Basic Linear Algebra Review

Matrices

An $m \times n$ **matrix** is a table of numbers with m rows, n columns:

- Example: 2×3 matrix:

$$\begin{pmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \end{pmatrix}$$

- Example: 3×3 "square" matrix:

$$\begin{pmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{pmatrix}$$

Matrix Notation

- We use upper-case letters for matrices.

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

- Sometimes use subscripts to denote particular elements: $A_{13} = 3, A_{21} = 4$
- A^T denotes the transpose of A:

$$A^T = \begin{pmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{pmatrix}$$

Matrix Addition and Scalar Multiplication

- We can add two matrices only if they are the same size.

- Addition occurs elementwise:

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} + \begin{pmatrix} 7 & 8 & 9 \\ -1 & -2 & -3 \end{pmatrix} = \begin{pmatrix} 8 & 10 & 12 \\ 3 & 3 & 3 \end{pmatrix}$$

- Scalar multiplication occurs elementwise, too:

$$2 \cdot \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} = \begin{pmatrix} 2 & 4 & 6 \\ 8 & 10 & 12 \end{pmatrix}$$

Matrix-Matrix Multiplication

- We can multiply two matrices A and B only if # cols in A is equal to # rows in B
- If $A = m \times n$ and $B = n \times p$, the result is $m \times p$.
  - This is **very useful**. Remember it!
- The low-level definition. the ij entry of the product is:

$$(AB)_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

Matrix-Matrix Multiplication Example

$$A = \begin{pmatrix} 1 & 2 & 1 \\ 3 & 4 & 5 \end{pmatrix} \qquad B = \begin{pmatrix} 3 & 6 \\ 1 & 3 \\ 4 & 8 \end{pmatrix}$$

- What is the size of AB?
- What is $(AB)_{12}$?

Matrix-Matrix Multiplication Properties

- Distributive: $A(B + C) = AB + AC$
- Associative: $(AB)C = A(BC)$
- **Not commutative in general**: $AB \neq BA$

Identity Matrices

- The $n \times n$ **identity matrix** I has ones along the diagonal:

$$\begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$$

- If A is $n \times m$, then $IA = A$.

- If B is $m \times n$, then $BI = B$.

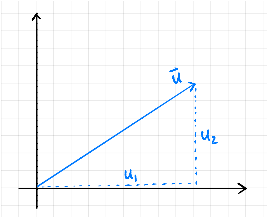
Vectors

- An d-**vector** is an $d \times 1$ matrix.
- Often use arrow, lower-case letters to denote: $\vec{x}$.
- Often write $\vec{x} \in \mathbb{R}^d$ to say $\vec{x}$ is a d vector.
- Example. A 4-vector:

$$\begin{pmatrix} 2 \\ 1 \\ 5 \\ -3 \end{pmatrix}$$

Geometric Meaning of Vectors

- A vector $\vec{u} = (u_1, \dots, u_d)^T$ is an arrow to the point $(u_1, \dots, u_d)$:



- The length, or **norm**, of $\vec{u}$ is
$$\|\vec{u}\| = \sqrt{u_1^2 + u_2^2 + \dots + u_d^2}.$$
- A **unit vector** is a vector of norm 1.

Dot Products

- The **dot product** of two d-vectors $\vec{u}$ and $\vec{v}$ is:

$$\vec{u} \cdot \vec{v} = \vec{u}^T \vec{v}$$

- Using low-level matrix multiplication definition:

$$\vec{u} \cdot \vec{v} = \sum_{i=1}^{n} u_i v_i$$

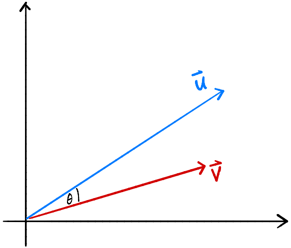
$$= u_1 v_1 + u_2 v_2 + \ldots + u_n v_n$$

Dot Product Example

$$\vec{u} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \qquad \vec{v} = \begin{pmatrix} 4 \\ 5 \\ 6 \end{pmatrix} \qquad \vec{u} \cdot \vec{v} =$$

Geometric Interpretation of Dot Product

- $\vec{u} \cdot \vec{v} = \|\vec{u}\|\,\|\vec{v}\| \cos\theta.$



Which of these is another expression for the norm of $\vec{u}$?

a) $\vec{u} \cdot \vec{u}$
b) $\sqrt{\vec{u}^2}$
c) $\sqrt{\vec{u} \cdot \vec{u}}$
d) $\vec{u}^2$

