Linear Regression

Raguvir Kunani

Data 100, Spring 2020

Simple Linear Regression Quick Review

Simple linear regression involves finding a "line of best fit" that explains the relationship between **2** variables x and y. In fancy math terms:

$$a^*, b^* = \arg\min_{a,b} \sum_{i=1}^{n} (y_i - (\underbrace{a + bx_i}_{\text{predicted } y \text{ for input } x_i}))^{\boxed{2}} \leftarrow \text{MSE}$$

sum over all n training data points ↗

All this really means is we are trying to find the values of a and b so that a line with the equation $\hat{y} = a + bx$ best fits the data.

$x^* = \arg\min_x f(x)$ means x^* is equal to the value of x that minimizes $f(x)$

Residuals

The **residual** is a measure of how "off" our prediction was. In simple linear regression, the residual for the i^{th} point is:

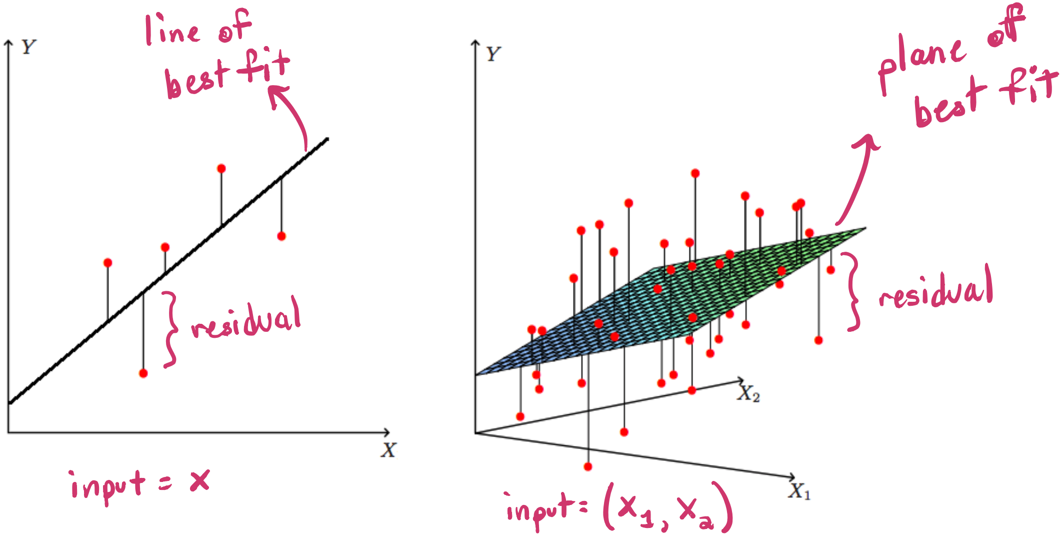
$$\text{true value} \longrightarrow y_i - (\underbrace{a + bx_i}_{\text{predicted value}})$$

So really what linear regression is doing is finding the a and b that minimize the sum of (squared) residuals. This view of linear regression as minimizing residuals is very helpful to understanding linear regression in higher dimensions.

Multiple

~~Multivariate~~ Linear Regression in a Graph

In multiple dimensions, linear regression goes from finding a *line* of best fit to finding a *plane* of best fit.



Keep in mind that the quantity you are trying to predict is still a scalar quantity, but now you have multiple inputs (labeled $x_1, x_2, \ldots, x_d$).

~~Multivariate~~ Multiple Linear Regression in Math

Remember that the simple linear regression problem minimizes the sum of the squared residuals $y_i - (a + bx_i)^2$. We can use the same residuals interpretation to formulate the ~~multivariate~~ multiple linear regression problem, but we need to rewrite the residuals for the multidimensional case.

Let's say our model is of the form $\hat{y} = \overset{\text{bias term}}{\theta_0} + \theta_1 \overset{\text{inputs}}{x_1} + \theta_2 x_2$

$x_i = \begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix}$

Then the residuals become $\underset{\text{true value}}{y_i} - (\underbrace{\theta_0 + \theta_1 x_{i,1} + \theta_2 x_{i,2}}_{\text{predicted value}})$

The sum of the squared residuals is $\underset{\text{sum over all } n \text{ training data points}}{\sum_{i=1}^{n}} \underbrace{y_i - (\theta_0 + \theta_1 x_{i,1} + \theta_2 x_{i,2})}_{\text{residual}}{}^{\underset{\text{MSE}}{\boxed{2}}}$

~~Multivariate~~ Multiple Linear Regression V1

Now we can write our new ~~multivariate~~ multiple linear regression problem:

$$\theta_0^*, \theta_1^*, \theta_2^* = \arg\min_{\theta_0, \theta_1, \theta_2} \sum_{i=1}^{n} \left(y_i - (\underbrace{\theta_0 + \theta_1 x_{i,1} + \theta_2 x_{i,2}}_{\uparrow \text{ gets uglier as \# dimensions increases}})\right)^2$$

This looks horrendous, and will only get worse with more dimensions.

This is where linear algebra comes into play. Let's rewrite the above problem using matrix vector notation.

