Lecture 9 – Multiple Linear Regression and Feature Engineering

DSC 40A, Fall 2021 @ UC San Diego

Suraj Rampure, with help from **many others**

Announcements

- Midterm grades released; submit regrade requests by Friday night.
- Groupwork 4 out later today, due **Thursday at 11:59pm**.
- Homework 4 out later today, due **Monday at 11:59pm**.
- Come to the DSC Faculty-Student Mixer at 1pm today!
  - Zoom link: `https://ucsd.zoom.us/j/98335299546`.

Agenda

- Recap of Lecture 8.
- Using multiple features.
- Practical demo.
- Interpreting weights.
- Feature engineering.

Recap of Lecture 8

Regression and linear algebra

- Last time, we used linear algebra to fit a prediction rule of the form

$$\underline{H(x) = w_0 + w_1 x}$$

- To do so, we first defined a **design matrix** X, **parameter vector** $\vec{w}$, and **observation vector** $\vec{y}$ as follows:

$w_0 + w_1 x_1$

$w_0 + w_1 x_2$

design matrix

$$X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \dots & \dots \\ 1 & x_n \end{bmatrix}, \qquad \vec{w} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix}, \qquad \vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{bmatrix}$$

parameter vector

observation vector

- We also re-wrote our prediction rule as a matrix-vector multiplication, defining the **hypothesis vector** $\vec{h}$ as

$$\vec{h} = X\vec{w}$$

Minimizing mean squared error

- With our new linear algebra formulation of regression, our mean squared error now looks like:

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

- To find $\vec{w}^*$, the optimal parameter vector, we took the gradient of $R_{sq}(\vec{w})$ with respect to $\vec{w}$, set it equal to 0, and solved.
- The result is the **normal equations**:

Aw = b

$$X^T X \vec{w}^* = X^T y$$

- When $X^T X$ is invertible, an equivalent form is

$$\vec{w}^* = (X^T X)^{-1} X^T y$$

  - This gives the same w_0^* and w_1^* as our formulas from Lecture 6.

Using multiple features

Using multiple features

- How do we predict salary given **multiple** features?
- We believe salary is a function of experience *and* GPA.
- In other words, we believe 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

$$H_1(\text{experience}, \text{GPA}) = \$2,000 \times (\text{experience}) + \$\overset{10000}{\cancel{40,000}} \times \frac{\text{GPA}}{\cancel{4.0}}$$

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

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

bad

Linear prediction rules

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

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

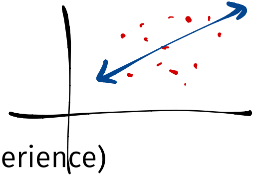
$$w_0 \cdot \square + w_1 \cdot \square + w_2 \cdot \square$$

- Note that H is **linear in the parameters** w_0, w_1, w_2.
  - H is a linear combination of features (1, experience, GPA) with ws as the coefficients (w_0, w_1, and w_2).

- As a result, we can solve the **normal equations** to find w_0^*, w_1^*, and w_2^*!

- Linear regression with multiple features is called **multiple linear regression**.

Geometric interpretation



Question: The prediction rule

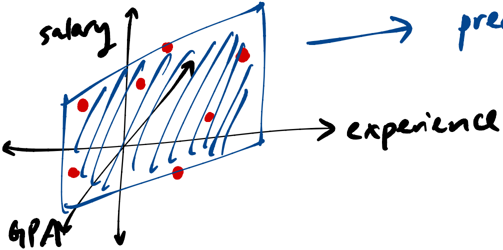
$$H(\text{experience}) = w_0 + w_1(\text{experience})$$

looks like a line in 2D.

1. How many dimensions do we need to graph

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

2. What is the shape of the prediction rule?



prediction rule is a plane!

