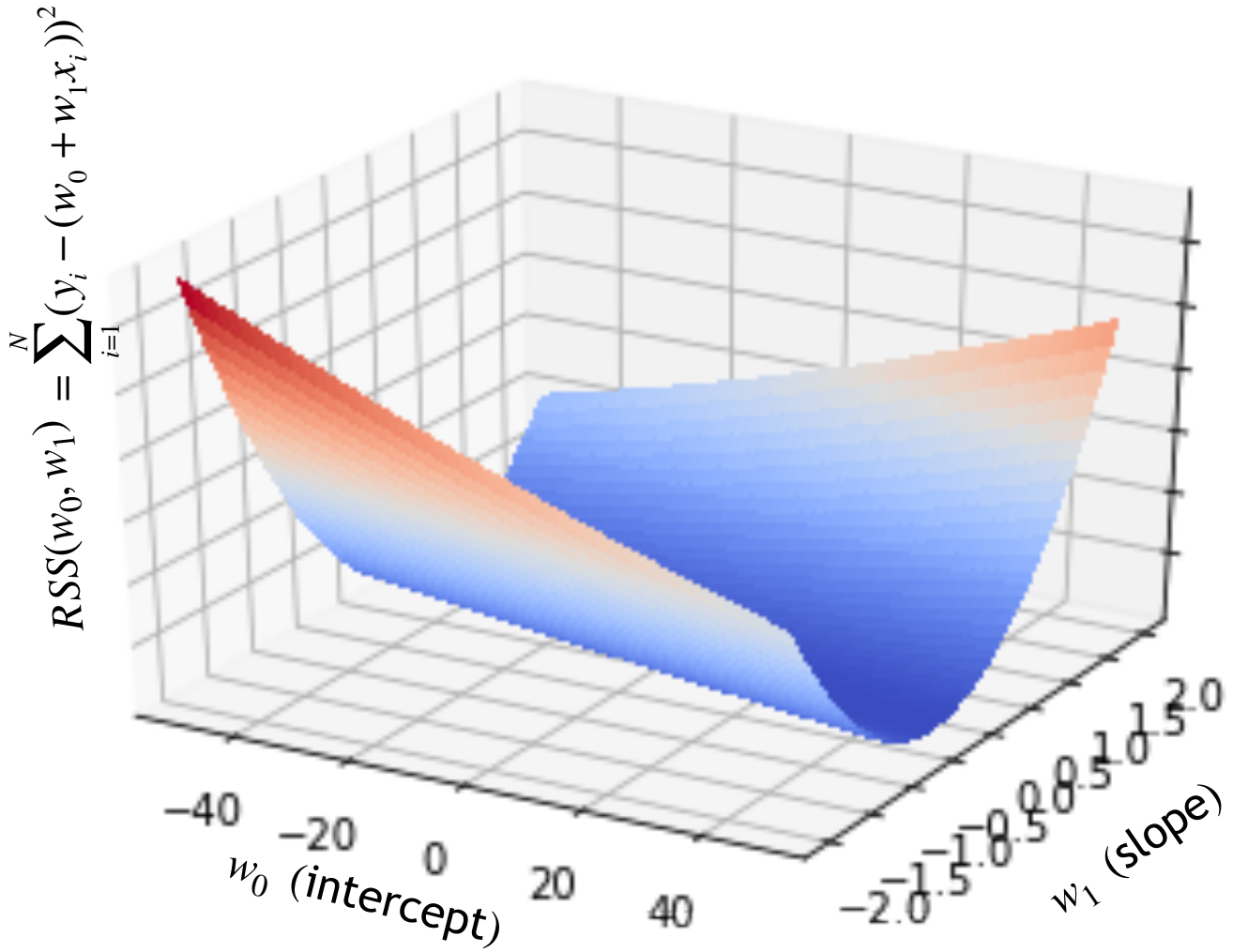





# Batch Gradient Descent!

"Some of the figures in this presentation are taken from "An Introduction to Statistical Learning, with applications in R" (Springer, 2013) with permission from the authors: G. James, D. Witten, T. Hastie and R. Tibshirani "

# If we change $w=(w_0, w_1)$, $\mathcal{E}_{in}(w)=(1/N)RSS(w)$ (the cost) changes



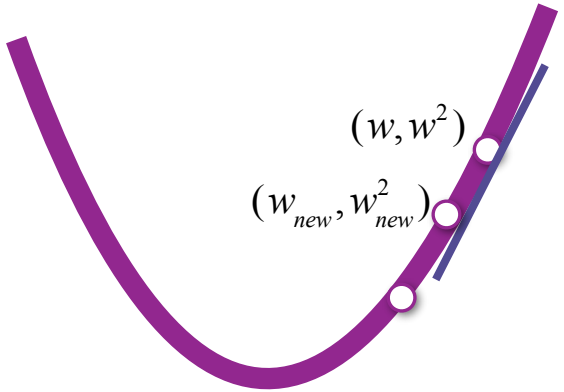

From calculus we know that the direction for the maximum rate of change for a function J(w) is

$$\nabla J(\mathbf{w}) = \begin{bmatrix} \frac{\partial J(\mathbf{w})}{\partial w_0} \\ \frac{\partial J(\mathbf{w})}{\partial w_1} \\ \vdots \\ \frac{\partial J(\mathbf{w})}{\partial w_d} \end{bmatrix}$$

# Another approach to minimize a function

- Notice that the step size decreases as we move toward the minimum
  - If we are at w = 10 the step size is 2
  - If we are at 8 the step size is 1.6
- If we started at x = -10, the derivative is negative, so $w_{new} = (-10) - \alpha \cdot 2 \cdot (-10)$
  $= (-10) - (0.1) \cdot 2 \cdot (-10)$
  $= (-10) + 2 = -8$
- If $\alpha$ is too large, then we can move too far (i.e overshoot). For our example:
  - If $\alpha = 1$ $w_{new} = 10 - 1 \cdot 2 \cdot 10 = -10$
  - if $\alpha > 1$ $|w_{new}| > |w|$ so the f(w) increases instead of reaching the minimum for subsequent updates!
  - if $\alpha = 0.7$ $w_{new} = 10 - (0.7) \cdot 2 \cdot 10 = -4$ we overshoot (however the function does decrease)

f(w) = w² df/dw = 2w




$w_{new} = w - \alpha \cdot 2 \cdot w$ $\alpha = 0.1$

$w_{new} = 10 - \alpha \cdot 2 \cdot 10 = 10 - (0.1) \cdot 2 \cdot 10 = 8$

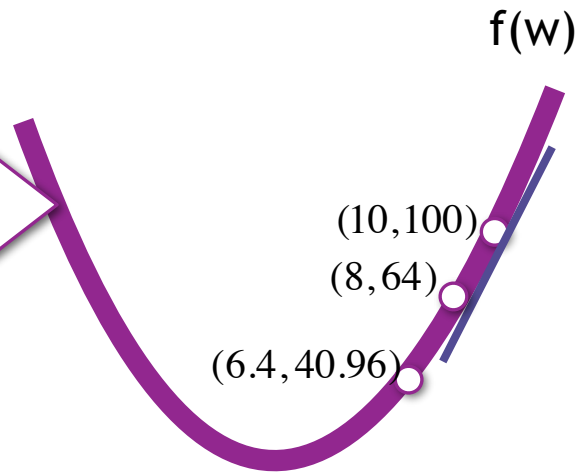
$w_{new} = 8 - \alpha \cdot 2 \cdot 8 = 8 - (0.1) \cdot 2 \cdot 8 = 6.4$