Properties of the Dot Product

- Commutative: $\vec{u} \cdot \vec{v} = \vec{v} \cdot \vec{u}$
- Distributive: $\vec{u} \cdot (\vec{v} + \vec{w}) = \vec{u} \cdot \vec{v} + \vec{u} \cdot \vec{w}$
- Linear: $\vec{u} \cdot (\alpha\vec{v} + \beta\vec{w}) = \alpha\vec{u} \cdot v + \beta\vec{u} \cdot \vec{w}$

Matrix-Vector Multiplication

- Special case of matrix-matrix multiplication.
- Result is always a vector with same number of rows as the matrix.
- One view: a “mixture” of the columns.

$$\begin{pmatrix} 1 & 2 & 1 \\ 3 & 4 & 5 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} = a_1 \begin{pmatrix} 1 \\ 3 \end{pmatrix} + a_2 \begin{pmatrix} 2 \\ 4 \end{pmatrix} + a_3 \begin{pmatrix} 1 \\ 5 \end{pmatrix}$$

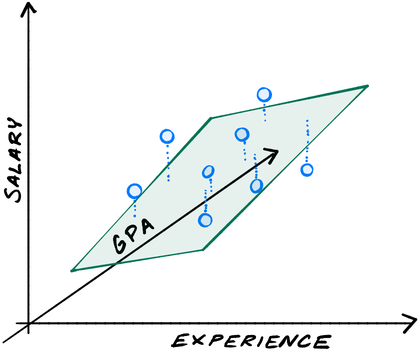
Matrices and Functions

- Matrix-vector multiplication takes in a vector, outputs a vector.
- An $m \times n$ matrix is an encoding of a function mapping $\mathbb{R}^m$ to $\mathbb{R}^n$.
- Matrix multiplication evaluates that function.

Today

- How do we predict salary given **multiple** features?
  - years of experience, number of internships, GPA, etc.

- We'll need to use some linear algebra...



CSE 151A

Intro to Machine Learning

Lecture 07 – Part 01

Setup

Today

- How do we predict salary given **multiple** features?
  - years of experience, number of internships, GPA, etc.

Using Multiple Features

- We believe salary is a function of experience *and* GPA.

- I.e., there is a function H so that:

$$\text{salary} \approx H(\text{years of experience}, \text{GPA})$$

- Recall: H is a **prediction rule**.

- **Our goal**: find a good prediction rule, H.

Example Prediction Rules

$$H_1(\text{experience}, \text{GPA}) = \$40{,}000 \times \frac{\text{GPA}}{4.0} + \$2,000 \times (\text{experience})$$

$$H_2(\text{experience}, \text{GPA}) = \$60{,}000 \times 1.05^{(\text{experience}+\text{GPA})}$$

$$H_3(\text{experience}, \text{GPA}) = \sin(\text{GPA}) + \cos(\text{experience})$$

Linear Prediction Rule

- We'll restrict ourselves to **linear** prediction rules:

$$H(\text{experience}, \text{GPA}) = w_0 + w_1 \times (\text{experience}) + w_2 \times (\text{GPA})$$

- Can add more features, too[1]:

$$H(\text{experience}, \text{GPA}, \#\ \text{internships}) = w_0 + w_1 \times (\text{experience}) + w_2 \times (\text{GPA}) + w_3 \times (\#\ \text{of internships})$$

- Interpretation of w_i: the **weight** of feature x_i.

[1] In practice, might use tens, hundreds, even thousands of features.

Feature Vectors

- In general, if $x_1, \dots, x_d$ are d features:

$$H(x_1, \dots, x_d) = w_0 + w_1 x_1 + w_2 x_2 + \dots + w_d x_d$$

- Nicer to pack into a **feature vector** and **parameter vector**:

$$\vec{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{pmatrix} \qquad \vec{w} = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \\ w_d \end{pmatrix}$$

Augmented Feature Vectors

- The **augmented feature vector** $\text{Aug}(\vec{x})$ is the vector obtained by adding a 1 to the front of $\vec{x}$:

$$\vec{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{pmatrix} \qquad \text{Aug}(\vec{x}) = \begin{pmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_d \end{pmatrix} \qquad \vec{w} = \begin{pmatrix} w_0 \\ w_2 \\ \vdots \\ w_d \\ w_d \end{pmatrix}$$