Simplifying the Expression for Model's Prediction

Let's start by collecting all the parameters into a vector $\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{bmatrix}$.

Then we can write $\theta_0 + \theta_1 x_{i,1} + \theta_2 x_{i,2}$ as $\begin{bmatrix} 1 \\ x_{i,1} \\ x_{i,2} \end{bmatrix} \cdot \theta$.

dot product

$\begin{bmatrix} a \\ b \end{bmatrix} \cdot \begin{bmatrix} c \\ d \end{bmatrix} = ac + bd$

If we define a vector $\phi_i = \begin{bmatrix} 1 \\ x_{i,1} \\ x_{i,2} \end{bmatrix}$,

then we can write $\underbrace{\theta_0 + \theta_1 x_{i,1} + \theta_2 x_{i,2}}$ even more simply as $\underbrace{\phi_i \cdot \theta}$.

model's prediction for input x_i

Multiple ~~Multivariate~~ Linear Regression V2

We can update our multiple ~~multivariate~~ linear regression problem to be:

$$\theta^* = \arg\min_{\theta} \sum_{i=1}^{n} (y_i - (\phi_i \cdot \theta))^2$$

This looks a lot better, but we can simplify even more.

Let's write the true values and our predicted values as n-dimensional vectors: $y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$ and $\hat{y} = \begin{bmatrix} \phi_1 \cdot \theta \\ \phi_2 \cdot \theta \\ \vdots \\ \phi_n \cdot \theta \end{bmatrix}$.

$\hat{y}$ is a common notation for predicted values

n-dimensional because there are n data points

~~Multivariate~~ Multiple Linear Regression V3

Recall $||z||$ is the norm of the vector z and $||z||^2 = \sum_{i=1}^{n} z_i^2$.

Then we rewrite our linear regression problem as:

$$\theta^* = \arg\min_{\theta} ||y - \hat{y}||^2$$

There's just one last simplification. Note that

$$\hat{y} = \begin{bmatrix} \phi_1 \cdot \theta \\ \phi_2 \cdot \theta \\ \vdots \\ \phi_n \cdot \theta \end{bmatrix} = \begin{bmatrix} - & \phi_1 & - \\ - & \phi_2 & - \\ & \vdots & \\ - & \phi_n & - \end{bmatrix} \cdot \theta = \Phi \cdot \theta$$

where Φ is a matrix that contains the ϕ_i as its rows.

~~Multivariate~~ Multiple Linear Regression V4

Using Φ, we can write the ~~multivariate~~ multiple linear regression problem as:

$$\theta^* = \arg\min_{\theta} ||y - \Phi \cdot \theta||^2 = \boxed{\arg\min_{\theta} ||y - \Phi\theta||^2}$$

This is the final version of the problem! This representation is much more compact and *does not change* as the number of dimensions increase.

Note that this version of the problem is exactly equivalent to the original version we started with. I did not change the problem; I just rewrote the problem into a form that allows us to use linear algebra.

A Closer Look at $||y - \Phi\theta||^2$

Since $||z||^2$ represents the length of z, we can think of $||x - z||^2$ as the "length of the difference" between x and z.

Put more simply, $||x - z||^2$ is the distance between x and z.

This means linear regression, in trying to minimize $||y - \Phi\theta||^2$, is really trying to minimize the distance between y and $\Phi\theta$.

This sounds very similar to minimizing residuals! In this case, the residual is a vector, specifically $y - \Phi\theta$.

How to Compute θ^*: Part 1

Since our goal is to choose θ^* such that $\Phi\theta^*$ is as close to y as possible, it seems natural to set $\Phi\theta^* = y$ and solve for θ^*.

However, this equation rarely has a unique solution. This could happen for 2 main reasons:

1. $y \notin \mathrm{range}(\Phi)$. This means there is no vector v such that $\Phi v = y$. (More on this in the next slide).
2. Φ is not invertible. If this is the case, create a new Φ and start over.

Thus, the best we can do is choose θ^* that minimizes the distance between y and $\Phi\theta^*$.

A Closer Look at $y \notin \text{range}(\Phi)$

The matrix-vector product $\Phi\theta$ can be thought of as a linear combination of the columns of Φ.

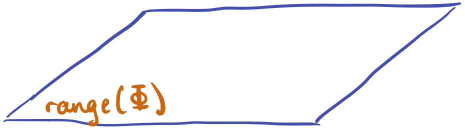
Φ_i is the i^{th} column of Φ matrix

θ_i is the i^{th} element of θ vector

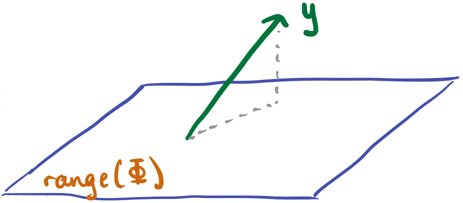
$$\Phi\theta = \theta_0 \Phi_0 + \theta_1 \Phi_1 + \dots + \theta_d \Phi_d$$

The set of all possible linear combinations of the columns of Φ is called $\text{range}(\Phi)$ because it contains all possible outputs of multipltying Φ by any vector.