# Another approach to minimize a function

- Notice that the step size decreases as we move toward the minimum
  - If we are at w = 10 the step size is 2
  - If we are at 8 the step siz
- If we started at x = -10, th negative, so

$$w_{new} = (-10) - \alpha \cdot 2 \cdot (-10)$$
$$= (-10) - (0.1) \cdot 2 \cdot (-10)$$
$$= (-10) + 2 = -8$$

The derivate gives the direction to move




$$w_{new} = w - \alpha \cdot 2 \cdot w \qquad \alpha = 0.1$$
$$w_{new} = 10 - \alpha \cdot 2 \cdot 10 = 10 - (0.1) \cdot 2 \cdot 10 = 8$$
$$w_{new} = 8 - \alpha \cdot 2 \cdot 8 = 8 - (0.1) \cdot 2 \cdot 8 = 6.4$$

- If $\alpha$ is too large, then we can move too far (i.e overshoot). For our example:
  - If $\alpha = 1$ $\quad w_{new} = 10 - 1 \cdot 2 \cdot 10 = -10$
  - if $\alpha > 1$ $\quad |w_{new}| > |w|$ so the f(w) increases instead of reaching the minimum for subsequent updates!
  - if $\alpha = 0.7$ $\quad w_{new} = 10 - (0.7) \cdot 2 \cdot 10 = -4$ we overshoot (however the function does decrease)

# Cost function

- Minimizing $RSS(\mathbf{w}) = \sum_{i=1}^{N}((\mathbf{w}^T x^{(i)}) - y^{(i)})^2$ (our cost function for linear regression) is the same as minimizing:

$$\mathrm{J}(\mathbf{w}) = \frac{1}{2N}\sum_{i=1}^{N}((\mathbf{w}^T x^{(i)}) - y^{(i)})^2 = \frac{1}{2N}\mathrm{np.sum(np.square}(X\mathbf{w} - y)) \quad = \frac{1}{2N} RSS(\mathbf{w})$$

- Both RSS and J are convex function (as was $x^2$), so we can use the gradient to find the minimum by taking a sequence of steps. Here is the derivative for the J function wrt the parameters:

$$\frac{\partial J(\mathbf{w})}{\partial w_0} = \frac{1}{N}\sum_{i=1}^{N}(w_0 x_0^{(i)} + w_1 x_1^{(i)} + w_2 x_2^{(i)} + \cdots + w_j x_j^{(i)} + \cdots + w_d x_d^{(i)} - y_i)x_0^{(i)} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{w}^T x^{(i)} - y^{(i)})x_0^{(i)}$$

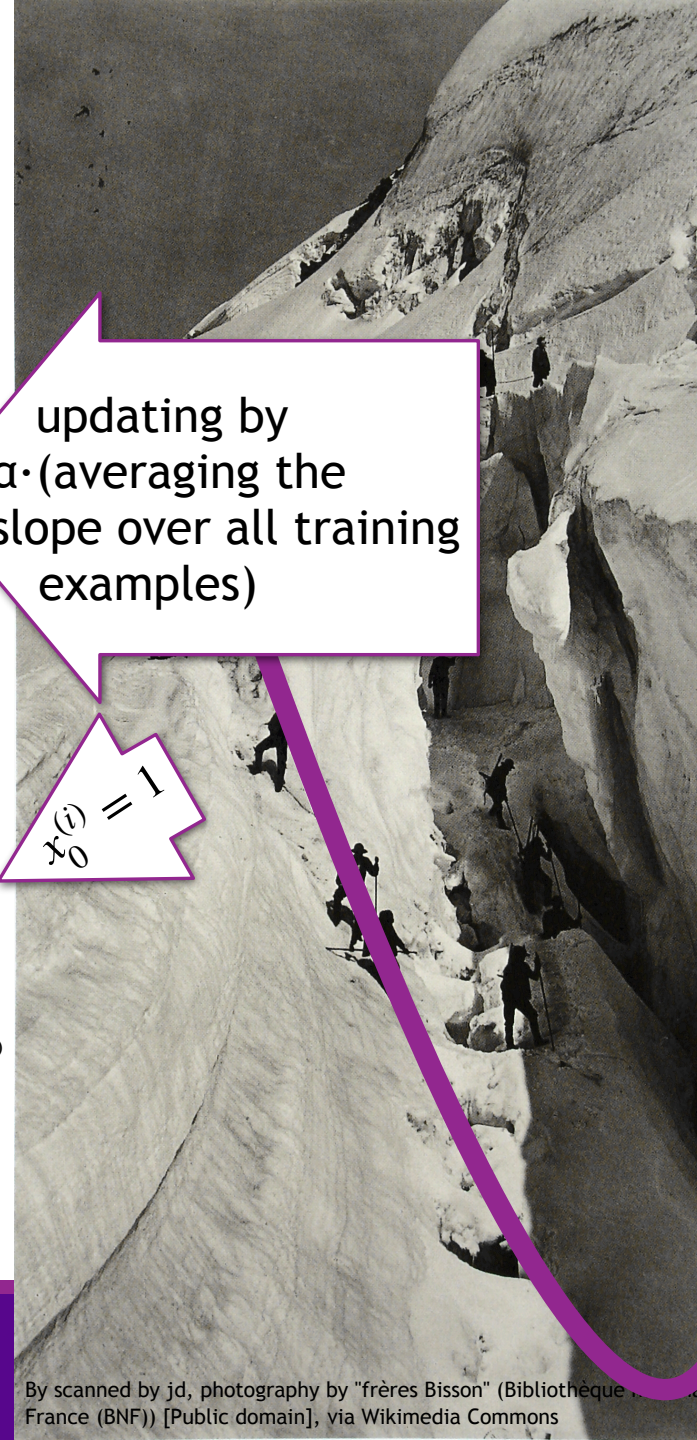
$$\frac{\partial J(\mathbf{w})}{\partial w_j} = \frac{1}{N}\sum_{i=1}^{N}(w_0 x_0^{(i)} + w_1 x_1^{(i)} + w_2 x_2^{(i)} + \cdots + w_j x_j^{(i)} + \cdots + w_d x_d^{(i)} - y^{(i)})x_j^{(i)} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{w}^T x^{(i)} - y^{(i)})x_j^{(i)}$$

individual slope

average slope

updating by α·(averaging the slope over all training examples)

$x_0^{(i)} = 1$



By scanned by jd, photography by "frères Bisson" (Bibliothèque … France (BNF)) [Public domain], via Wikimedia Commons



A very nice explaination:
http://theory.stanford.edu/~tim/s16/l/l5.pdf
http://theory.stanford.edu/~tim/s16/l/l6.pdf