- Then:

$$\begin{aligned} H(x_1, \ldots, x_d) &= w_0 + w_1 x_1 + w_2 x_2 + \ldots + w_d x_d \\ &= \text{Aug}(\vec{x}) \cdot \vec{w} \end{aligned}$$

Example

- Recall the prediction rule:

$$H_1(\text{experience}, \text{GPA}) = \$40{,}000 \times \frac{\text{GPA}}{4.0} + \$2,000 \times (\text{experience})$$

- This is linear. If x_1 is experience, x_2 is GPA, then:

$$\vec{w} = \begin{pmatrix} w_0 \\ w_1 \\ w_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 2{,}000 \\ 10{,}000 \end{pmatrix}$$

- Prediction for 2 years experience, 3.0 GPA:

$$\text{Aug}(\vec{x}) = \begin{pmatrix} \\ \end{pmatrix} \qquad H(\vec{x}) = \text{Aug}(\vec{x}) \cdot \vec{w} =$$

The Data

- For each person, collect 3 features, plus salary:

Person #	Experience	GPA	# Internships	Salary
1	3	3.7	1	85,000
2	6	3.3	2	95,000
3	10	3.1	3	105,000

- We represent each person with a **data vector**:

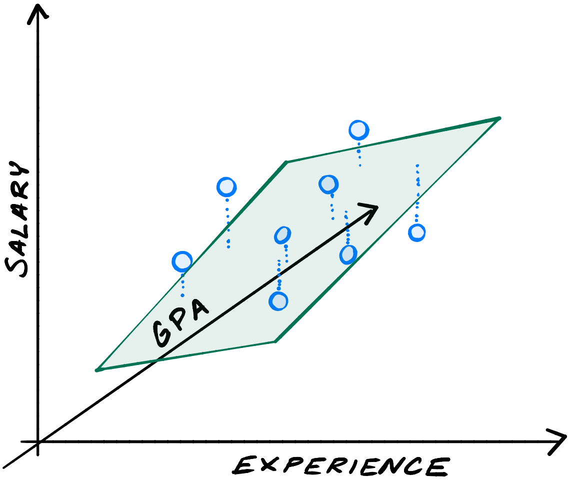
$$\vec{x}^{(1)} = \begin{pmatrix} 3 \\ 3.7 \\ 1 \end{pmatrix}, \qquad \vec{x}^{(2)} = \begin{pmatrix} 6 \\ 3.3 \\ 2 \end{pmatrix}, \qquad \vec{x}^{(3)} = \begin{pmatrix} 10 \\ 3.1 \\ 3 \end{pmatrix}$$

Notation

- $\vec{x}^{(i)}$ is the ith data vector.
- $x_j^{(i)}$ is the jth feature in the ith data vector.
- If there are d features:

$$\vec{x}^{(i)} = \begin{pmatrix} x_1^{(i)} \\ x_2^{(i)} \\ \vdots \\ x_d^{(i)} \end{pmatrix}$$

Geometric Interpretation



The General Problem

- Have n **training examples**: $\left(\vec{x}^{(1)}, y_1\right), \ldots, \left(\vec{x}^{(n)}, y_n\right)$
- We want to find a good linear prediction rule:

$$H(\vec{x}) = \vec{w} \cdot \text{Aug}(\vec{x})$$

- To do so, we'll minimize the mean squared error:

$$\begin{aligned} R_{\text{sq}}(\vec{w}) &= \frac{1}{n} \sum_{i=1}^{n} \left(H(\vec{x}^{(i)}) - y_i\right)^2 \\ &= \frac{1}{n} \sum_{i=1}^{n} \left(\left(\vec{w} \cdot \text{Aug}(\vec{x}^{(i)})\right) - y_i\right)^2 \end{aligned}$$

The Risk

- With d features, we have $d + 1$ parameters: $w_0, w_1, \ldots, w_d$.
- The risk $R_{\text{sq}}(\vec{w})$ is a function from $\mathbb{R}^{d+1}$ to $\mathbb{R}^1$.
- It is a $(d + 1)$-dimensional hypersurface.
- **No hope of visualizing it directly when $d \geq 2$.**

Rewriting the Mean Squared Error

- Let $\vec{e}$ be such that e_i is the (signed) error on ith example:

$$e_i = \left(\vec{w} \cdot \text{Aug}(\vec{x}^{(i)})\right) - y_i$$

- Then:

$$\begin{aligned} R_{\text{sq}}(\vec{w}) &= \frac{1}{n} \sum_{i=1}^{n} \left[\left(\vec{w} \cdot \text{Aug}(\vec{x}^{(i)})\right) - y_i\right]^2 \\ &= \frac{1}{n} \sum_{i=1}^{n} e_i^2 \end{aligned}$$

