EE4077 Fundamentals of Machine Learning

Linear Regression – I

Fall 2021

EE–Marmara University

Outline

Linear Regression

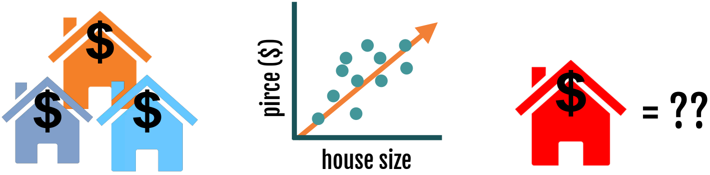
Recap of MLE/MAP

Linear Regression

- Motivation
- Algorithm
- Univariate solution
- Multivariate Solution
- Probabilistic interpretation
- Computational and numerical optimization

Task 1: Regression

How much should you sell your house for?



input: houses & features **learn**: $x \to y$ relationship **predict**: y (*continuous*)

Course Covers: Linear/Ridge Regression, Loss Function, SGD, Feature Scaling, Regularization, Cross Validation

Supervised Learning

Supervised learning

In a supervised learning problem, you have access to input variables (X) and outputs (Y), and the goal is to predict an output given an input

- Examples:
  - **Housing prices (Regression)**: predict the price of a house based on features (size, location, etc)
  - **Cat vs. Dog (Classification)**: predict whether a picture is of a cat or a dog

Regression

Predicting a continuous outcome variable:

- Predicting a company's future stock price using its profit and other financial info
- Predicting annual rainfall based on local flora and fauna
- Predicting distance from a traffic light using LIDAR measurements

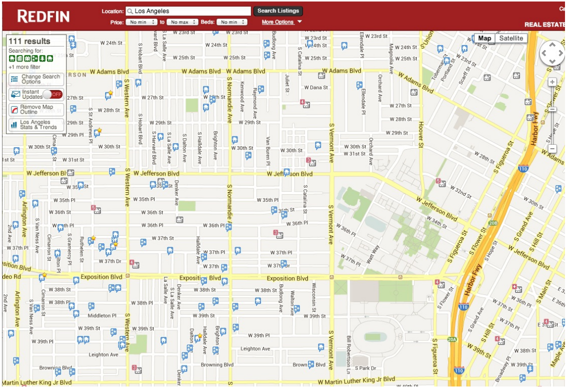
Magnitude of the error matters:

- We can measure 'closeness' of prediction and labels, leading to different ways to evaluate prediction errors.
  - Predicting stock price: better to be off by 1$ than by 20$
  - Predicting distance from a traffic light: better to be off 1 m than by 10 m
- We should choose learning models and algorithms accordingly.

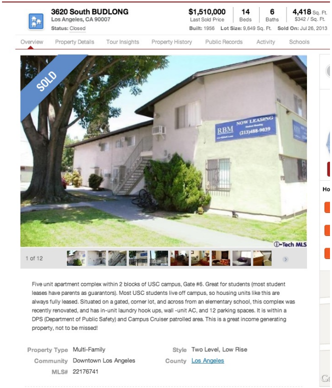
Ex: predicting the sale price of a house

Retrieve historical sales records

(This will be our training data)



Features used to predict


3620 South BUDLONG
Los Angeles, CA 90007
Status: Closed

$1,510,000 Last Sold Price | 14 Beds | 6 Baths | 4,418 Sq. Ft. $342 / Sq. Ft.
Built: 1956 Lot Size: 9,649 Sq. Ft. Sold On: Jul 26, 2013

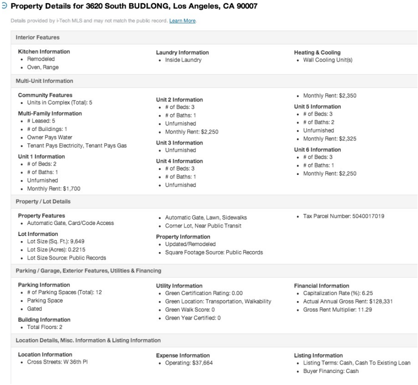
Overview | Property Details | Tour Insights | Property History | Public Records | Activity | Schools

SOLD

1 of 12

Five unit apartment complex within 2 blocks of USC campus, Gate #6. Great for students (most student leases have parents as guarantors). Most USC students live off campus, so housing units like this are always fully leased. Situated on a gated, corner lot, and across from an elementary school, this complex was recently renovated, and has in-unit laundry hook ups, wall -unit AC, and 12 parking spaces. It is within a DPS (Department of Public Safety) and Campus Cruiser patrolled area. This is a great income generating property, not to be missed!