# Gradient Descent Optimization

- The gradient is: $\frac{\partial J(\mathbf{w})}{\partial w_j} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{w}^T\mathbf{x}^{(i)} - y^{(i)})x_j^{(i)}$
- To decrease the cost, we update the parameters, $\mathbf{w} = [w_0, w_1, \ldots, w_d]$, using the update rule:

for $i = 1$ to num_iter

$$temp0 = w_0 - \alpha\frac{\partial J(\mathbf{w})}{\partial w_0} = w_0 - \frac{\alpha}{N}\sum_{i=1}^{N}(w^T\mathbf{x}^{(i)} - y^{(i)})x_0^{(i)}$$

$x_0^{(i)} = 1$

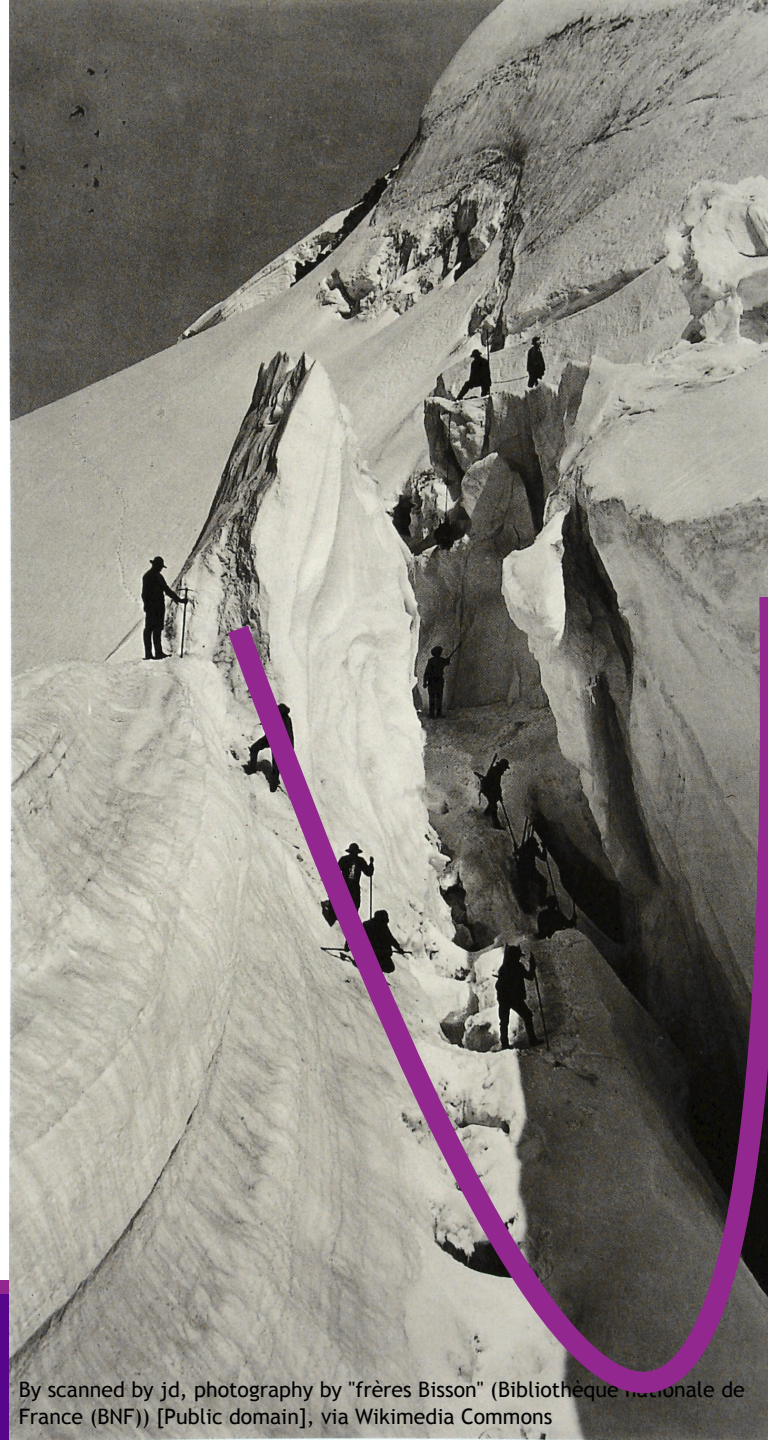
$$temp1 = w_1 - \alpha\frac{\partial J(\mathbf{w})}{\partial w_1} = w_1 - \frac{\alpha}{N}\sum_{i=1}^{N}(w^T\mathbf{x}^{(i)} - y^{(i)})x_1^{(i)}$$

$$\vdots$$

$$tempd = w_d - \alpha\frac{\partial J(\mathbf{w})}{\partial w_d} = w_d - \frac{\alpha}{N}\sum_{i=1}^{N}(\mathbf{w}^T x^{(i)} - y^{(i)})x_d^{(i)}$$

$$w_0 = temp0$$
$$w_1 = temp1$$
$$\vdots$$
$$w_d = tempd$$

Simultaneous update



By scanned by jd, photography by "frères Bisson" (Bibliothèque nationale de France (BNF)) [Public domain], via Wikimedia Commons

# Vectorized Implementation

$$\hat{y}^{(i)} = w_0 \cdot 1 + w_1 \cdot x_1^{(i)} + w_2 \cdot x_2^{(i)} + \cdots + w_d \cdot x_d^{(i)}$$

We want to compute:

$$\frac{\partial J(\mathbf{w})}{\partial w_j} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{w}^T\mathbf{x}^{(i)} - y^{(i)})x_j^{(i)} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}^{(i)} - y^{(i)})x_j^{(i)}$$

Remember how to compute

$$\hat{\mathbf{y}} = X\mathbf{w} = \begin{bmatrix} 1 & x_1^{(1)} & \dots & x_d^{(1)} \\ 1 & x_1^{(2)} & \dots & x_d^{(2)} \\ \vdots & \vdots & & \vdots \\ 1 & x_1^{(N)} & \dots & x_d^{(N)} \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{d-1} \\ w_d \end{bmatrix} = \begin{bmatrix} \hat{y}^{(1)} \\ \hat{y}^{(2)} \\ \vdots \\ \hat{y}^{(N-1)} \\ \hat{y}^{(N)} \end{bmatrix}$$

$$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = \frac{1}{N}\begin{bmatrix} 1 & 1 & \dots & 1 \\ x_1^{(1)} & x_1^{(2)} & \dots & x_1^{(N)} \\ \vdots & \vdots & & \vdots \\ x_d^{(1)} & x_d^{(2)} & \dots & x_d^{(N)} \end{bmatrix}\begin{bmatrix} \hat{y}^{(1)} - y^{(1)} \\ \hat{y}^{(2)} - y^{(2)} \\ \vdots \\ \hat{y}^{(N)} - y^{(N)} \end{bmatrix} = \begin{bmatrix} \frac{\partial J(\mathbf{w})}{\partial w_0} \\ \frac{\partial J(\mathbf{w})}{\partial w_1} \\ \vdots \\ \frac{\partial J(\mathbf{w})}{\partial w_d} \end{bmatrix} = \frac{1}{N}X^T(X\mathbf{w} - \mathbf{y})$$

# Gradient Descent Optimization Using Vectorization $\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = \frac{1}{N} X^T (X\mathbf{w} - \mathbf{y})$

