18-661 Introduction to Machine Learning

Linear Regression – II

Spring 2025

ECE – Carnegie Mellon University

Announcements

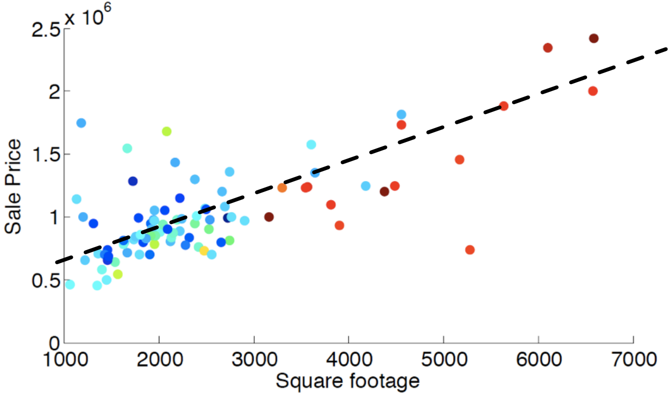
- HW 1 is due on Friday. Note that you can use up to 2 late days per homework, and up to 5 during the semester.
- First mini-exam is on Feb 10th. You are allowed to bring 1 one-sided handwritten US-letter-sized cheat sheet. No electronic devices are permitted. Calculators are allowed but will not be necessary.

Outline

1. Review of Linear Regression
2. Gradient Descent Methods
3. Feature Scaling
4. Ridge Regression
5. Non-linear Basis Functions

Review of Linear Regression

Example: Predicting House Prices



Sale price $\approx$ price_per_sqft $\times$ square_footage + fixed_expense

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 $\mathbf{w} = [w_0 \ w_1 \ w_2 \ \cdots \ w_D]^\top$ parameters.
- **Training data**: $\mathcal{D} = \{(\mathbf{x}_n, y_n), n = 1, 2, \ldots, N\}$

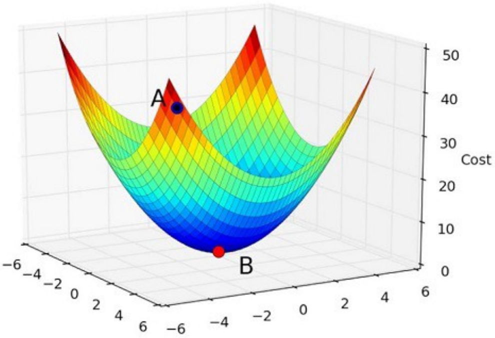
Minimize the residual sum of squares:

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

A Simple Case: x Is One-dimensional (D=1)

Residual sum of squares:

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



CONVEX function (has a unique global minimum w_0^*, w_1^*)

A Simple Case: x Is One-dimensional (D=1)

Residual sum of squares:

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

Stationary points:
Take derivative with respect to parameters and set it to zero

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

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

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 $\bar{y} = \frac{1}{N} \sum_n y_n$.

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

$RSS(\mathbf{w})$ 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)

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

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:

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

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

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

Why Minimize the RSS?

Probabilistic interpretation

- **Noisy observation model** for generating the dataset:

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

  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):

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

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

- Maximize over w_0 and w_1:

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

- This gives a solid footing to our intuition: minimizing RSS($\tilde{\mathbf{w}}$) is a sensible thing based on reasonable modeling assumptions.

Optimizing σ^2

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

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

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

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

- Estimating σ^* tells us how much noise there is in our predictions. For example, it allows us to place confidence intervals around our predictions.

Gradient Descent Methods

Outline

Three Optimization Methods

Want to Minimize

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

- Least-Squares Solution; taking the derivative and setting it to zero
- Batch Gradient Descent
- Stochastic Gradient Descent

For simplicity of notation, we will replace the augmented parameter $\tilde{\mathbf{w}}$ with $\mathbf{w}$ and the augmented design matrix $\tilde{\mathbf{X}}$ with $\mathbf{X}$ from now on

Computational Complexity

Bottleneck of computing the solution?

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

How many operations do we need?