Rewriting the Mean Squared Error

- Let $\vec{e}$ be such that e_i is the (signed) error on ith example:

$$e_i = \left(\vec{w} \cdot \text{Aug}(\vec{x}^{(i)})\right) - y_i$$

- Then:

$$R_{\text{sq}}(\vec{w}) = \frac{1}{n} \sum_{i=1}^{n} \left[\left(\vec{w} \cdot \text{Aug}(\vec{x}^{(i)})\right) - y_i\right]^2$$

$$= \frac{1}{n} \sum_{i=1}^{n} e_i^2$$

Rewriting the Mean Squared Error

- Define $\vec{y} = (y_1, \dots, y_n)^T$. Then:

$$\vec{e} = \begin{pmatrix} \left(\vec{w} \cdot \text{Aug}(\vec{x}^{(1)})\right) - y_1 \\ \left(\vec{w} \cdot \text{Aug}(\vec{x}^{(2)})\right) - y_2 \\ \vdots \\ \left(\vec{w} \cdot \text{Aug}(\vec{x}^{(n)})\right) - y_n \end{pmatrix} =$$

- $\vec{h}$ is the vector of predictions.

Rewriting the Mean Squared Error

- So far: $R_{\text{sq}}(\vec{w}) = \frac{1}{n}\|\vec{e}\|^2$, and $\vec{e} = \vec{h} - \vec{y}$.
- Therefore:

$$R_{\text{sq}}(\vec{w}) = \frac{1}{n}\|\vec{h} - \vec{y}\|^2$$

- $\vec{w}$ is hidden inside of $\vec{h}$, let's pull it out.

Rewriting the Mean Squared Error

- Define the **design matrix** X:

$$X = \begin{pmatrix} \text{Aug}(\vec{x}^{(1)}) & \longrightarrow \\ \text{Aug}(\vec{x}^{(2)}) & \longrightarrow \\ \vdots & \vdots \\ \text{Aug}(\vec{x}^{(n)}) & \longrightarrow \end{pmatrix} = \begin{pmatrix} 1 & x_1^{(1)} & x_2^{(1)} & \dots & x_d^{(1)} \\ 1 & x_1^{(2)} & x_2^{(2)} & \dots & x_d^{(2)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_1^{(n)} & x_2^{(n)} & \dots & x_d^{(n)} \end{pmatrix}$$

- Then $\vec{h} = X\vec{w}$.

Rewriting the Mean Squared Error

- The mean squared error is:

$$R_{\text{sq}}(\vec{w}) = \frac{1}{n}\|X\vec{w} - \vec{y}\|^2$$

  where X is the **design matrix** containing the data, $\vec{w}$ is the **parameter vector**, and $\vec{y}$ is the vector of **observations** (or right answers).

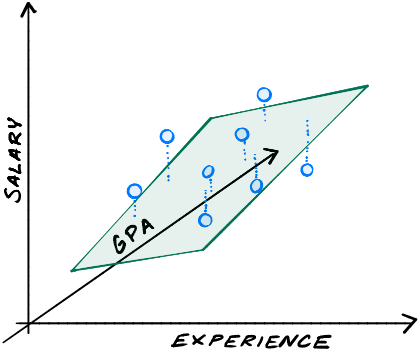
- To minimize MSE: take derivative (gradient), set to zero, solve.

Minimizing the MSE: Gradient Edition

- The vector of partial derivatives is called the **gradient**:

$$\left(\frac{\partial R_{\text{sq}}}{\partial w_0}(\vec{w}), \quad \frac{\partial R_{\text{sq}}}{\partial w_1}(\vec{w}), \quad \frac{\partial R_{\text{sq}}}{\partial w_2}(\vec{w}), \quad \dots, \quad \frac{\partial R_{\text{sq}}}{\partial w_d}(\vec{w}) \right)^T$$

- Written: $\nabla_{\vec{w}} R_{\text{sq}}(\vec{w})$ or $\frac{dR_{\text{sq}}}{d\vec{w}}(\vec{w})$
- Strategy:
  1. Compute the gradient of $R_{\text{sq}}(\vec{w})$.
  2. Set it to zero and solve for $\vec{w}$.