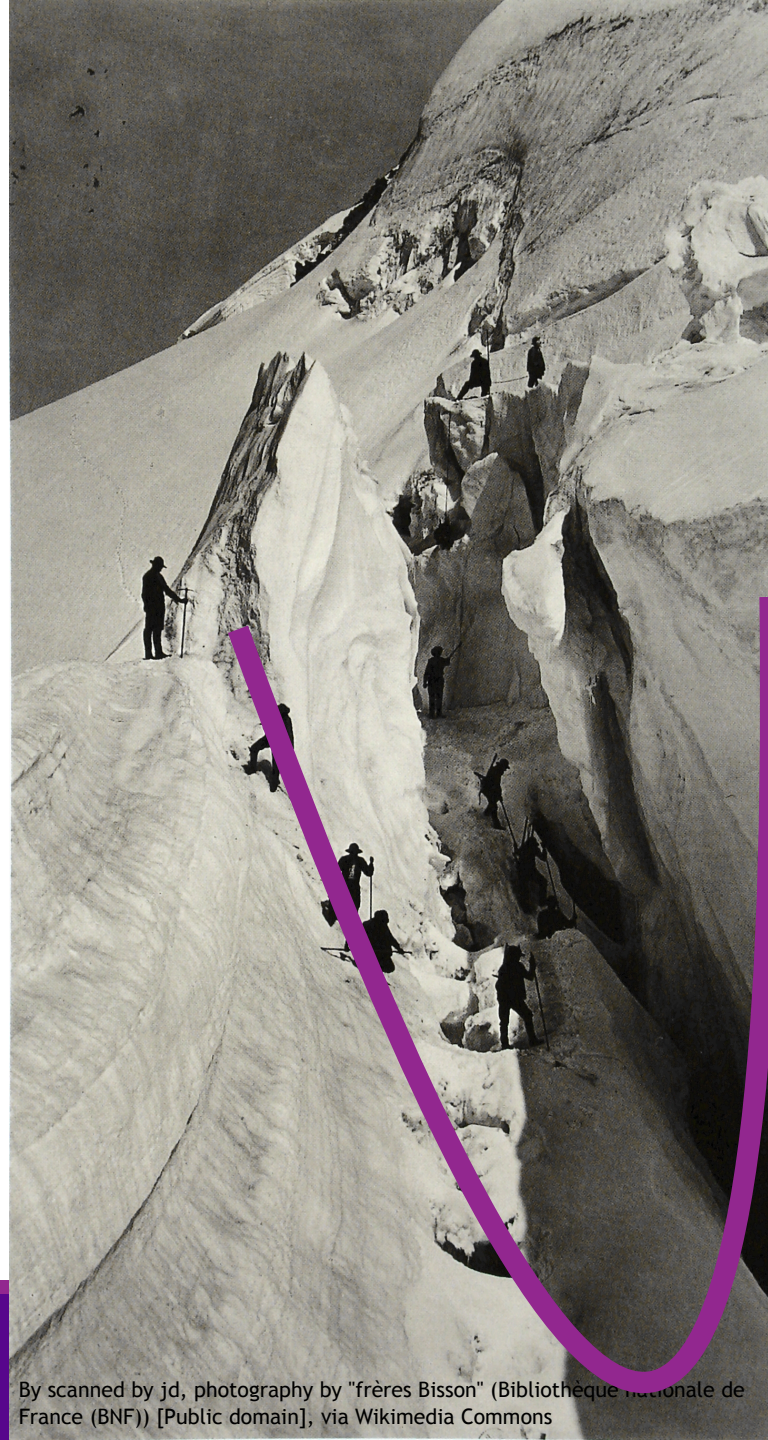
- We can perform gradient update for all parameters $\mathbf{w} = [w_0, w_1, \ldots, w_d]$ in a single line of vectorized code

for $i = 1$ to num_iter

$$\mathbf{w} = \mathbf{w} - \alpha \frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = \mathbf{w} - \alpha \frac{1}{N} X^T (X\mathbf{w} - \mathbf{y})$$

Simultaneous update

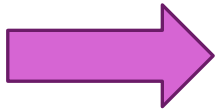
- Note that:
  - $\mathbf{w}$ and $\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}}$ are $d \times 1$ column vectors
  - all $\mathbf{w}$ are getting updated simultaneously and we do not need to store them in temporary variable



By scanned by jd, photography by "frères Bisson" (Bibliothèque nationale de France (BNF)) [Public domain], via Wikimedia Commons

# Outline

- Motivating Example:  Understanding glucose levels in diabetes patients
- Multiple variable linear models
- Least squares solutions
  - Normal Equations
  - Gradient descent
  - Feature scaling
- Evaluating our hypothesis
- Extensions
- Removing features



# Feature Scaling
# AKA Data Normalization

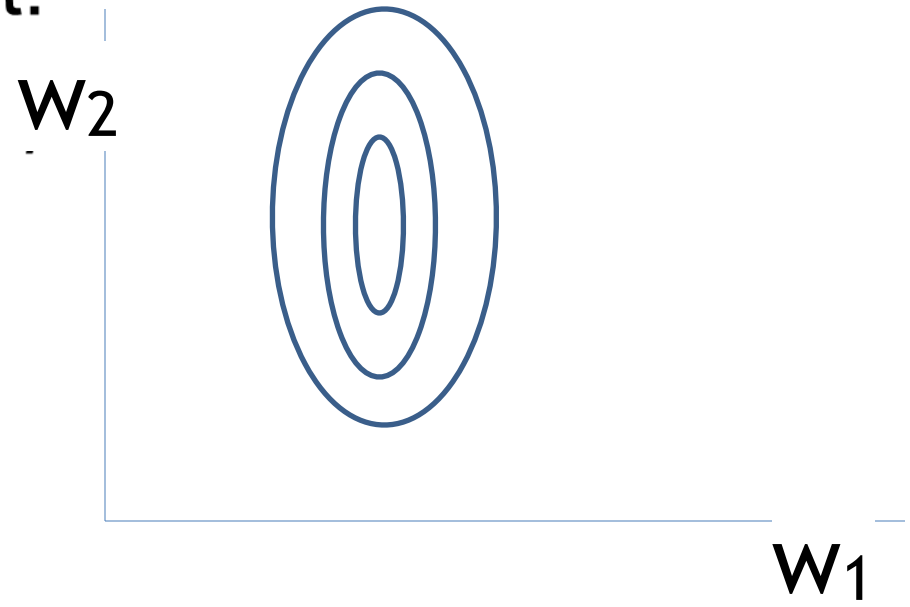
- Before applying many machine learning algorithms make sure that the features that are on a similar scale to prevent one feature from overly influencing the algorithm.
- Feature scaling is typically done before performing gradient descent to improve the rate the algorithm converges
- Eg: Say you are using 2 features for predicting housing price problem:
  - $X_1$ = size (0 – 4000 sq ft)
  - $X_2$ = No. of bedrooms (1 – 5)

# Types of Feature Scaling
# AKA Data Normalization

- Min/Max Normalization
- Mean Centering
- Scaling to Unit Length
- For other methods see https://en.wikipedia.org/wiki/Feature_scaling