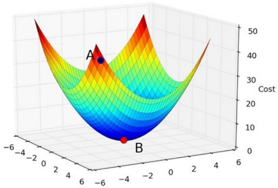
- $O(\mathrm{ND}^2)$ for matrix multiplication $\mathbf{X}^\top\mathbf{X}$
- $O(\mathrm{D}^3)$ (e.g., using Gauss-Jordan elimination) or $O(\mathrm{D}^{2.373})$ (recent theoretical advances) for matrix inversion of $\mathbf{X}^\top\mathbf{X}$
- $O(\mathrm{ND})$ for matrix multiplication $\mathbf{X}^\top\boldsymbol{y}$
- $O(\mathrm{D}^2)$ for $\left(\boldsymbol{X}^\top \boldsymbol{X}\right)^{-1}$ times $\boldsymbol{X}^\top \boldsymbol{y}$

$O(\mathrm{ND}^2) + O(\mathrm{D}^3)$ – 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? $O(\text{ND})$

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 $\boldsymbol{v}$

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

Three Optimization Methods

Want to Minimize

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

- Least-Squares Solution; taking the derivative and setting it to zero
- Batch Gradient Descent
- Stochastic Gradient Descent

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. Randomly choose a training sample $\boldsymbol{x}_t$
  2. Compute its contribution to the gradient

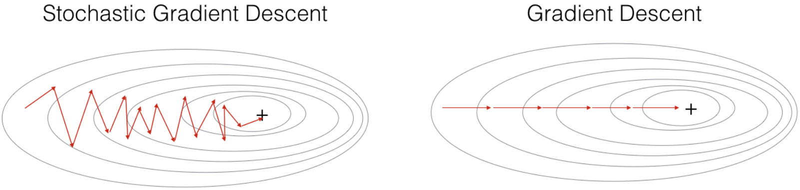
$$\boldsymbol{g}_t = (\boldsymbol{x}_t^\top \boldsymbol{w}^{(t)} - y_t)\mathbf{x}_t$$

  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?

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

Example: Least Squares Solution

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

The w_0 and w_1 that minimize this are given by:

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

$$\begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 1.5 & 2.5 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 1.5 \\ 1 & 2.5 \end{bmatrix} \right)^{-1} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 1.5 & 2.5 \end{bmatrix} \begin{bmatrix} 2 \\ 3.5 \\ 3 \\ 4.5 \end{bmatrix}$$

$$\begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = \begin{bmatrix} 0.45 \\ 1.6 \end{bmatrix}$$

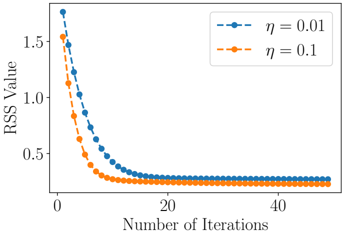
Minimum RSS is $RSS^* = ||\mathbf{X}\mathbf{w}^{LMS} - \mathbf{y}||_2^2 = 0.2236$

Example: Batch Gradient Descent

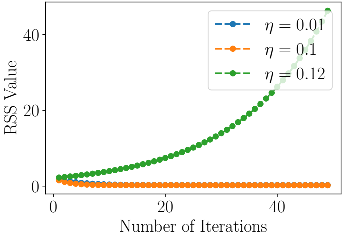
sqft (1000's)	sale price (100k)
1	2
2	3.5
1.5	3
2.5	4.5

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta \nabla RSS(\boldsymbol{w}) = \boldsymbol{w}^{(t)} - \eta \mathbf{X}^\top \left(\mathbf{X}\boldsymbol{w}^{(t)} - \boldsymbol{y} \right)$$

Larger η gives faster convergence



But too large η makes GD unstable

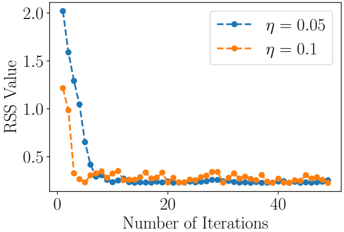


Example: Stochastic Gradient Descent

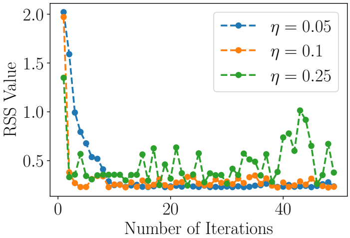
sqft (1000's)	sale price (100k)
1	2
2	3.5
1.5	3
2.5	4.5