CSE 151A

Intro to Machine Learning

Lecture 07 – Part 02

The Gradient

Minimizing the MSE: Gradient Edition

- The vector of partial derivatives is called the **gradient**:

$$\left(\frac{\partial R_{\text{sq}}}{\partial w_0}(\vec{w}), \quad \frac{\partial R_{\text{sq}}}{\partial w_1}(\vec{w}), \quad \frac{\partial R_{\text{sq}}}{\partial w_2}(\vec{w}), \quad \ldots, \quad \frac{\partial R_{\text{sq}}}{\partial w_d}(\vec{w})\right)^T$$

- Written: $\nabla_{\vec{w}} R_{\text{sq}}(\vec{w})$ or $\frac{dR_{\text{sq}}}{d\vec{w}}(\vec{w})$
- Strategy:
  1. Compute the gradient of $R_{\text{sq}}(\vec{w})$.
  2. Set it to zero and solve for $\vec{w}$.

Minimizing the MSE

- We want to compute:

$$\frac{d}{d\vec{w}}\left[R_{\text{sq}}(\vec{w})\right] = \frac{d}{d\vec{w}}\left[\|X\vec{w} - \vec{y}\|^2\right]$$

- Step 1: Rewrite squared norm using dot product. Recall:

$$(A + B)^T = A^T + B^T$$
$$(AB)^T = B^T A^T$$
$$\vec{u} \cdot \vec{v} = \vec{v} \cdot \vec{u}$$
$$(\vec{u} + \vec{v}) \cdot (\vec{w} + \vec{z}) = \vec{u} \cdot \vec{w} + \vec{u} \cdot \vec{x} + \vec{v} \cdot \vec{w} + \vec{v} \cdot \vec{z}$$
$$\|\vec{u}\|^2 = \vec{u} \cdot \vec{u}$$

Step 1: Rewriting squared norm

$$
\begin{aligned}
\|X\vec{w} - \vec{y}\|^2 &= \\
&= \\
&= \\
&=
\end{aligned}
$$

Step 2: Take gradients

$$\frac{d}{d\vec{w}}\left[R_{\text{sq}}(\vec{w})\right] = \frac{d}{d\vec{w}}\left[\vec{w}^T X^T X \vec{w} - 2\vec{y}^T X \vec{w} + \vec{y}^T \vec{y}\right]$$

$$=$$

Claim

- $\frac{d}{d\vec{w}}\left[\vec{w}^T X^T X \vec{w}\right] = 2X^T X \vec{w}$
- $\frac{d}{d\vec{w}}\left[\vec{y}^T X \vec{w}\right] = X^T \vec{y}$
- $\frac{d}{d\vec{w}}\left[\vec{y}^T \vec{y}\right] = 0$

Example

Show $\frac{d}{d\vec{w}}\left[\vec{y}^T X \vec{w}\right] = X^T \vec{y}$

Step 2: Take gradients

$$\frac{d}{d\vec{w}}\left[R_{\text{sq}}(\vec{w})\right] = \frac{d}{d\vec{w}}\left[\vec{w}^T X^T X\vec{w} - 2\vec{y}^T X\vec{w} + \vec{y}^T\vec{y}\right]$$

$$=$$

The Normal Equations

- To minimize $R_{sq}(\vec{w})$, set gradient to zero, solve for $\vec{w}$:

$$2X^T X\vec{w} - 2X^T\vec{y} = 0 \implies X^T X\vec{w} = X^T\vec{y}$$

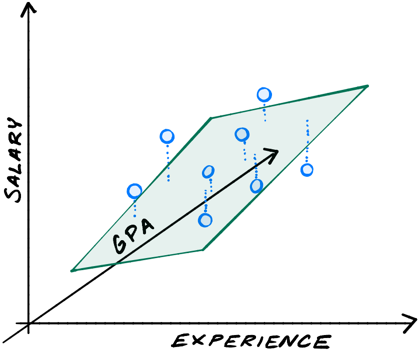
- This is a system of equations in matrix form, called the **normal equations**.

- Solution[2]: $\vec{w} = (X^T X)^{-1} X^T \vec{y}$.

[2] Don't actually compute inverse! Use Gaussian elimination.

Regression with Multiple Features

- We want to find $\vec{w}$ which minimizes $\|X\vec{w} - \vec{y}\|^2$.
- The answer: $\vec{w} = (X^T X)^{-1} X^T \vec{y}$.



CSE 151A

Intro to Machine Learning

Lecture 07 – Part 03

Interpreting Weights

Interpreting $\vec{w}$

- With d features, $\vec{w}$ has $d + 1$ entries.
- w_0 is the **bias**.
- $w_1, \ldots, w_d$ each give the **weight** of a feature.

$$H(\vec{x}) = w_0 + w_1 x_1 + \ldots + w_d x_d$$

- Sign of w_i tells us about relationship between ith feature and outcome.