Example dataset

- For each of n people, collect each feature, plus salary:

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

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

$$\vec{x}_1 = \begin{bmatrix} 3 \\ 3.7 \end{bmatrix}, \quad \vec{x}_2 = \begin{bmatrix} 6 \\ 3.3 \end{bmatrix}, \quad \vec{x}_3 = \begin{bmatrix} 10 \\ 3.1 \end{bmatrix}$$

simple linear regression:

$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$

multiple linear regression:

$(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_n, y_n)$

The hypothesis vector

- When our prediction rule is

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

the hypothesis vector $\vec{h} \in \mathbb{R}^n$ can be written

$$\vec{h} = \begin{bmatrix} H(\text{experience}_1, \text{GPA}_1) \\ H(\text{experience}_2, \text{GPA}_2) \\ \cdots \\ H(\text{experience}_n, \text{GPA}_n) \end{bmatrix} = \begin{bmatrix} 1 & \text{experience}_1 & \text{GPA}_1 \\ 1 & \text{experience}_2 & \text{GPA}_2 \\ \cdots & \cdots & \cdots \\ 1 & \text{experience}_n & \text{GPA}_n \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix}$$

$w_0 + w_1(\text{experience}_1) + w_2(\text{GPA}_1)$

How do we find $\vec{w}^*$?

- To find the best parameter vector, $\vec{w}^*$, we can use the design matrix and observation vector

$$X = \begin{bmatrix} 1 & \text{experience}_1 & \text{GPA}_1 \\ 1 & \text{experience}_2 & \text{GPA}_2 \\ \dots & \dots & \dots \\ 1 & \text{experience}_n & \text{GPA}_n \end{bmatrix}, \vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{bmatrix}$$

and solve the **normal equations**

$$X^T X \vec{w}^* = X^T \vec{y}$$

- Notice that the rows of the design matrix are the (transposed) feature vectors, with an additional 1 in front.

Notation for multiple linear regression

- We will need to keep track of multiple[1] features for every individual in our data set.
- As before, subscripts distinguish between individuals in our data set. We have n individuals (or **training examples**).
- Superscripts distinguish between features.[2] We have d features.
    - experience = $x^{(1)}$
    - GPA = $x^{(2)}$

$x_i^{(j)}$ = feature j for data point i

$x_4^{(2)}$ = GPA of person 4

[1]In practice, we might use hundreds or even thousands of features.
[2]Think of them as new variable names, such as new letters.

Augmented feature vectors

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

$$\vec{x} = \begin{bmatrix} x^{(1)} \\ x^{(2)} \\ \vdots \\ x^{(d)} \end{bmatrix} \qquad \text{Aug}(\vec{x}) = \begin{bmatrix} 1 \\ x^{(1)} \\ x^{(2)} \\ \vdots \\ x^{(d)} \end{bmatrix} \qquad \vec{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix}$$

x_i $\quad \text{Aug}(\vec{x}_i) = \begin{bmatrix} 1 \\ x_i \end{bmatrix}$

- Then, our prediction rule is

$$\begin{aligned} H(\vec{x}) &= w_0 + w_1 x^{(1)} + w_2 x^{(2)} + \ldots + w_d x^{(d)} \\ &= \vec{w} \cdot \text{Aug}(\vec{x}) \end{aligned}$$

$\text{Aug}(\vec{x}_i) \cdot \begin{bmatrix} w_0 \\ w_1 \end{bmatrix}$

The general problem

- We have n data points (or **training examples**): $(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n)$ where each $\vec{x}_i$ is a feature vector of d features:

$$\vec{x}_i = \begin{bmatrix} x_i^{(1)} \\ x_i^{(2)} \\ \ldots \\ x_i^{(d)} \end{bmatrix}$$

- We want to find a good linear prediction rule:

$$\begin{aligned} H(\vec{x}) &= w_0 + w_1 x^{(1)} + w_2 x^{(2)} + \ldots + w_d x^{(d)} \\ &= \vec{w} \cdot \text{Aug}(\vec{x}) \end{aligned}$$

The general solution

columns = features

- Use design matrix

rows = data points

$$X = \begin{bmatrix} 1 & x_1^{(1)} & x_1^{(2)} & \dots & x_1^{(d)} \\ 1 & x_2^{(1)} & x_2^{(2)} & \dots & x_2^{(d)} \\ \dots & \dots & \dots & & \dots \\ 1 & x_n^{(1)} & x_n^{(2)} & \dots & x_n^{(d)} \end{bmatrix} = \begin{bmatrix} \text{Aug}(\vec{x}_1)^T \\ \text{Aug}(\vec{x}_2)^T \\ \dots \\ \text{Aug}(\vec{x}_n)^T \end{bmatrix}$$

and observation vector to solve the **normal equations**

$$X^T X \vec{w}^* = X^T \vec{y}$$

to find the optimal parameter vector.