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta \nabla RSS(\boldsymbol{w}) = \boldsymbol{w}^{(t)} - \eta \left(\mathbf{x}_t^\top \boldsymbol{w}^{(t)} - \boldsymbol{y} \right) \mathbf{x}_t$$

Larger η gives faster convergence

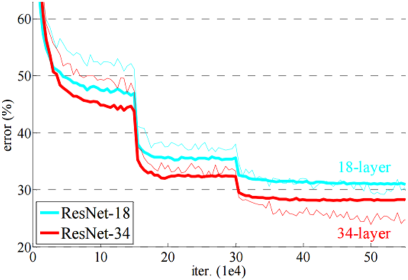


But too large η makes SGD unstable



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.



Gradient Descent Methods in Machine Learning

- Batch gradient descent computes the exact gradient.
- Stochastic gradient descent approximates the gradient with a single data point; its expectation equals the true gradient.
- Mini-batch variant: set the batch size to trade-off between accuracy of estimating gradient and computational cost
- Similar ideas extend to other ML optimization problems.

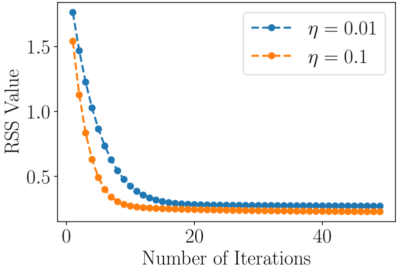
Feature Scaling

Outline

Batch Gradient Descent: Scaled Features

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

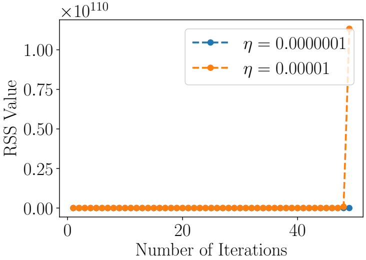
$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta \nabla RSS(\boldsymbol{w}) = \boldsymbol{w}^{(t)} - \eta \mathbf{X}^\top \left(\mathbf{X} \boldsymbol{w}^{(t)} - \boldsymbol{y} \right)$$



Batch Gradient Descent: Without Feature Scaling

sqft	sale price
1000	200,000
2000	350,000
1500	300,000
2500	450,000

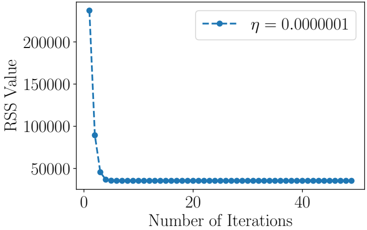
- Least-squares solution is $(w_0^*, w_1^*) = (45000, 160)$
- $\nabla RSS(\boldsymbol{w}^{(t)}) = \mathbf{X}^\top \left(\mathbf{X}\boldsymbol{w}^{(t)} - \boldsymbol{y}\right)$ becomes HUGE, causing instability
- We need a tiny η to compensate, but this can cause numerical issues



Batch Gradient Descent: Without Feature Scaling

sqft	sale price
1000	200,000
2000	350,000
1500	300,000
2500	450,000

- Least-squares solution is $(w_0^*, w_1^*) = (45000, 160)$
- $\nabla RSS(\boldsymbol{w})$ becomes HUGE, causing instability
- We need a tiny η to compensate, but this leads to slow convergence



How to Scale Features?

- **Min-max normalization**

$$x'_d = \frac{x_d - \min_n(x_d)}{\max_n x_d - \min_n x_d}$$

The min and max are taken over the possible values $x_d^{(1)}, \ldots x_d^{(N)}$ of x_d in the dataset. This will result in all scaled features $0 \leq x_d \leq 1$

- **Mean normalization**

$$x'_d = \frac{x_d - \text{avg}(x_d)}{\max_n x_d - \min_n x_d}$$

This will result in all scaled features $-1 \leq x_d \leq 1$.

Labels $y^{(1)}, \ldots y^{(N)}$ should be similarly re-scaled

Several other methods: e.g., dividing by standard deviation (Z-score normalization)

Ridge Regression

Outline

What if $X^\top X$ Is Not Invertible?

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

Why might this happen?