Property Type: Multi-Family
Style: Two Level, Low Rise
Community: Downtown Los Angeles
County: Los Angeles
MLS#: 22176741


Property Details for 3620 South BUDLONG, Los Angeles, CA 90007

Details provided by i-Tech MLS and may not match the public record. Learn More

Interior Features

Kitchen Information
- Remodeled
- Oven, Range

Laundry Information
- Inside Laundry

Heating & Cooling
- Wall Cooling Unit(s)

Multi-Unit Information

Community Features
- Units in Complex (Total): 5

Multi-Family Information
- # Leased: 5
- # of Buildings: 1
- Owner Pays Water
- Tenant Pays Electricity, Tenant Pays Gas

Unit 1 Information
- # of Beds: 2
- # of Baths: 1
- Unfurnished
- Monthly Rent: $1,700

Unit 2 Information
- # of Beds: 3
- # of Baths: 1
- Unfurnished
- Monthly Rent: $2,250

Unit 3 Information
- Unfurnished

Unit 4 Information
- # of Beds: 3
- # of Baths: 1
- Unfurnished
- Monthly Rent: $2,350

Unit 5 Information
- # of Beds: 3
- # of Baths: 2
- Unfurnished
- Monthly Rent: $2,325

Unit 6 Information
- # of Beds: 3
- # of Baths: 1
- Monthly Rent: $2,250

Property / Lot Details

Property Features
- Automatic Gate, Card/Code Access
- Automatic Gate, Lawn, Sidewalks
- Corner Lot, Near Public Transit

Lot Information
- Lot Size (Sq. Ft.): 9,649
- Lot Size (Acres): 0.2215
- Lot Size Source: Public Records

Property Information
- Updated/Remodeled
- Square Footage Source: Public Records
- Tax Parcel Number: 5040017019

Parking / Garage, Exterior Features, Utilities & Financing

Parking Information
- # of Parking Spaces (Total): 12
- Parking Space
- Gated

Building Information
- Total Floors: 2

Utility Information
- Green Certification Rating: 0.00
- Green Location: Transportation, Walkability
- Green Walk Score: 0
- Green Year Certified: 0

Financial Information
- Capitalization Rate (%): 6.25
- Actual Annual Gross Rent: $128,331
- Gross Rent Multiplier: 11.29

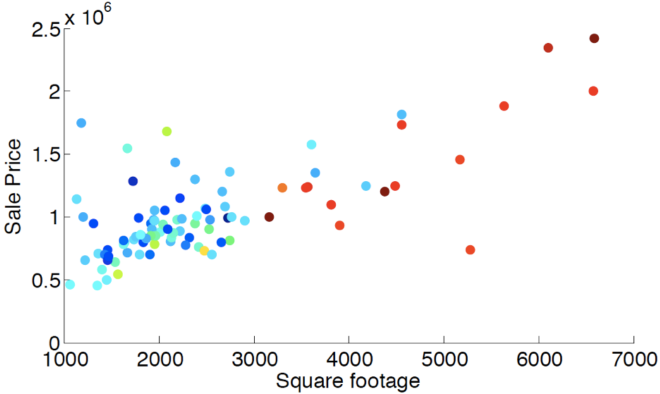
Location Details, Misc. Information & Listing Information

Location Information
- Cross Streets: W 36th Pl

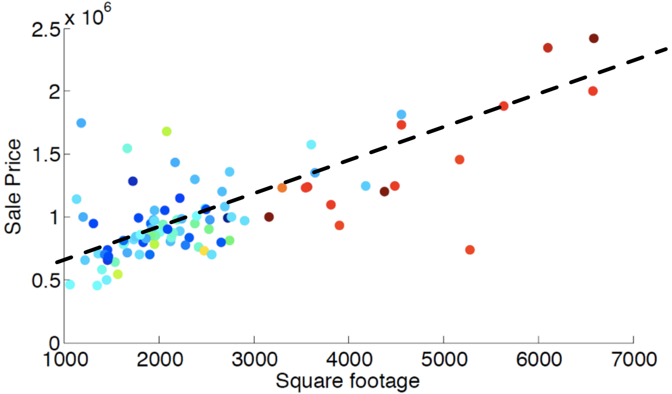
Expense Information
- Operating: $37,664

Listing Information
- Listing Terms: Cash, Cash To Existing Loan
- Buyer Financing: Cash

Correlation between square footage and sale price

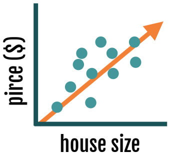


Roughly linear relationship



Sale price ≈ price_per_sqft × square_footage + fixed_expense

Data Can be Compactly Represented by Matrices



- Learn parameters (w_0, w_1) of the orange line $y = w_1 x + w_0$

Sq.ft

$$\text{House 1: } 1000 \times w_1 + w_0 = 200,000$$

$$\text{House 2: } 2000 \times w_1 + w_0 = 350,000$$

- Can represent compactly in matrix notation

$$\begin{bmatrix} 1000 & 1 \\ 2000 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_0 \end{bmatrix} = \begin{bmatrix} 200,000 \\ 350,000 \end{bmatrix}$$

Some Concepts That You Should Know

- Invertibility of Matrices and Computing Inverses
- Vector Norms – L2, Frobenius etc., Inner Products
- Eigenvalues and Eigen-vectors
- Singular Value Decomposition
- Covariance Matrices and Positive Semi-definite-ness

Excellent Resources:

- Essence of Linear Algebra YouTube Series
- Prof. Gilbert Strang's course at MIT