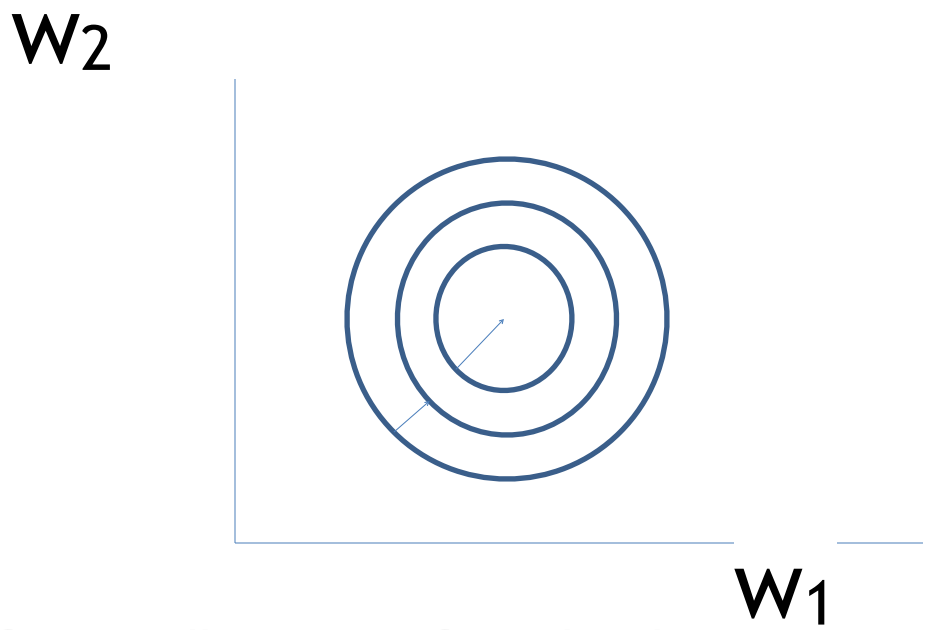
# Cost Function

- Visualizing cost function using a contour plot.



# Min/Max Scaling

- $X_1$ = Size(sq ft) / 4000. So, $0 \leq X_1 \leq 1$.
- $X_2$ = No. of bedrooms / 5. So, $0 \leq X_2 \leq 1$.




- After scaling, cost function becomes spherical or we can say contours become circular.

# Min/Max Normalization

- Scale the range of features to become [0,1] ( or [-1,1] )
- Steps:

  For each feature, j, in the data
  - Find the range of values [min, max] for that feature
  - Update every example's jth feature: $x^{(i)}_j = (x^{(i)}_j - \text{min})/(\text{max}-\text{min})$

Eg: If the range of feature 1 is [0, 4000]

update all training example such that $x^{(i)}_1 = (x^{(i)}_1 - 0)/4000$. Now the range of $x^{(i)}_1$ is [0,1]

If the range if the second feature is [1,5] update all training examples $x^{(i)}_2 = (x^{(i)}_2 - 1)/4$. Now the range of the second feature is [0, 1]

# Mean Centering

- Replace x with x-μ to make features have 0 mean. (Don't apply to $x_0$ =1)
- Perform this for each feature separately
- For example if one of your features was the number of rooms
  - find the average (aka mean) of the number of room in your training examples (average 2.3 rooms)
  - subtract this value from the feature (number of rooms) from every training example( i.e. subtract 2.3 from the number of rooms for each example)

# Scaling to Unit Length

- Scale each feature so the feature vector has unit length

# Mean Centering and Scaling to Unit Length example

Diabetes data: "these data are first standardized to have zero mean and unit L2 norm before they are used in the examples."

Original Data

| age | sex | bmi | map | tc | ldl | hdl | tch | ltg | glu |
|---|---|---|---|---|---|---|---|---|---|
| 59 | 2 | 32.1 | 101 | 157 | 93.2 | 38 | 4 | 4.8598 | 87 |
| 48 | 1 | 21.6 | 87 | 183 | 103.2 | 70 | 3 | 3.8918 | 69 |
| 72 | 2 | 30.5 | 93 | 156 | 93.6 | 41 | 4 | 4.6728 | 85 |
| 24 | 1 | 25.3 | 84 | 198 | 131.4 | 40 | 5 | 4.8903 | 89 |
| 50 | 1 | 23 | 101 | 192 | 125.4 | 52 | 4 | 4.2905 | 80 |
| 23 | 1 | 22.6 | 89 | 139 | 64.8 | 61 | 2 | 4.1897 | 68 |
| 36 | 2 | 22 | 90 | 160 | 99.6 | 50 | 3 | 3.9512 | 82 |

$X =$

| | age | sex | bmi | map | tc | ldl | hdl | tch | ltg | glu |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.038 | 0.051 | 0.062 | 2.187e-02 | -0.044 | -3.482e-02 | 0.043 | -0.003 | 0.020 | -0.018 |
| 1 | -0.002 | -0.045 | -0.051 | -2.633e-02 | -0.008 | -1.916e-02 | -0.074 | -0.039 | -0.068 | -0.092 |
| 1 | 0.085 | 0.051 | 0.044 | -5.670e-03 | -0.046 | -3.419e-02 | 0.032 | -0.003 | 0.003 | -0.026 |
| 1 | -0.089 | -0.045 | -0.012 | -3.666e-02 | 0.012 | 2.499e-02 | 0.036 | 0.034 | 0.023 | -0.009 |
| 1 | 0.005 | -0.045 | -0.036 | 2.187e-02 | 0.004 | 1.560e-02 | -0.008 | -0.003 | -0.032 | -0.047 |
| 1 | -0.093 | -0.045 | -0.041 | -1.944e-02 | -0.069 | -7.929e-02 | -0.041 | -0.076 | -0.041 | -0.096 |
| 1 | -0.046 | 0.051 | -0.047 | -1.600e-02 | -0.040 | -2.480e-02 | -0.001 | -0.039 | -0.063 | -0.038 |

Data from https://web.stanford.edu/~hastie/Papers/LARS/diabetes.data

work area

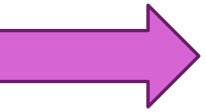
| AGE | | mean | | |
|---|---|---|---|---|
| 59 | - | 48.5 | = | 10.5 |
| 72 | | 48.5 | | -0.5 |
| 24 | | 48.5 | | 23.5 |
| 50 | | 48.5 | | -24.5 |
| 23 | | 48.5 | | 1.5 |
| 36 | | 48.5 | | -25.5 |
| | | | | -12.5 |

l2 norm of the age feature is 275.3

| | | |
|---|---|---|
| 10.5/275.3 | = | 0.038 |
| -0.5/275.3 | | -0.002 |
| 23.5/275.3 | | 0.085 |
| -24.5/275.3 | | -0.089 |
| 1.5/275.3 | | 0.0054 |
| -25.5/275.3 | | -0.093 |
| -12.5/275.3 | | -0.045 |