- **Answer 1:** $\text{N} < \text{D}$. Not enough data to estimate all parameters. $\mathbf{X}^\top \mathbf{X}$ is not full-rank
- **Answer 2:** Columns of $\boldsymbol{X}$ are not linearly independent, e.g., some features are linear functions of other features. In this case, solution is not unique. Examples:
    - A feature is a re-scaled version of another, for example, having two features correspond to length in meters and feet respectively
    - Same feature is repeated twice (e.g., when there are many features)
    - A feature has the same value for all data points
    - A feature is a linear combination of others, such as the sum of two features being equal to a third feature

Example: Matrix $X^\top X$ Is Not Invertible

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

Design matrix and target vector:

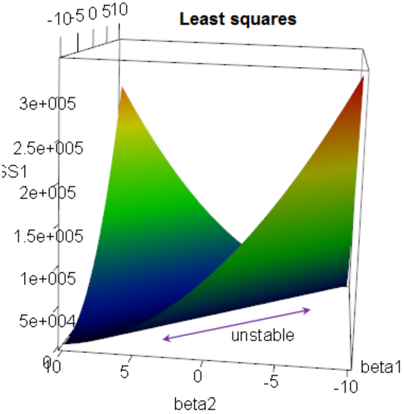
$$\mathbf{X} = \begin{bmatrix} 1 & 1 & 2 \\ 1 & 2 & 2 \\ 1 & 1.5 & 2 \\ 1 & 2.5 & 2 \end{bmatrix}, \quad \mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} 2 \\ 3.5 \\ 3 \\ 4.5 \end{bmatrix}$$

The 'bathrooms' feature is redundant, so we don't need w_2

$$\begin{aligned} y &= w_0 + w_1 x_1 + w_2 x_2 \\ &= w_0 + w_1 x_1 + w_2 \times 2, \quad \text{since } x_2 \text{ is always 2!} \\ &= w_{0,eff} + w_1 x_1, \quad \text{where } w_{0,eff} = (w_0 + 2w_2) \end{aligned}$$

What Does the RSS Look Like?

- When $\boldsymbol{X}^\top \boldsymbol{X}$ is not invertible, the RSS objective function has a ridge, that is, the minimum is a line instead of a single point



In our example, this line is $w_{0,eff} = (w_0 + 2w_2)$

How Do You Fix This Issue?

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

- Manually remove redundant features
- But this can be tedious and non-trivial, especially when a feature is a linear combination of several other features

Need a general way that doesn't require manual feature engineering

SOLUTION: Ridge Regression

Ridge Regression

Intuition: what does a non-invertible $\boldsymbol{X}^\top \boldsymbol{X}$ mean?

Consider the EVD (why does this exist?) of this matrix:

$$\boldsymbol{X}^\top \boldsymbol{X} = \boldsymbol{V} \begin{bmatrix} \lambda_1 & 0 & 0 & \cdots & 0 \\ 0 & \lambda_2 & 0 & \cdots & 0 \\ 0 & \cdots & \cdots & \cdots & 0 \\ 0 & \cdots & \cdots & \lambda_r & 0 \\ 0 & \cdots & \cdots & 0 & 0 \end{bmatrix} \boldsymbol{V}^\top$$

where $\lambda_1 \geq \lambda_2 \geq \cdots \lambda_r > 0$ and $r <$ D. We will have a divide by zero issue when computing $(\boldsymbol{X}^\top \boldsymbol{X})^{-1}$...

Fix the problem: ensure all singular values are non-zero:

$$\boldsymbol{X}^\top \boldsymbol{X} + \lambda \boldsymbol{I} = \boldsymbol{V} \text{diag}(\lambda_1 + \lambda, \lambda_2 + \lambda, \cdots, \lambda) \boldsymbol{V}^\top$$

where $\lambda > 0$ and $\boldsymbol{I}$ is the identity matrix.