Matrix Inverse

- Let us solve the house-price prediction problem

$$\begin{bmatrix} 1000 & 1 \\ 2000 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_0 \end{bmatrix} = \begin{bmatrix} 200,000 \\ 350,000 \end{bmatrix} \tag{1}$$

$$\begin{bmatrix} w_1 \\ w_0 \end{bmatrix} = \left(\begin{bmatrix} 1000 & 1 \\ 2000 & 1 \end{bmatrix} \right)^{-1} \begin{bmatrix} 200,000 \\ 350,000 \end{bmatrix} \tag{2}$$

$$= \frac{1}{-1000} \begin{bmatrix} 1 & -1 \\ -2000 & 1000 \end{bmatrix} \begin{bmatrix} 200,000 \\ 350,000 \end{bmatrix} \tag{3}$$

$$= \frac{1}{-1000} \begin{bmatrix} 150,000 \\ -5 \times 10^7 \end{bmatrix} \tag{4}$$

$$\begin{bmatrix} w_1 \\ w_0 \end{bmatrix} = \begin{bmatrix} 150 \\ 50,000 \end{bmatrix} \tag{5}$$

You could have data from many houses

- Sale_price = price_per_sqft $\times$ square_footage + fixed_expense + unexplainable_stuff
- Want to learn the price_per_sqft and fixed_expense
- Training data: past sales record.

sqft	sale price
2000	800K
2100	907K
1100	312K
5500	2,600K
...	...

Problem: there isn't a $\mathbf{w} = [w_1, w_0]^T$ that will satisfy all equations

Want to predict the best price_per_sqft and fixed_expense

- Sale_price = price_per_sqft × square_footage + fixed_expense + unexplainable_stuff
- Want to learn the price_per_sqft and fixed_expense
- Training data: past sales record.

sqft	sale price	prediction
2000	810K	720K
2100	907K	800K
1100	312K	350K
5500	2,600K	2,600K
...	...	...

Reduce prediction error

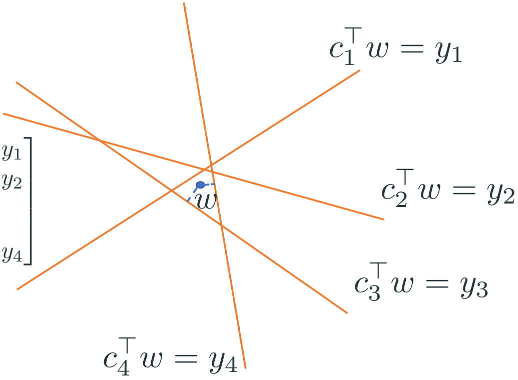
How to measure errors?

- absolute difference: $|\text{prediction} - \text{sale price}|$.
- squared difference: $(\text{prediction} - \text{sale price})^2$ [differentiable!].

sqft	sale price	prediction	abs error	squared error
2000	810K	720K	90K	8100
2100	907K	800K	107K	107^2
1100	312K	350K	38K	38^2
5500	2,600K	2,600K	0	0
...	...			

Geometric Illustration: Each house corresponds to one line

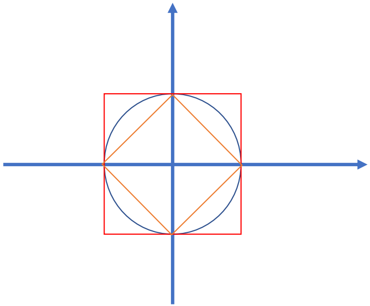
$$r = \begin{bmatrix} c_1^\top w - y_1 \\ c_2^\top w - y_2 \\ \vdots \\ c_4^\top w - y_4 \end{bmatrix} = Aw - y$$



- Want to find $\mathbf{w}$ that minimizes the difference between $\mathbf{Xw}$, $\mathbf{y}$
- But since this a vector, we need an operator that can map the residual vector $r(\mathbf{w}) = \mathbf{y} - \mathbf{Xw}$ to a scalar

Norms and Loss Functions

- A vector norm is any function $f : \mathbb{R}^n \to \mathbb{R}$ with
  - $f(x) \geq 0$ and $f(x) = 0 \iff x = 0$
  - $f(ax) = |a| f(x)$ for $a \in \mathbb{R}$
  - triangle inequality: $f(x+y) \leq f(x) + f(y)$
- e.g., ℓ_2 norm: $\|x\|_2 = \sqrt{x^\top x} = \sqrt{\sum_{i=1}^n x_i^2}$
- e.g., ℓ_1 norm: $\|x\|_1 = \sum_{i=1}^n |x_i|$
- e.g., ℓ_∞ norm: $\|x\|_\infty = \max |x_i|$



from inside to outside: ℓ_1, ℓ_2, ℓ_∞ norm ball.

Minimize squared errors

Our model:

Sale_price =

price_per_sqft $\times$ square_footage + fixed_expense + unexplainable_stuff

Training data:

sqft	sale price	prediction	error	squared error
2000	810K	720K	90K	8100
2100	907K	800K	107K	107^2
1100	312K	350K	38K	38^2
5500	2,600K	2,600K	0	0
...	...			
Total				$8100 + 107^2 + 38^2 + 0 + \cdots$

Aim:

Adjust price_per_sqft and fixed_expense such that the sum of the squared error is minimized — i.e., the unexplainable_stuff is minimized.