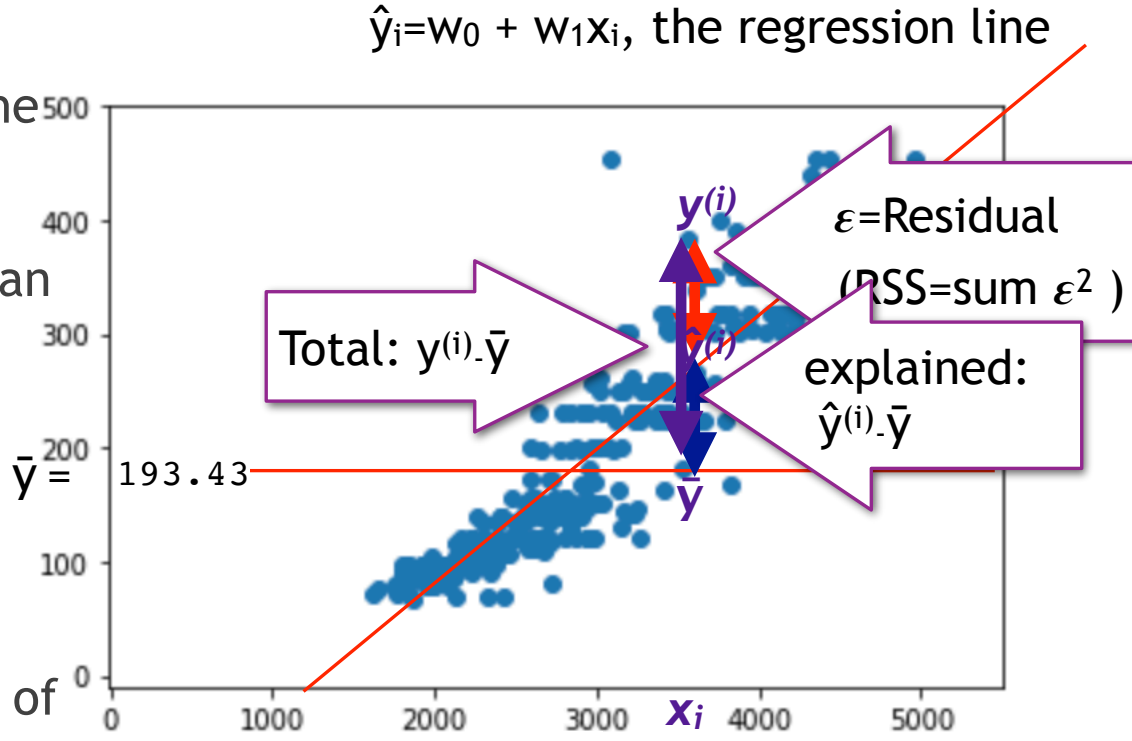
# Outline

- Motivating Example:  Understanding glucose levels in diabetes patients
- Multiple variable linear models
- Least squares solutions
  - Normal Equations
  - Gradient descent
  - Feature scaling
- Evaluating our hypothesis
- Extensions



# Understanding the Error

- RSS $= \sum_{i=1}^{N} (y^{(i)} - \hat{y}^{(i)})^2$ how much the $y^{(i)}$ differs around the prediction line
- ESS $= \sum_{i=1}^{N} (\hat{y}^{(i)} - \bar{y})^2$ how much the estimate differs from the the mean
- TSS $= \sum_{i=1}^{N} (y^{(i)} - \bar{y})^2$ how much the $y^{(i)}$ varies from the mean
- TSS = RSS + ESS (a proof can be found at https://en.wikipedia.org/wiki/Explained_sum_of_squares)
- If TSS is close to RSS then most of the error is due to random variation
- $R^2$ =1-RSS/TSS= (TSS - RSS)/TSS = ESS/TSS is the amount of error explained
- $0 \leq R^2 \leq 1$




RSS/TSS is amount of variation not explained by hypothesis

# $R^2$ intuition, "goodness of fit"

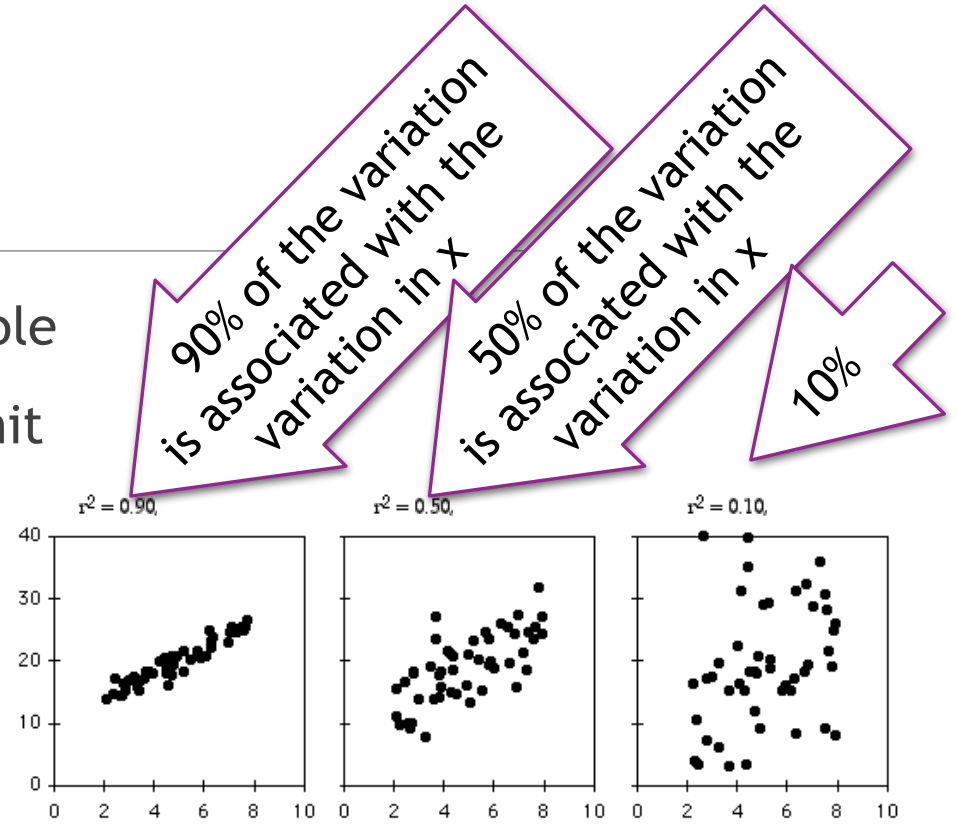
- We used linear regression to find the best estimate for our coefficients
- Computing the RSS (residual sum of squares) gives an intuition on how much error our model has.
- We can represent how well our models is doing as a percentage of reduction of the original prediction error.

$$R^2 = \text{Explained variation/total variation} = \frac{TSS - RSS}{TSS} = 1 - \frac{RSS}{TSS} \qquad TSS = \sum_{i=1}^{n}(y_i - \bar{y})^2 \qquad RSS = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

- $R^2$ is called the *coefficient of determination*
- $R^2$ is always between 0 and 1 (0 meaning the models does not explain any of the variation and 1 meaning the model explains all of the variation
- The higher the $R^2$ score, the better the model fits the data
- Limitations:
  - $R^2$ cannot determine if the coefficient estimates and predictions are biased - you must look at the residual plots
  - A low score might be a good model (e.g. attempting to model human behavior typically has $R^2$ less than 50%)