Regularized Least Squares (Ridge Regression)

Solution

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

This is equivalent to adding an extra term to $RSS(\boldsymbol{w})$

$$\overbrace{\frac{1}{2}\left\{\boldsymbol{w}^\top \boldsymbol{X}^\top \boldsymbol{X} \boldsymbol{w} - 2\left(\boldsymbol{X}^\top \boldsymbol{y}\right)^\top \boldsymbol{w} + \text{const.}\right\}}^{RSS(\boldsymbol{w})} + \underbrace{\frac{1}{2}\lambda \|w\|_2^2}_{\text{regularization}}$$

$$\frac{1}{2}\left\{\boldsymbol{w}^\top \left(\boldsymbol{X}^\top \boldsymbol{X} + \lambda \boldsymbol{I}\right) \boldsymbol{w} - 2\left(\boldsymbol{X}^\top \boldsymbol{y}\right)^\top \boldsymbol{w} + \text{const.}\right\}$$

Benefits

- Numerically more stable, invertible matrix
- Force **w** to be small
- Prevent overfitting — more on this in the next lecture

Ridge Regression on Our Example

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

The 'bathrooms' feature is redundant, so we don't need w_2

$$
\begin{aligned}
y &= w_0 + w_1 x_1 + w_2 x_2 \\
&= w_0 + w_1 x_1 + w_2 \times 2, && \text{since } x_2 \text{ is always 2!} \\
&= w_{0,eff} + w_1 x_1, && \text{where } w_{0,eff} = (w_0 + 2w_2) \\
&= 0.45 + 1.6 x_1 && \text{Should get this}
\end{aligned}
$$

Ridge Regression on Our Example

The 'bathrooms' feature is redundant, so we don't need w_2

$$\begin{aligned} y &= w_0 + w_1 x_1 + w_2 x_2 \\ &= w_0 + w_1 x_1 + w_2 \times 2, \quad \text{since } x_2 \text{ is always 2!} \\ &= w_{0,eff} + w_1 x_1, \quad \text{where } w_{0,eff} = (w_0 + 2w_2) \\ &= 0.45 + 1.6 x_1 \quad \text{Should get this} \end{aligned}$$

Compute the solution for $\lambda = 0.5$

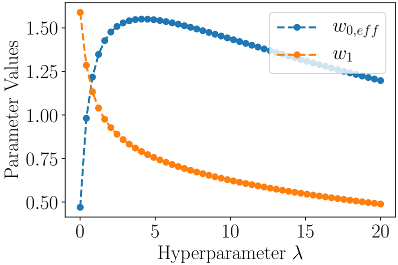
$$\begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix} = \left(\boldsymbol{X}^\top \boldsymbol{X} + \lambda \boldsymbol{I}\right)^{-1} \boldsymbol{X}^\top \boldsymbol{y}$$

$$\begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} 0.208 \\ 1.247 \\ 0.4166 \end{bmatrix} \quad \text{recall } \begin{bmatrix} w_{0,eff} \\ w_1 \end{bmatrix} = \begin{bmatrix} 0.45 \\ 1.6 \end{bmatrix} \text{ for LMS}$$

How Does λ Affect the Solution?

$$\begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix} = \left(\boldsymbol{X}^\top \boldsymbol{X} + \lambda \boldsymbol{I}\right)^{-1} \boldsymbol{X}^\top \boldsymbol{y}$$

Let us plot $w_{0,eff} = w_0 + 2w_2$ and w_1 for different $\lambda \in [0.01, 20]$

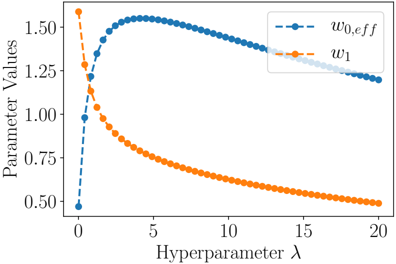


Setting small λ gives almost the least-squares solution, but it can cause numerical instability in the inversion

How to Choose λ?

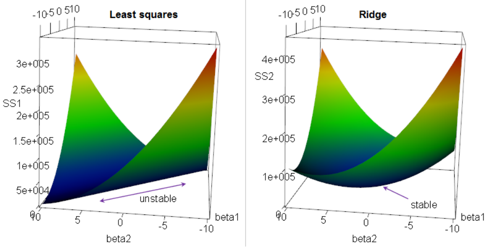
λ is referred to as a *hyperparameter*

- Associated with the estimation method, not the dataset
- In contrast, $\boldsymbol{w}$ is the parameter vector
- Use validation set or cross-validation to find good choice of λ (more on this in the next lecture)



Why Is It Called Ridge Regression?