Linear regression

Setup:

- **Input**: $\mathbf{x} \in \mathbb{R}^D$ (covariates, predictors, features, etc)
- **Output**: $y \in \mathbb{R}$ (responses, targets, outcomes, outputs, etc)
- **Model**: $f : \mathbf{x} \to y$, with $f(\mathbf{x}) = w_0 + \sum_{d=1}^{D} w_d x_d = w_0 + \mathbf{w}^\top \mathbf{x}$.
  - $\mathbf{w} = [w_1 \ w_2 \ \cdots \ w_D]^\top$: *weights*, *parameters*, or *parameter vector*
  - w_0 is called *bias*.
  - Sometimes, we also call $\tilde{\mathbf{w}} = [w_0 \ w_1 \ w_2 \ \cdots \ w_D]^\top$ parameters.
- **Training data**: $\mathcal{D} = \{(\mathbf{x}_n, y_n), n = 1, 2, \ldots, N\}$

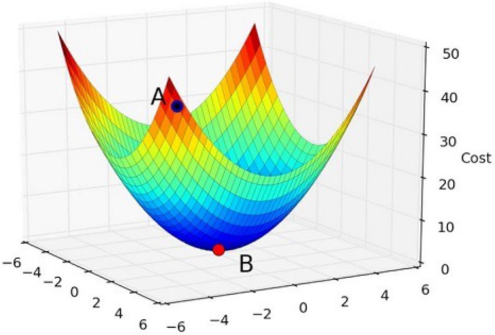
Minimize the Residual sum of squares:

$$RSS(\tilde{\mathbf{w}}) = \sum_{n=1}^{N} [y_n - f(\mathbf{x}_n)]^2 = \sum_{n=1}^{N} [y_n - (w_0 + \sum_{d=1}^{D} w_d x_{nd})]^2$$

A simple case: x is just one-dimensional (D=1)

Residual sum of squares:

$$RSS(\tilde{\mathbf{w}}) = \sum_n [y_n - f(\mathbf{x}_n)]^2 = \sum_n [y_n - (w_0 + w_1 x_n)]^2$$



What kind of function is this? CONVEX (has a unique global minimum)

A simple case: x is just one-dimensional (D=1)

Residual sum of squares:

$$RSS(\tilde{\mathbf{w}}) = \sum_n [y_n - f(\mathbf{x}_n)]^2 = \sum_n [y_n - (w_0 + w_1 x_n)]^2$$

Stationary points:

Take derivative with respect to parameters and set it to zero

$$\frac{\partial RSS(\tilde{\mathbf{w}})}{\partial w_0} = 0 \Rightarrow -2 \sum_n [y_n - (w_0 + w_1 x_n)] = 0,$$

$$\frac{\partial RSS(\tilde{\mathbf{w}})}{\partial w_1} = 0 \Rightarrow -2 \sum_n [y_n - (w_0 + w_1 x_n)] x_n = 0.$$

A simple case: x is just one-dimensional (D=1)

$$\frac{\partial RSS(\tilde{\mathbf{w}})}{\partial w_0} = 0 \Rightarrow -2\sum_n [y_n - (w_0 + w_1 x_n)] = 0$$

$$\frac{\partial RSS(\tilde{\mathbf{w}})}{\partial w_1} = 0 \Rightarrow -2\sum_n [y_n - (w_0 + w_1 x_n)] x_n = 0$$

Simplify these expressions to get the "Normal Equations":

$$\sum y_n = N w_0 + w_1 \sum x_n$$

$$\sum x_n y_n = w_0 \sum x_n + w_1 \sum x_n^2$$

Solving the system we obtain the least squares coefficient estimates:

$$w_1 = \frac{\sum (x_n - \bar{x})(y_n - \bar{y})}{\sum (x_i - \bar{x})^2} \qquad \text{and} \qquad w_0 = \bar{y} - w_1 \bar{x}$$

where $\bar{x} = \frac{1}{N}\sum_n x_n$ and $\bar{y} = \frac{1}{N}\sum_n y_n$.

Example

sqft (1000's)	sale price (100k)
1	2
2	3.5
1.5	3
2.5	4.5

Residual sum of squares:

$$RSS(\tilde{\mathbf{w}}) = \sum_n [y_n - f(\mathbf{x}_n)]^2 = \sum_n [y_n - (w_0 + w_1 x_n)]^2$$

The w_1 and w_0 that minimize this are given by:

$$w_1 = \frac{\sum (x_n - \bar{x})(y_n - \bar{y})}{\sum (x_i - \bar{x})^2} \qquad \text{and} \qquad w_0 = \bar{y} - w_1 \bar{x}$$

where $\bar{x} = \frac{1}{N} \sum_n x_n$ and $\bar{y} = \frac{1}{N} \sum_n y_n$.