Interpreting the parameters

"hyperplane"

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

$$H(\vec{x}) = w_0 + \underbrace{w_1}\, x^{(1)} + \ldots + w_d x^{(d)}$$

slopes for each feature

- The sign of w_i tells us about the relationship between ith feature and the output of our prediction rule.

Practical demo

Example: predicting sales

$\rightarrow n$

- For each of 26 stores, we have:
  - net sales, $\rightarrow y$
  - square feet,
  - inventory,
  - advertising expenditure,
  - district size, and
  - number of competing stores.

- Goal: predict net sales given square footage, inventory, etc.

- To begin:

$$H(\text{square feet, competitors}) = w_0 + w_1(\text{square feet}) + w_2(\text{competitors})$$

Example: predicting sales

$$H(\text{square feet}, \text{competitors}) = w_0 + w_1(\text{square feet}) + w_2(\text{competitors})$$

Discussion Question

What will be the sign of w_1^* and w_2^*?

A) $w_1^* = +, \quad w_2^* = -$

B) $w_1^* = +, \quad w_2^* = +$

C) ~~$w_1^* = -, \quad w_2^* = -$~~

D) ~~$w_1^* = -, \quad w_2^* = +$~~

To answer, go to `menti.com` **and enter 5115 8817.**

Follow along with the demo by clicking the **code** link on the course website next to Lecture 9.

Interpreting weights

Discussion Question

Which feature has the greatest effect on the outcome?

A) square feet: $w_1^* = 16.202$
B) competing stores: $w_2^* = -5.311$
C) inventory: $w_3^* = 0.175$
D) advertising: $w_4^* = 11.526$
E) district size: $w_5^* = 13.580$

To answer, go to `menti.com` and enter 5115 8817.

Which features are most "important"?

- The most important feature is **not necessarily** the feature with largest weight.

- Features are measured in different units, scales.
  - Suppose I fit one prediction rule, H_1, with sales in dollars, and another prediction rule, H_2, with sales in thousands of dollars.
  - Sales is just as important in both prediction rules.
  - But the weight of sales in H_1 will be 1000 times smaller than the weight of sales in H_2.
  - Intuitive explanation: $5 \times 45000 = (5 \times 1000) \times 45$.

- **Solution**: we should **standardize** each feature, i.e. convert each feature to standard units.

Standard units

"z-scores"

- Recall from Lecture ~~6~~ 7: to convert a feature $x_1, x_2, \ldots, x_n$ to standard units, we use the formula

# of SDs x_i is above the mean

$$x_i \text{ in standard units} = \frac{x_i - \bar{x}}{\sigma_x}$$

mean(x)

SD(x)

- Example: 1, 7, 7, 9
  - Mean: 6

  $$SD(x) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

  - Standard deviation:

$$\sqrt{\frac{1}{4}\left((-5)^2 + (1)^2 + (1)^2 + (3)^2\right)} = 3$$

  - Standardized data:

$$\frac{1-6}{3} = -\frac{5}{3}, \qquad \frac{7-6}{3} = \frac{1}{3}, \qquad \frac{7-6}{3} = \frac{1}{3}, \qquad \frac{9-6}{3} = 1$$