Graphically, $\text{range}(\Phi)$ is depicted as:



Then $y \notin \text{range}(\Phi)$ is depicted as:

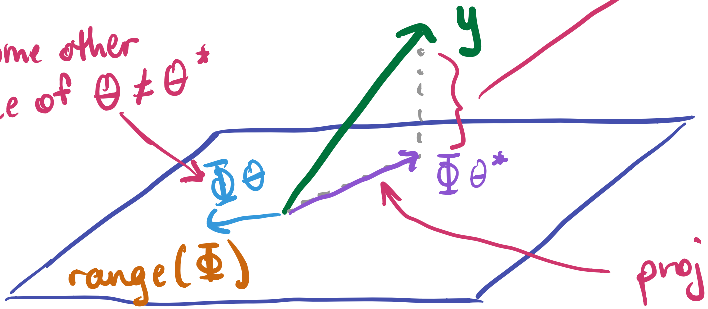


How to Compute θ^*: Part 2

Since we are stuck with $y \notin \text{range}(\Phi)$, we can't set $\Phi\theta^* = y$.

So how else are we supposed to choose θ^* such that the distance between y and $\Phi\theta^*$ is minimized?

Projections!



some other choice of $\theta \neq \theta^*$

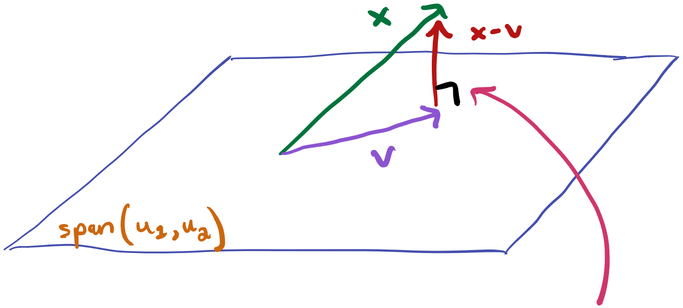
this distance is the straight line distance from y to range(Φ)

projection of y onto range(Φ)

you can see that out of all vectors in range(Φ), $\Phi\theta^*$ has the closest distance to y

Projections and Their Important Properties

The projection of a vector x onto $\text{span}(u_1, u_2)$ is the vector in the span of u_1 and u_2 that is closest in distance to x.



v is the projection of x onto span(u_1, u_2)

If v is the projection of x onto $\text{span}(u_1, u_2)$, then $x - v$ is orthogonal to $\text{span}(u_1, u_2)$. This is because the straight line distance is the closest distance between any 2 points.

Using Projections to Solve Linear Regression

From the previous slide, we know that $y - \Phi\theta^*$ is orthogonal to $\text{range}(\Phi)$. (because $\Phi\theta^*$ is the projection of y onto range (Φ)

Since $\text{range}(\Phi)$ is the set of all linear combinations of the columns of Φ, it must also be true that $y - \Phi\theta^*$ is orthogonal to each column of Φ:

$$\Phi_1 \cdot (y - \Phi\theta^*) = \Phi_2 \cdot (y - \Phi\theta^*) = \ldots = \Phi_d \cdot (y - \Phi\theta^*) = 0.$$

↑ d columns of Φ

We can write all these equations using matrix form as:

$$\begin{bmatrix} \text{---} & \Phi_1 & \text{---} \\ \text{---} & \Phi_2 & \text{---} \\ & \vdots & \\ \text{---} & \Phi_d & \text{---} \end{bmatrix} \cdot (y - \Phi\theta^*) = \Phi^T (y - \Phi\theta^*) = 0$$

↗ Φ^T because Φ_i is a column of Φ

The Normal Equations

Solving the equation from last slide:

$$\Phi^T(y - \Phi\theta^*) = 0$$

$$\Phi^T y - (\Phi^T\Phi)\theta^* = 0$$

$$\boxed{(\Phi^T\Phi)\theta^* = \Phi^T y}$$

These are the normal equations you saw in lecture! Solving for θ^*,

$$\theta^* = (\Phi^T\Phi)^{-1}\Phi^T y$$

Some Terminology

There are a lot of different names for the process of finding θ^*:

1. Training a least squares model
2. Fitting a least squares model
3. Finding θ that minimizes MSE

All of these things just mean setting up and solving the normal equations $(\Phi^T\Phi)\theta^* = \Phi^T y$.

When you call `model.fit(X, y)`, solving the normal equations is what goes on behind the scenes.

What Happens After Finding θ^*?

After you've computed θ^*, you can use your model to make predictions!

new data point

ϕ vector

prediction for x

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix} \rightarrow \phi = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{bmatrix} \rightarrow \theta^* \cdot \phi = \hat{y}$$

(1 for bias term)

When you call `model.predict(x)`, this is what is happening behind the scenes.

Feedback Form

This *anonymous* form is for me to learn what I can do to ensure you all get the most out of this class (and specifically, discussion and lab). This form will be open all semester, and I'll be checking it regularly. Be as ruthless as you want, I promise my feelings won't get hurt.

Feedback Form: tinyurl.com/raguvirTAfeedback