Example

sqft (1000's)	sale price (100k)
1	2
2	3.5
1.5	3
2.5	4.5

Residual sum of squares:

$$RSS(\tilde{\mathbf{w}}) = \sum_n [y_n - f(\mathbf{x}_n)]^2 = \sum_n [y_n - (w_0 + w_1 x_n)]^2$$

The w_1 and w_0 that minimize this are given by:

$$w_1 \approx 1.6$$
$$w_0 \approx 0.45$$

Least Mean Squares when $\mathbf{x}$ is D-dimensional

sqft (1000's)	bedrooms	bathrooms	sale price (100k)
1	2	1	2
2	2	2	3.5
1.5	3	2	3
2.5	4	2.5	4.5

$RSS(\tilde{\mathbf{w}})$ in matrix form:

$$RSS(\tilde{\mathbf{w}}) = \sum_n [y_n - (w_0 + \sum_d w_d x_{nd})]^2 = \sum_n [y_n - \tilde{\mathbf{w}}^\top \tilde{\mathbf{x}}_n]^2,$$

where we have redefined some variables (by augmenting)

$$\tilde{\mathbf{x}} \leftarrow [1 \ x_1 \ x_2 \ \ldots \ x_D]^\top, \quad \tilde{\mathbf{w}} \leftarrow [w_0 \ w_1 \ w_2 \ \ldots \ w_D]^\top$$

Least Mean Squares when x is D-dimensional

$RSS(\tilde{\mathbf{w}})$ in matrix form:

$$RSS(\tilde{\mathbf{w}}) = \sum_n [y_n - (w_0 + \sum_d w_d x_{nd})]^2 = \sum_n [y_n - \tilde{\mathbf{w}}^\top \tilde{\mathbf{x}}_n]^2,$$

where we have redefined some variables (by augmenting)

$$\tilde{\mathbf{x}} \leftarrow [1 \ x_1 \ x_2 \ \ldots \ x_D]^\top, \quad \tilde{\mathbf{w}} \leftarrow [w_0 \ w_1 \ w_2 \ \ldots \ w_D]^\top$$

which leads to

$$\begin{aligned} RSS(\tilde{\mathbf{w}}) &= \sum_n (y_n - \tilde{\mathbf{w}}^\top \tilde{\mathbf{x}}_n)(y_n - \tilde{\mathbf{x}}_n^\top \tilde{\mathbf{w}}) \\ &= \sum_n \tilde{\mathbf{w}}^\top \tilde{\mathbf{x}}_n \tilde{\mathbf{x}}_n^\top \tilde{\mathbf{w}} - 2 y_n \tilde{\mathbf{x}}_n^\top \tilde{\mathbf{w}} + \text{const.} \\ &= \left\{ \tilde{\mathbf{w}}^\top \left(\sum_n \tilde{\mathbf{x}}_n \tilde{\mathbf{x}}_n^\top \right) \tilde{\mathbf{w}} - 2 \left(\sum_n y_n \tilde{\mathbf{x}}_n^\top \right) \tilde{\mathbf{w}} \right\} + \text{const.} \end{aligned}$$

$RSS(\tilde{\mathbf{w}})$ in new notations

From previous slide:

$$RSS(\tilde{\mathbf{w}}) = \left\{ \tilde{\mathbf{w}}^\top \left(\sum_n \tilde{\mathbf{x}}_n \tilde{\mathbf{x}}_n^\top \right) \tilde{\mathbf{w}} - 2 \left(\sum_n y_n \tilde{\mathbf{x}}_n^\top \right) \tilde{\mathbf{w}} \right\} + \text{const.}$$

Design matrix and target vector:

$$\tilde{\mathbf{X}} = \begin{pmatrix} \tilde{\mathbf{x}}_1^\top \\ \tilde{\mathbf{x}}_2^\top \\ \vdots \\ \tilde{\mathbf{x}}_N^\top \end{pmatrix} \in \mathbb{R}^{N \times (D+1)}, \quad \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix} \in \mathbb{R}^N$$

Compact expression:

$$RSS(\tilde{\mathbf{w}}) = \|\tilde{\mathbf{X}}\tilde{\mathbf{w}} - \mathbf{y}\|_2^2 = \left\{ \tilde{\mathbf{w}}^\top \tilde{\mathbf{X}}^\top \tilde{\mathbf{X}} \tilde{\mathbf{w}} - 2 \left(\tilde{\mathbf{X}}^\top \mathbf{y} \right)^\top \tilde{\mathbf{w}} \right\} + \text{const}$$

Example: $RSS(\tilde{\mathbf{w}})$ in compact form

sqft (1000's)	bedrooms	bathrooms	sale price (100k)
1	2	1	2
2	2	2	3.5
1.5	3	2	3
2.5	4	2.5	4.5

Design matrix and target vector:

$$\tilde{\mathbf{X}} = \begin{pmatrix} \tilde{\mathbf{x}}_1^\top \\ \tilde{\mathbf{x}}_2^\top \\ \vdots \\ \tilde{\mathbf{x}}_N^\top \end{pmatrix} \in \mathbb{R}^{N\times(D+1)}, \quad \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix} \in \mathbb{R}^N$$

. Compact expression:

$$RSS(\tilde{\mathbf{w}}) = \|\tilde{\mathbf{X}}\tilde{\mathbf{w}} - \mathbf{y}\|_2^2 = \left\{ \tilde{\mathbf{w}}^\top \tilde{\mathbf{X}}^\top \tilde{\mathbf{X}} \tilde{\mathbf{w}} - 2\left(\tilde{\mathbf{X}}^\top \mathbf{y}\right)^\top \tilde{\mathbf{w}} \right\} + \text{const}$$

Example: $RSS(\tilde{\mathbf{w}})$ in compact form

sqft (1000's)	bedrooms	bathrooms	sale price (100k)
1	2	1	2
2	2	2	3.5
1.5	3	2	3
2.5	4	2.5	4.5