- When $\boldsymbol{X}^\top \boldsymbol{X}$ is not invertible, the RSS objective function has a ridge, that is, the minimum is a line instead of a single point
- Adding the regularizer term $\frac{1}{2}\lambda\|w\|_2^2$ yields a unique minimum, thus avoiding instability in matrix inversion



Probabilistic Interpretation of Ridge Regression

Add a term to the objective function.

- Choose the parameters to not just minimize risk (i.e., minimize the RSS), but also avoid being too large.

$$\frac{1}{2}\left\{\boldsymbol{w}^\top \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{w} - 2\left(\boldsymbol{X}^\top \boldsymbol{y}\right)^\top \boldsymbol{w}\right\} + \frac{1}{2}\lambda\|w\|_2^2$$

Probabilistic interpretation: Place a prior on our weights

- Interpret $\boldsymbol{w}$ as a random variable
- Assume that each w_d is centered around zero
- Use observed data $\mathcal{D}$ to update our prior belief on $\boldsymbol{w}$

Gaussian priors lead to ridge regression.

Review: Probabilistic Interpretation of Linear Regression

Linear Regression model: $Y = \boldsymbol{w}^\top \boldsymbol{X} + \eta$
$\eta \sim N(0, \sigma_0^2)$ is a Gaussian random variable and $Y \sim N(\boldsymbol{w}^\top \boldsymbol{X}, \sigma_0^2)$

Frequentist interpretation: We assume that $\boldsymbol{w}$ is fixed.

- The likelihood function maps parameters to probabilities

$$L : \boldsymbol{w}, \sigma_0^2 \mapsto p(\mathcal{D}|\boldsymbol{w}, \sigma_0^2) = p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}, \sigma_0^2) = \prod_n p(y_n|\boldsymbol{x}_n, \boldsymbol{w}, \sigma_0^2)$$

- Maximizing the likelihood with respect to $\boldsymbol{w}$ minimizes the RSS and yields the LMS solution:

$$\boldsymbol{w}^{\mathrm{LMS}} = \boldsymbol{w}^{\mathrm{ML}} = \arg\max_{\boldsymbol{w}} L(\boldsymbol{w}, \sigma_0^2)$$

Probabilistic Interpretation of Ridge Regression

Ridge Regression model: $Y = \boldsymbol{w}^\top \boldsymbol{X} + \eta$

- $Y \sim N(\boldsymbol{w}^\top \boldsymbol{X}, \sigma_0^2)$ is a Gaussian random variable (as before)
- $w_d \sim N(0, \sigma^2)$ are i.i.d. Gaussian random variables (unlike before)
- Note that all w_d share the same variance σ^2

- To find $\boldsymbol{w}$ given data $\mathcal{D}$, compute the posterior distribution of $\boldsymbol{w}$:

$$p(\boldsymbol{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\boldsymbol{w})p(\boldsymbol{w})}{p(\mathcal{D})}$$

- Maximum a posterior (MAP) estimate:

$$\boldsymbol{w}^{\mathrm{MAP}} = \arg\max_{\boldsymbol{w}} p(\boldsymbol{w}|\mathcal{D}) = \arg\max_{\boldsymbol{w}} p(\mathcal{D}|\boldsymbol{w})p(\boldsymbol{w})$$

Estimating w

Let $\mathbf{x}_1, \ldots, \mathbf{x}_N$ be i.i.d. with $y|\boldsymbol{w}, \boldsymbol{x} \sim N(\boldsymbol{w}^\top \boldsymbol{x}, \sigma_0^2)$; $w_d \sim N(0, \sigma^2)$.