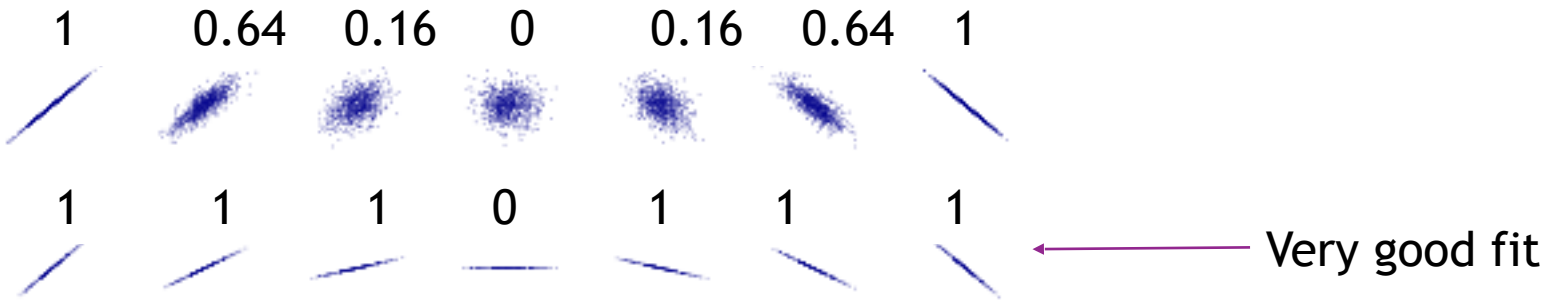
# Minimum RSS

- MSE = (1/N)·RSS is in squared units of the response variable
- MSE is hard to interpret without the context of what a unit means
- $R^2$ is easy to interpret without knowing the units of the response variable
- $R^2$ explains portion of variance in y explained by x

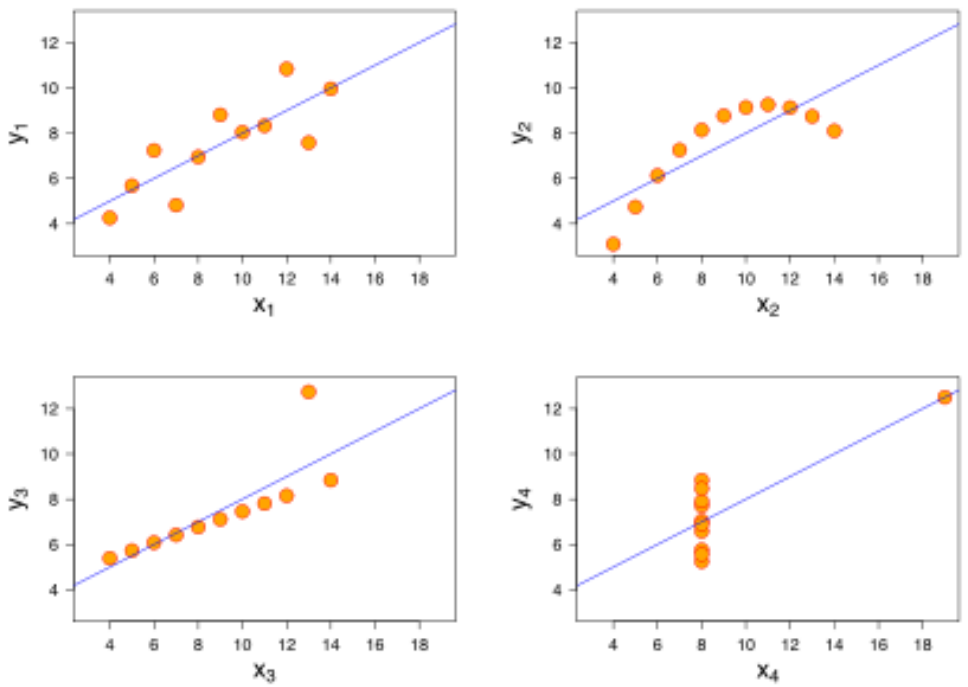


# Visually seeing correlation

- $R^2 \approx 1$ Linear model is a ver good fit
- $R^2 \approx 0$ Linear model is a poor fit



# When the Error is Large…




- Many sources of error for a linear model
- Always good to visually inspect the scatter plot
  - Look for trends
- Example to the left
  - All four data sets have same regression line
  - But, errors and their reasons are different
- How would you describe these errors?

# Simple Example

- From: http://stattrek.com/regression/regression-example.aspx?Tutorial=AP
  - Very nice simple problems
- Predict aptitude on one test from an earlier test
- Draw a scatter plot and regression line

## How to Find the Regression Equation

In the table below, the $x_i$ column shows scores on the aptitude test. Similarly, the $y_i$ column shows statistics grades. The last two rows show sums and mean scores that we will use to conduct the regression analysis.

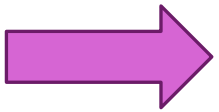
| | Student | $x_i$ | $y_i$ | $(x_i - \bar{x})$ | $(y_i - \bar{y})$ | $(x_i - \bar{x})^2$ | $(y_i - \bar{y})^2$ | $(x_i - \bar{x})(y_i - \bar{y})$ |
|---|---|---|---|---|---|---|---|---|
| | 1 | 95 | 85 | 17 | 8 | 289 | 64 | 136 |
| | 2 | 85 | 95 | 7 | 18 | 49 | 324 | 126 |
| | 3 | 80 | 70 | 2 | -7 | 4 | 49 | -14 |
| | 4 | 70 | 65 | -8 | -12 | 64 | 144 | 96 |
| | 5 | 60 | 70 | -18 | -7 | 324 | 49 | 126 |
| **Sum** | | 390 | 385 | | | 730 | 630 | 470 |
| **Mean** | | 78 | 77 | | | | | |

The regression equation is a linear equation of the form: $\hat{y} = b_0 + b_1 x$ . To conduct a regression analysis, we need to solve for $b_0$ and $b_1$. Computations are shown below.

| | |
|---|---|
| $b_1 = \Sigma [ (x_i - \bar{x})(y_i - \bar{y}) ] / \Sigma [ (x_i - \bar{x})^2 ]$<br>$b_1 = 470/730 = 0.644$ | $b_0 = \bar{y} - b_1 * \bar{x}$<br>$b_0 = 77 - (0.644)(78) = 26.768$ |