Design matrix and target vector:

$$\tilde{\mathbf{X}} = \begin{pmatrix} \tilde{\mathbf{x}}_1^\top \\ \tilde{\mathbf{x}}_2^\top \\ \vdots \\ \tilde{\mathbf{x}}_N^\top \end{pmatrix} = \begin{bmatrix} 1 & 1 & 2 & 1 \\ 1 & 2 & 2 & 2 \\ 1 & 1.5 & 3 & 2 \\ 1 & 2.5 & 4 & 2.5 \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} 2 \\ 3.5 \\ 3 \\ 4.5 \end{bmatrix}$$

. Compact expression:

$$RSS(\tilde{\mathbf{w}}) = \|\tilde{\mathbf{X}}\tilde{\mathbf{w}} - \mathbf{y}\|_2^2 = \left\{ \tilde{\mathbf{w}}^\top \tilde{\mathbf{X}}^\top \tilde{\mathbf{X}} \tilde{\mathbf{w}} - 2 \left(\tilde{\mathbf{X}}^\top \mathbf{y} \right)^\top \tilde{\mathbf{w}} \right\} + \text{const}$$

Solution in matrix form

Compact expression

$$RSS(\tilde{\mathbf{w}}) = ||\tilde{\mathbf{X}}\tilde{\mathbf{w}} - \mathbf{y}||_2^2 = \left\{ \tilde{\mathbf{w}}^\top \tilde{\mathbf{X}}^\top \tilde{\mathbf{X}} \tilde{\mathbf{w}} - 2 \left(\tilde{\mathbf{X}}^\top \mathbf{y} \right)^\top \tilde{\mathbf{w}} \right\} + \text{const}$$

Gradients of Linear and Quadratic Functions

- $\nabla_{\mathbf{x}}(\mathbf{b}^\top \mathbf{x}) = \mathbf{b}$
- $\nabla_{\mathbf{x}}(\mathbf{x}^\top \mathbf{A} \mathbf{x}) = 2\mathbf{A}\mathbf{x}$ (symmetric $\mathbf{A}$)

Normal equation

$$\nabla_{\tilde{\mathbf{w}}} RSS(\tilde{\mathbf{w}}) = 2\tilde{\mathbf{X}}^\top \tilde{\mathbf{X}} \tilde{\mathbf{w}} - 2\tilde{\mathbf{X}}^\top \mathbf{y} = 0$$

This leads to the least-mean-squares (LMS) solution

$$\tilde{\mathbf{w}}^{LMS} = \left(\tilde{\mathbf{X}}^\top \tilde{\mathbf{X}} \right)^{-1} \tilde{\mathbf{X}}^\top \mathbf{y}$$

Example: $RSS(\tilde{\mathbf{w}})$ in compact form

sqft (1000's)	bedrooms	bathrooms	sale price (100k)
1	2	1	2
2	2	2	3.5
1.5	3	2	3
2.5	4	2.5	4.5

Write the least-mean-squares (LMS) solution

$$\tilde{\mathbf{w}}^{LMS} = \left(\tilde{\mathbf{X}}^\top \tilde{\mathbf{X}}\right)^{-1} \tilde{\mathbf{X}}^\top \mathbf{y}$$

Can use solvers in Matlab, Python etc., to compute this for any given $\tilde{\mathbf{X}}$ and $\mathbf{y}$.

Exercise: $RSS(\tilde{\mathbf{w}})$ in compact form

Using the general least-mean-squares (LMS) solution

$$\tilde{\mathbf{w}}^{LMS} = \left(\tilde{\mathbf{X}}^\top \tilde{\mathbf{X}}\right)^{-1} \tilde{\mathbf{X}}^\top \mathbf{y}$$

recover the uni-variate solution that we had computed earlier:

$$w_1 = \frac{\sum (x_n - \bar{x})(y_n - \bar{y})}{\sum (x_i - \bar{x})^2} \qquad \text{and} \qquad w_0 = \bar{y} - w_1 \bar{x}$$

where $\bar{x} = \frac{1}{N}\sum_n x_n$ and $\bar{y} = \frac{1}{N}\sum_n y_n$.

Exercise: $RSS(\tilde{\mathbf{w}})$ in compact form

For the 1-D case, the least-mean-squares solution is

$$\tilde{\mathbf{w}}^{LMS} = \left(\tilde{\mathbf{X}}^\top \tilde{\mathbf{X}}\right)^{-1} \tilde{\mathbf{X}}^\top \mathbf{y}$$

$$= \left(\begin{bmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_N \end{bmatrix} \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & \dots \\ 1 & x_N \end{bmatrix} \right)^{-1} \begin{bmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_N \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_N \end{bmatrix}$$