Standard units for multiple linear regression

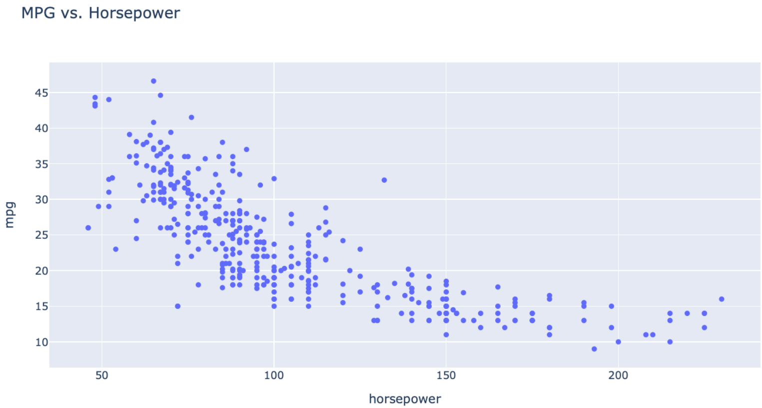
- The result of standardizing each feature (separately!) is that the units of each feature are on the same scale.
  - There's no need to standardize the outcome (net sales), since it's not being compared to anything.

- Then, solve the normal equations. The resulting $w_0^*, w_1^*, \ldots, w_d^*$ are called the **standardized regression coefficients**.

- Standardized regression coefficients can be directly compared to one another.

Let's jump back to our demo notebook.

Feature engineering



Question: Would a linear prediction rule work well on this dataset?

A quadratic prediction rule

- It looks like there's some sort of quadratic relationship between horsepower and mpg in the last scatter plot. We want to try and fit a prediction rule of the form

$$H(x) = w_0 + w_1 x + w_2 x^2$$

  - Note that this still a linear model, because it is **linear in the parameters**!

$w_0 \cdot \boxed{} + w_1 \cdot \boxed{} + w_2 \cdot \boxed{} + \dots$

- We can do that, by choosing our two "features" to be x_i and x_i^2, respectively.

  horsepower$_i$

  - In other words, $x_i^{(1)} = \cancel{x_i}$ and $x_i^{(2)} = \cancel{x_i^2}$ horsepower$_i^2$

  - More generally, we can create new features out of existing features.

A quadratic prediction rule

- Desired prediction rule: $H(x) = w_0 + w_1 x + w_2 x^2$.
- The resulting design matrix looks like this:

$$X = \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ \dots & & \vdots \\ 1 & x_n & x_n^2 \end{bmatrix} \qquad \vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

- To find optimal parameter vector $\vec{w}^*$: solve the **normal equations**!

$$X^T X \vec{w}^* = X^T \vec{y}$$

More examples

- What if we want to use a prediction rule of the form $H(x) = w_0 + w_1 x + w_2 x^2 + w_3 x^3$?

$$X = \begin{bmatrix} 1 & x_1 & x_1^2 & x_1^3 \\ 1 & x_2 & x_2^2 & x_2^3 \\ & & \vdots & \\ 1 & x_n & x_n^2 & x_n^3 \end{bmatrix} \qquad \vec{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{bmatrix}$$

- What if we want to use a prediction rule of the form $H(x) = w_1 \frac{1}{x^2} + w_2 \sin x + w_3 e^x$?

$$X = \begin{bmatrix} \frac{1}{x_1^2} & \sin x_1 & e^{x_1} \\ \frac{1}{x_2^2} & \sin x_2 & e^{x_2} \\ & \vdots & \end{bmatrix} \qquad \vec{w} = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix}$$

Feature engineering

- More generally, we can create new features out of existing information in our dataset. This process is called **feature engineering**.
  - In this class, feature engineering will mostly be restricted to creating non-linear functions of existing features (as in the previous example).
  - In the future you'll learn how to do other things, like encode categorical information.

Summary

$$H(x) = e^{w_1 x} + \sin(w_2 x)$$

$$w_1 \cdot [\;] + w_2 \cdot [\;]$$

Summary

- The normal equations can be used to solve the **multiple linear regression** problem, where we use multiple features to predict an outcome.
- We can interpret the parameters as weights. The signs of weights give meaningful information, but we can only compare weights if our features are standardized.
- We can create non-linear features out of existing features. This process is called feature engineering.
  - A prediction rule is linear as long as it is **linear in the parameters**. The features themselves don't have to be linear.

Next time

- A few more examples of feature engineering.
- A high-level overview of machine learning.
- New idea: clustering.