Example: Predicting Sales

- For each of 26 stores, we have:
  - net sales,
  - size (sq ft),
  - inventory,
  - advertising expenditure,
  - district size,
  - number of competing stores.

- Goal: predict net sales given size, inventory, etc.

To begin...

$$H(\text{size}, \text{competitors}) = w_0 + w_1 \times \text{size} + w_2 \times \text{competitors}$$

What will be the sign of w_1 and w_2?

$$H(\text{size}, \text{competitors}) = w_0 + w_1 \times \text{size} + w_2 \times \text{competitors}$$

(DEMO)

Interpreting Weights

Which has the greatest effect on the outcome?

A) size:	w_1 = 16.20
B) inventory:	w_2 = 0.17
C) advertising:	w_3 = 11.53
D) district size:	w_4 = 13.58
E) competing stores:	w_5 = –5.31

Which features are most "important"?

- **Not necessarily** the feature with largest weight.
- Features are measured in different units, scales.
- We should **standardize** each feature.

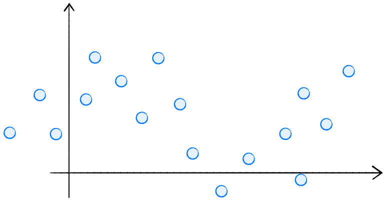
Standard Units

- Standardize each feature (store size, inventory, etc.) separately.
- No need to standardize outcome (net sales).
- Solve normal equations. The resulting $w_0, w_1, \ldots, w_d$ are called the **standardized regression coefficients**.
- They can be directly compared to one another.

(DEMO)

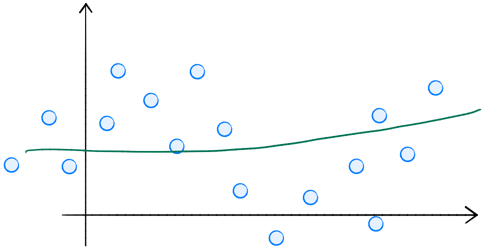
Fitting Non-Linear Patterns

- Fit a 4th-order polynomial to the data:



- Fit rule of the form $H(x) = w_1 x^4 + w_0$.
  - Define $z_i = x_i^4$.
  - Use $w_1 = \dfrac{\sum(z_i - \bar{z})(y_i - \bar{y})}{\sum(z_i - \bar{z})^2}$ and $w_0 = \bar{y} - w_1 \bar{z}$.

The Result



- The rule $H(x) = w_1 x^4 + w_0$ **underfits** the data.

- We need a more complicated rule:

$$H(x) = w_4 x^4 + w_3 x^3 + w_2 x^2 + w_1 x + w_0$$

The Trick

- Treat x, x^2, x^3, x^4 as different features.
- Create design matrix:

$$X = \begin{pmatrix} 1 & x_1 & x_1^2 & x_1^3 & x_1^4 \\ 1 & x_2 & x_2^2 & x_2^3 & x_2^4 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & x_n^2 & x_n^3 & x_n^4 \end{pmatrix}$$

- Solve $X^T X \vec{w} = X^T \vec{w}$ for $\vec{w}$, as usual.
- Works for more than just polynomials.

(DEMO)

Polynomial Regression

- More complicated patterns can be fit with higher-order polynomials.
- If there are n points, a $n + 1$ degree polynomial can fit them exactly.
- But for high-order polynomials, it becomes **very hard** to solve the normal equations (numerical accuracy).

Polynomial Regression with Multiple Features

- Suppose we want to fit a rule of the form:

$$\begin{aligned} H(\text{size}, \text{competitors}) &= w_0 + w_1 \text{size} + w_2 \text{size}^2 \\ &\quad + w_3 \text{competitors} + w_4 \text{competitors}^2 \\ &= w_0 + w_1 s + w_2 s^2 + w_3 c + w_4 c^2 \end{aligned}$$

- Make design matrix:

$$X = \begin{pmatrix} 1 & s_1 & s_1^2 & c_1 & c_1^2 \\ 1 & s_2 & s_2^2 & c_2 & c_2^2 \\ \vdots & \vdots & \vdots & \vdots & \\ 1 & s_n & s_n^2 & c_n & c_n^2 \end{pmatrix}$$

Where c_i and s_i are the competitors and size of the ith store.