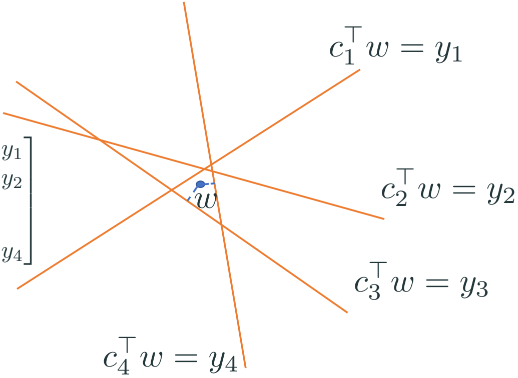
$$= \left(\begin{bmatrix} N & N\bar{x} \\ N\bar{x} & \sum_n x_n^2 \end{bmatrix} \right)^{-1} \begin{bmatrix} \sum_n y_n \\ \sum_n x_n y_n \end{bmatrix}$$

$$\begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = \frac{1}{\sum (x_i - \bar{x})^2)} \begin{bmatrix} \bar{y} \sum (x_i - \bar{x})^2 - \bar{x} \sum (x_n - \bar{x})(y_n - \bar{y}) \\ \sum (x_n - \bar{x})(y_n - \bar{y}) \end{bmatrix}$$

where $\bar{x} = \frac{1}{N} \sum_n x_n$ and $\bar{y} = \frac{1}{N} \sum_n y_n$.

Why is minimizing RSS sensible?

$$r = \begin{bmatrix} c_1^\top w - y_1 \\ c_2^\top w - y_2 \\ \vdots \\ c_4^\top w - y_4 \end{bmatrix} = Aw - y$$



- Want to find $\mathbf{w}$ that minimizes the difference between $\mathbf{Xw}$, $\mathbf{y}$
- But since this a vector, we need an operator that can map the residual vector $r(\mathbf{w}) = \mathbf{y} - \mathbf{Xw}$ to a scalar
- We take the sum of the squares of the elements of $r(\mathbf{w})$

Why is minimizing RSS sensible?

Probabilistic interpretation

- **Noisy observation model**:

$$Y = w_0 + w_1 X + \eta$$

  where $\eta \sim N(0, \sigma^2)$ is a Gaussian random variable

- Conditional likelihood of one training sample:

$$p(y_n|x_n) = N(w_0 + w_1 x_n, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{[y_n - (w_0 + w_1 x_n)]^2}{2\sigma^2}}$$

Probabilistic interpretation (cont'd)

Log-likelihood of the training data $\mathcal{D}$ (assuming i.i.d):

$$
\begin{aligned}
\log P(\mathcal{D}) &= \log \prod_{n=1}^{N} p(y_n|x_n) = \sum_n \log p(y_n|x_n) \\
&= \sum_n \left\{ -\frac{[y_n - (w_0 + w_1 x_n)]^2}{2\sigma^2} - \log \sqrt{2\pi}\sigma \right\} \\
&= -\frac{1}{2\sigma^2} \sum_n [y_n - (w_0 + w_1 x_n)]^2 - \frac{N}{2} \log \sigma^2 - N \log \sqrt{2\pi} \\
&= -\frac{1}{2} \left\{ \frac{1}{\sigma^2} \sum_n [y_n - (w_0 + w_1 x_n)]^2 + N \log \sigma^2 \right\} + \text{const}
\end{aligned}
$$

What is the relationship between minimizing RSS and maximizing the log-likelihood?

Maximum likelihood estimation

Estimating σ, w_0 and w_1 can be done in two steps

- Maximize over w_0 and w_1:

$$\max \ \log P(\mathcal{D}) \Leftrightarrow \min \ \sum_n [y_n - (w_0 + w_1 x_n)]^2 \leftarrow \text{This is RSS}(\tilde{\mathbf{w}})!$$

- Maximize over $s = \sigma^2$:

$$\frac{\partial \log P(\mathcal{D})}{\partial s} = -\frac{1}{2}\left\{-\frac{1}{s^2}\sum_n [y_n - (w_0 + w_1 x_n)]^2 + N\frac{1}{s}\right\} = 0$$

$$\rightarrow {\sigma^*}^2 = s^* = \frac{1}{N}\sum_n [y_n - (w_0 + w_1 x_n)]^2$$

How does this probabilistic interpretation help us?

- It gives a solid footing to our intuition: minimizing $\text{RSS}(\tilde{\mathbf{w}})$ is a sensible thing based on reasonable modeling assumptions.
- Estimating σ^* tells us how much noise there is in our predictions. For example, it allows us to place confidence intervals around our predictions.

Computational complexity of the Least Squares Solution

Bottleneck of computing the solution?

$$\boldsymbol{w} = \left(\boldsymbol{X}^\top \boldsymbol{X}\right)^{-1} \boldsymbol{X}\boldsymbol{y}$$

Matrix multiply of $\mathbf{X}^\top \mathbf{X} \in \mathbb{R}^{(D+1)\times(D+1)}$
Inverting the matrix $\mathbf{X}^\top \mathbf{X}$

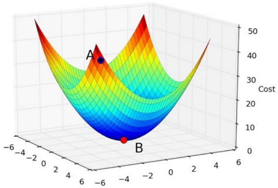
How many operations do we need?