Joint likelihood of data and parameters (given σ_0, σ):

$$p(\mathcal{D}, \boldsymbol{w}) = p(\mathcal{D}|\boldsymbol{w})p(\boldsymbol{w}) \quad = \prod_n p(y_n|\boldsymbol{x}_n, \boldsymbol{w}) \prod_d p(w_d)$$

Plugging in the Gaussian PDF, we get:

$$\begin{aligned}\log p(\mathcal{D}, \boldsymbol{w}) &= \sum_n \log p(y_n|\boldsymbol{x}_n, \boldsymbol{w}) + \sum_d \log p(w_d) \\ &= -\frac{\sum_n (\boldsymbol{w}^\top \boldsymbol{x}_n - y_n)^2}{2\sigma_0^2} - \sum_d \frac{1}{2\sigma^2} w_d^2 + \text{const}\end{aligned}$$

MAP estimate: $\boldsymbol{w}^{\mathrm{MAP}} = \arg\max_{\boldsymbol{w}} \log p(\mathcal{D}, \boldsymbol{w})$

$$\boldsymbol{w}^{\mathrm{MAP}} = \operatorname{argmin}_{\boldsymbol{w}} \left\{ \frac{\sum_n (\boldsymbol{w}^\top \boldsymbol{x}_n - y_n)^2}{2\sigma_0^2} + \frac{1}{2\sigma^2} \|\boldsymbol{w}\|_2^2 \right\}$$

Maximum a Posteriori (MAP) Estimate

MAP Estimate:

$$\boldsymbol{w}^{\mathrm{MAP}} = \operatorname{argmin}_{\boldsymbol{w}} \left\{ \frac{\sum_n (\boldsymbol{w}^\top \boldsymbol{x}_n - y_n)^2}{2\sigma_0^2} + \frac{1}{2\sigma^2} \|\boldsymbol{w}\|_2^2 \right\}$$

After multiplying by $2\sigma_0^2$:

$$\boldsymbol{w}^{\mathrm{MAP}} = \operatorname{argmin}_{\boldsymbol{w}} \left\{ \underbrace{\sum_n (\boldsymbol{w}^\top \boldsymbol{x}_n - y_n)^2}_{RSS} + \frac{\sigma_0^2}{\sigma^2} \underbrace{\|\boldsymbol{w}\|_2^2}_{regularizer} \right\}$$

which is the same as our ridge regression formulation if we define $\lambda = \sigma_0^2/\sigma^2 > 0$. This extra term $\|\boldsymbol{w}\|_2^2$ is called regularization/regularizer and controls the magnitude of $\boldsymbol{w}$.

What Does the MAP Estimate Tell Us?

$$\mathcal{E}(\boldsymbol{w}) = \sum_n (\boldsymbol{w}^\top \boldsymbol{x}_n - y_n)^2 + \lambda \|\boldsymbol{w}\|_2^2$$

where $\lambda > 0$ is used to denote σ_0^2 / σ^2.

Intuitions

- If $\lambda \to +\infty$, then $\sigma_0^2 \gg \sigma^2$: the variance of noise is far greater than what our prior model can allow for $\boldsymbol{w}$. In this case, our prior model on $\boldsymbol{w}$ will force $\boldsymbol{w}$ to be close to zero. Numerically,

$$\boldsymbol{w}^{\mathrm{MAP}} \to \mathbf{0}$$

- If $\lambda \to 0$, then we trust our data more. Numerically,

$$\boldsymbol{w}^{\mathrm{MAP}} \to \boldsymbol{w}^{\mathrm{LMS}} = \operatorname{argmin} \sum_n (\boldsymbol{w}^\top \boldsymbol{x}_n - y_n)^2$$

Outline

1. Review of Linear Regression
2. Gradient Descent Methods
3. Feature Scaling
4. Ridge Regression
5. Non-linear Basis Functions

Non-linear Basis Functions

Outline

Should We Always Use a Linear Model?

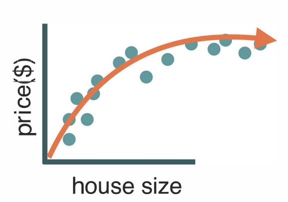


Figure 1: Sale price can saturate as square footage increases

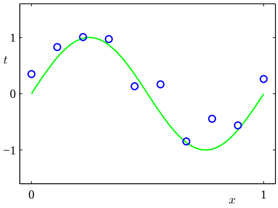


Figure 2: Temperature has cyclic variations over each year

General Nonlinear Basis Functions

We can use a nonlinear mapping to a new feature vector:

$$\phi(\boldsymbol{x}) : \boldsymbol{x} \in \mathbb{R}^D \to \boldsymbol{z} \in \mathbb{R}^M$$

- M is dimensionality of new features $\boldsymbol{z}$ (or $\phi(\boldsymbol{x})$)
- M could be greater than, less than, or equal to D

We can apply existing learning methods on the transformed data:

- linear methods: prediction is based on $\boldsymbol{w}^\top \phi(\boldsymbol{x})$
- other methods: nearest neighbors, decision trees, etc

Regression with Nonlinear Basis

Residual sum of squares

$$\sum_n [\boldsymbol{w}^\top \phi(\boldsymbol{x}_n) - y_n]^2$$

where $\boldsymbol{w} \in \mathbb{R}^M$, the same dimensionality as the transformed features $\phi(\boldsymbol{x})$.