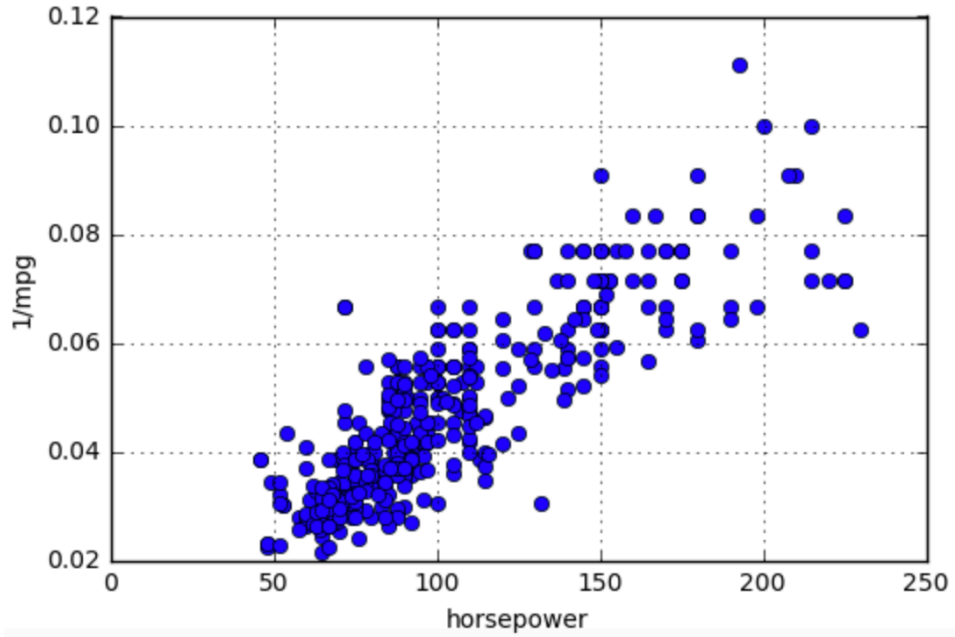
# Outline

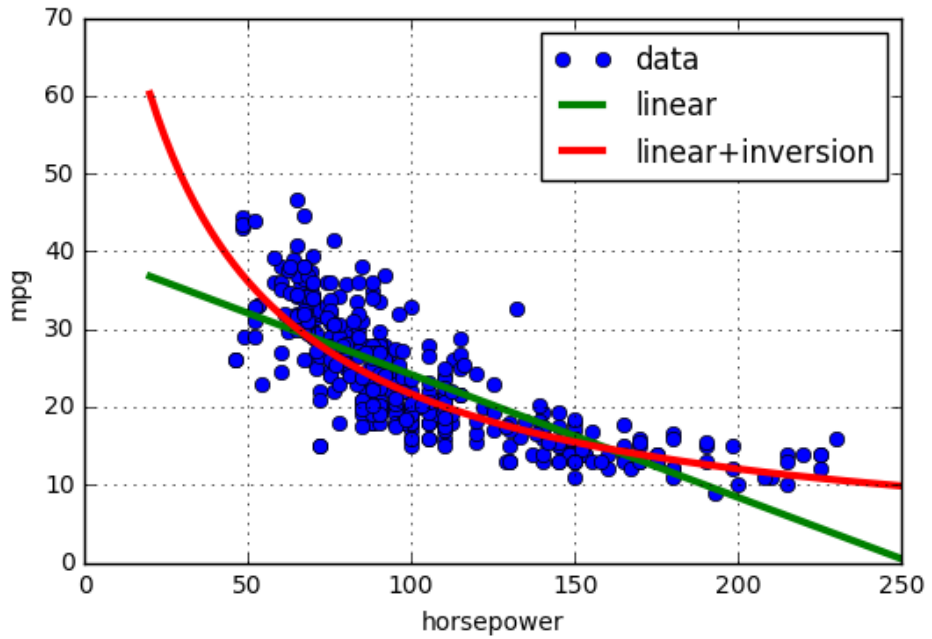
- Motivating Example:  Understanding glucose levels in diabetes patients
- Multiple variable linear models
- Least squares solutions
  - Normal Equations
  - Gradient descent
  - Feature scaling
- Evaluating our hypothesis
- Extensions



# A Better Model for the Auto Example

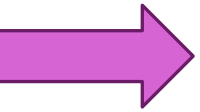
- Fit the inverse: $\frac{1}{\text{mpg}} = w_0 + w_1 \text{ horsepower}$
- Uses a nonlinear transformation
- Will cover transforming the data later



# Outline

- Motivating Example:  Understanding glucose levels in diabetes patients
- Multiple variable linear models
- Least squares solutions
  - Normal Equations
  - Gradient descent
  - Feature scaling
- Evaluating our hypothesis
- Special case:  Simple linear regression
- Extensions
  - Removing features



# Which features to select

- Subset selection: Identify a subset of the k predictors we believe are associated with the response. Then the least squares solution can be fit on the reduced set of variables
- Try all $2^k$-1 subsets of the features and select the best one
- We will use adjusted $R^2$ statistics to compare models when the number of features varies. ($R^2$ will go down when the number of features increases even if the features do not help predict the outcome!)

$$R^2_{adj} = 1 - \frac{RSS/(n-k-1)}{TSS/(n-1)}$$

- Where k equals the number of predictors in the model

# Subset Selection

$$R^2_{adj} = 1 - \frac{RSS/(n-q-1)}{TSS/(n-1)}$$

- Subset selection: Identify a subset of the k predictors we believe are associated with the response. Then the least squares solution can be fit on the reduced set of variables

**Algorithm 6.1** *Best subset selection*

1. Let $\mathcal{M}_0$ denote the *null model*, which contains no predictors. This model simply predicts the sample mean for each observation.
2. For $k = 1, 2, \ldots p$:
   (a) Fit all $\binom{p}{k}$ models that contain exactly $k$ predictors.
   (b) Pick the best among these $\binom{p}{k}$ models, and call it $\mathcal{M}_k$. Here *best* is defined as having the smallest RSS, or equivalently largest $R^2$.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, $C_p$ (AIC), BIC, or adjusted $R^2$.

Some of the figures in this presentation are taken from "An Introduction to Statistical Learning, with applications in R" (Springer, 2013) with permission from the authors: G. James, D. Witten, T. Hastie and R. Tibshirani

# Backward Selection

$$R^2_{adj} = 1 - \frac{RSS/(n-q-1)}{TSS/(n-1)}$$

- Backward elimination starts with all k predictors in the model
- Then repeatedly deletes the least significant predictor

**Algorithm 6.3** *Backward stepwise selection*

1. Let $\mathcal{M}_p$ denote the *full* model, which contains all $p$ predictors.
2. For $k = p, p-1, \ldots, 1$:
   (a) Consider all $k$ models that contain all but one of the predictors in $\mathcal{M}_k$, for a total of $k-1$ predictors.
   (b) Choose the *best* among these $k$ models, and call it $\mathcal{M}_{k-1}$. Here *best* is defined as having smallest RSS or highest $R^2$.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, $C_p$ (AIC), BIC, or adjusted $R^2$.

Some of the figures in this presentation are taken from "An Introduction to Statistical Learning, with applications in R" (Springer, 2013) with permission from the authors: G. James, D. Witten, T. Hastie and R. Tibshirani

# Forward Selection

$$R^2_{adj} = 1 - \frac{RSS/(n-q-1)}{TSS/(n-1)}$$

- Forward selections starts with no predictors predictors in the model
- Then repeatedly adds the most significant predictor

**Algorithm 6.2** *Forward stepwise selection*

1. Let $\mathcal{M}_0$ denote the *null* model, which contains no predictors.
2. For $k = 0, \ldots, p-1$:
   (a) Consider all $p-k$ models that augment the predictors in $\mathcal{M}_k$ with one additional predictor.
   (b) Choose the *best* among these $p-k$ models, and call it $\mathcal{M}_{k+1}$. Here *best* is defined as having smallest RSS or highest $R^2$.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, $C_p$ (AIC), BIC, or adjusted $R^2$.

Some of the figures in this presentation are taken from "An Introduction to Statistical Learning, with applications in R" (Springer, 2013) with permission from the authors: G. James, D. Witten, T. Hastie and R. Tibshirani