- $O(\mathrm{ND}^2)$ for matrix multiplication
- $O(\mathrm{D}^3)$ (e.g., using Gauss-Jordan elimination) or $O(\mathrm{D}^{2.373})$ (recent theoretical advances) for matrix inversion
- Impractical for very large D or N

Alternative method: Batch Gradient Descent

(Batch) Gradient descent

- Initialize $\mathbf{w}$ to $\boldsymbol{w}^{(0)}$ (e.g., randomly);
  set $t = 0$; choose $\eta > 0$
- Loop *until convergence*
  1. Compute the gradient
     $\nabla RSS(\boldsymbol{w}) = \mathbf{X}^\top(\mathbf{X}\boldsymbol{w}^{(t)} - \boldsymbol{y})$
  2. Update the parameters
     $\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta\nabla RSS(\boldsymbol{w})$
  3. $t \leftarrow t + 1$



What is the complexity of each iteration?
$O(\mathrm{ND})$

Why would this work?

If gradient descent converges, it will converge to the same solution as using matrix inversion.

This is because $RSS(\boldsymbol{w})$ is a convex function in its parameters $\boldsymbol{w}$

Hessian of RSS

$$RSS(\boldsymbol{w}) = \boldsymbol{w}^\top \mathbf{X}^\top \mathbf{X} \boldsymbol{w} - 2 \left(\mathbf{X}^\top \boldsymbol{y}\right)^\top \boldsymbol{w} + \text{const}$$
$$\Rightarrow \frac{\partial^2 RSS(\boldsymbol{w})}{\partial \boldsymbol{w} \boldsymbol{w}^\top} = 2\mathbf{X}^\top \mathbf{X}$$

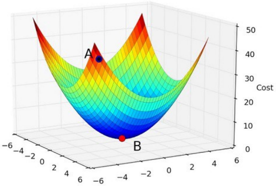
$\mathbf{X}^\top \mathbf{X}$ is positive semidefinite, because for any $\boldsymbol{v}$

$$\boldsymbol{v}^\top \mathbf{X}^\top \mathbf{X} \boldsymbol{v} = \|\mathbf{X}^\top \boldsymbol{v}\|_2^2 \geq 0$$

Alternative method: Batch Gradient Descent

(Batch) Gradient descent

- Initialize $\mathbf{w}$ to $\boldsymbol{w}^{(0)}$ (e.g., randomly);
  set $t = 0$; choose $\eta > 0$
- Loop *until convergence*
  1. Compute the gradient
     $\nabla RSS(\boldsymbol{w}) = \mathbf{X}^{\top}\mathbf{X}\boldsymbol{w}^{(t)} - \mathbf{X}^{\top}\boldsymbol{y}$
  2. Update the parameters
     $\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta\nabla RSS(\boldsymbol{w})$
  3. $t \leftarrow t + 1$



What is the complexity of each iteration?
$O(\text{ND})$

Stochastic gradient descent (SGD)

Widrow-Hoff rule: update parameters using one example at a time

- Initialize $\boldsymbol{w}$ to some $\boldsymbol{w}^{(0)}$; set $t = 0$; choose $\eta > 0$
- Loop *until convergence*
  1. random choose a training a sample $\boldsymbol{x}_t$
  2. Compute its contribution to the gradient

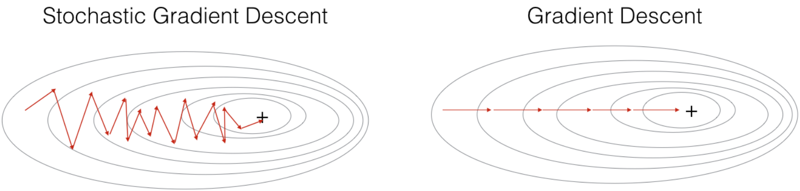
$$\boldsymbol{g}_t = (\boldsymbol{x}_t^\top \boldsymbol{w}^{(t)} - y_t)\mathbf{x}_t$$

  3. Update the parameters
     $\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta \boldsymbol{g}_t$
  4. $t \leftarrow t + 1$

How does the complexity per iteration compare with gradient descent?

- $O(\text{ND})$ for gradient descent versus $O(\text{D})$ for SGD

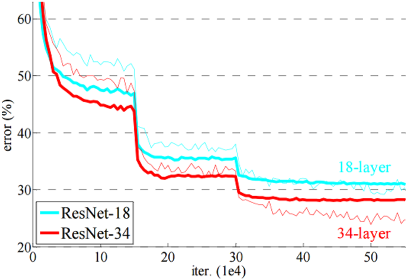
SGD versus Batch GD



- SGD reduces per-iteration complexity from $O(\mathrm{ND})$ to $O(\mathrm{D})$
- But it is noisier and can take longer to converge

How to Choose Learning Rate η in practice?

- Try $0.0001, 0.001, 0.01, 0.1$ etc. on a validation dataset (more on this later) and choose the one that gives fastest, stable convergence
- Reduce η by a constant factor (eg. 10) when learning saturates so that we can reach closer to the true minimum.
- More advanced learning rate schedules such as AdaGrad, Adam, AdaDelta are used in practice.



Mini-Summary

- Linear regression is the linear combination of features $f : \boldsymbol{x} \to y$, with $f(\boldsymbol{x}) = w_0 + \sum_d w_d x_d = w_0 + \boldsymbol{w}^\top \boldsymbol{x}$
- If we minimize residual sum of squares as our learning objective, we get a closed-form solution of parameters
- Probabilistic interpretation: maximum likelihood if assuming residual is Gaussian distributed
- Gradient Descent and mini-batch SGD can overcome computational issues