The LMS solution can be formulated with the new design matrix

$$\boldsymbol{\Phi} = \begin{pmatrix} \phi(\boldsymbol{x}_1)^\top \\ \phi(\boldsymbol{x}_2)^\top \\ \vdots \\ \phi(\boldsymbol{x}_N)^\top \end{pmatrix} \in \mathbb{R}^{N \times M}, \quad \boldsymbol{w}^{\mathrm{LMS}} = \left(\boldsymbol{\Phi}^\top \boldsymbol{\Phi}\right)^{-1} \boldsymbol{\Phi}^\top \boldsymbol{y}$$

Example: Flexibility in Designing New Features!

x_1, Area (1k sqft)	x_1^2, Area^2	Price (100k)
1	1	2
2	4	3.5
1.5	2.25	3
2.5	6.25	4.5

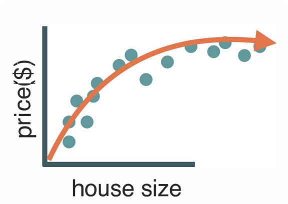


Figure 3: Add x_1^2 as a feature to allow us to fit quadratic, instead of linear functions of the house area x_1

Example: Flexibility in Designing New Features!

x_1, front (100ft)	x_2 depth (100ft)	$10x_1x_2$, Lot (1k sqft)	Price (100k)
0.5	0.5	2.5	2
0.5	1	5	3.5
0.8	1.5	12	3
1.0	1.5	15	4.5

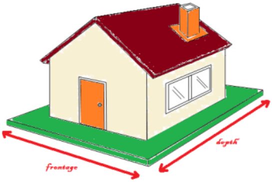


Figure 4: Instead of having frontage and depth as two separate features, it may be better to consider the lot-area, which is equal to frontage×depth

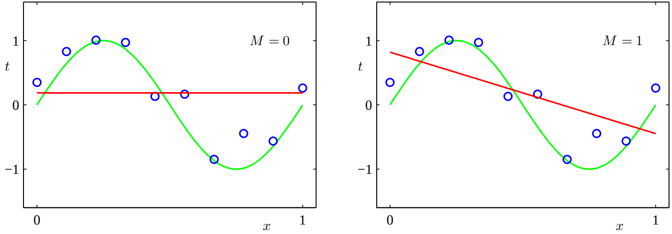
Example with Regression

Polynomial basis functions

$$\phi(x) = \begin{bmatrix} 1 \\ x \\ x^2 \\ \vdots \\ x^M \end{bmatrix} \Rightarrow f(x) = w_0 + \sum_{m=1}^{M} w_m x^m$$

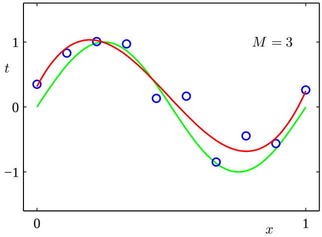
Fitting samples from a sine function:

underfitting since $f(x)$ is too simple

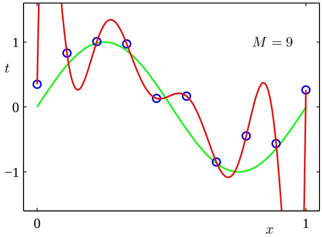


Adding Higher-order Terms

M=3



M=9: overfitting



More complex features lead to better results on the training data, but potentially worse results on new data, e.g., test data!

You Should Know

- Advantages and disadvantages of the least-mean-squares, batch gradient descent, and stochastic gradient descent solution methods
- Examples of feature scaling and why it can be important
- Formulation and solution of ridge regression
- Probabilistic interpretation of ridge regression
- How to use nonlinear basis functions in linear regression