Math for Data Science

Curvilinear Models and Summations

Joanna Bieri DATA100

Important Information

- Email: joanna_bieri@redlands.edu
- Office Hours take place in Duke 209 unless otherwise noted – Office Hours Schedule

Today's Goals:

- Continue Empirical Modeling
- Cataloging Functions and Transformations.
- Nonlinear Models - Linearization
- Summation Notation

Empirical Modeling with Nonlinear Functions

- Square Root
$$y = f(x) = a\sqrt{x} + b$$
- Absolute Value
$$y = f(x) = a|x| + b$$
- Polynomials higher than degree 1
$$y = f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + ...$$
- Sine
$$y = f(x) = a\sin(bx) + c$$
- Cosine
$$y = f(x) = a\cos(bx) + c$$

NOTE We will save Exponential and Logarithmic Functions for a day of their own.

Exploring Nonlinear Curves

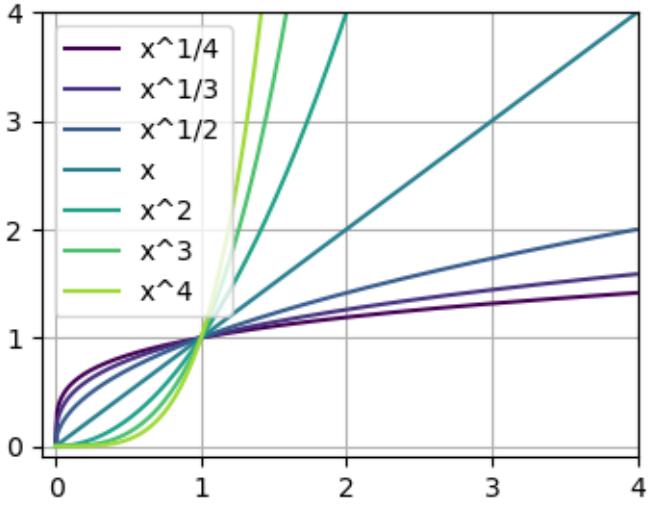
Powers of x

We will start by considering functions that are powers of x^n where n can be any real number, but typically we think of it as an integer $n = \pm 1, \pm 2, pm3$ or a fraction $n = \frac{1}{m}$ where m is an integer. This includes the polynomials and square roots.

Here are our big questions:

- What does the power do to the shape of the function?
- How fast do these increase or decrease as x gets large?
- How can we shift them: up, down, left, right?
- How can we stretch or compress them?

Exploring Nonlinear Curves

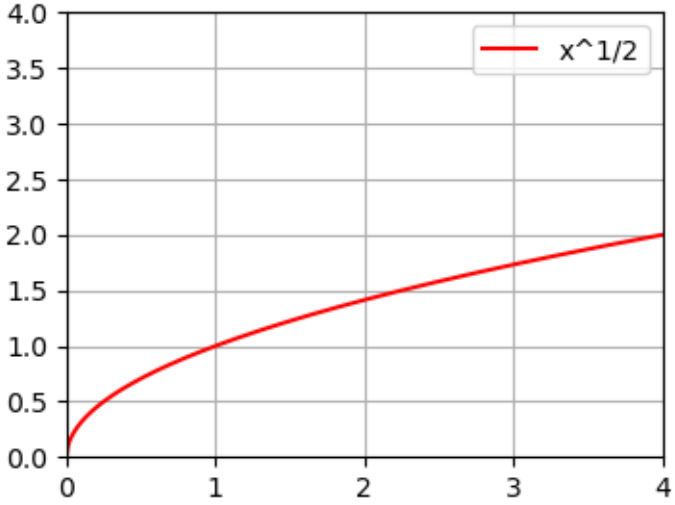


Exploring Nonlinear Curves

```
x = np.arange(0,4,0.01)
n = 1/2
f = x**n

plt.plot(x,f,color='red',label='x^1/2')
plt.legend()
plt.ylim(0,4)
plt.xlim(0,4)
plt.grid()
plt.show()
```

Exploring Nonlinear Curves



Transforming Powers of x

What we noticed about each of the funtions above:

- Increasing the power of the exponent in the polynomial changes the shape.
  - $x < 1$ the functions slope (incline) decreases as x increases.
  - $x > 1$ the functions slope (incline) increases as x increases.
  - As x increases they all keep increasing.
  - They all go through (0.0)

Transforming Powers of x

How could we change some of the shape given a single function? Lets redo the plot of

$$y = \sqrt{x}$$

that we did above, but transform it!

$$y = a\sqrt{x - b} + c$$

Here a will stretch or compress the curve, b will shift the curve left or right, and c will move it up and down.

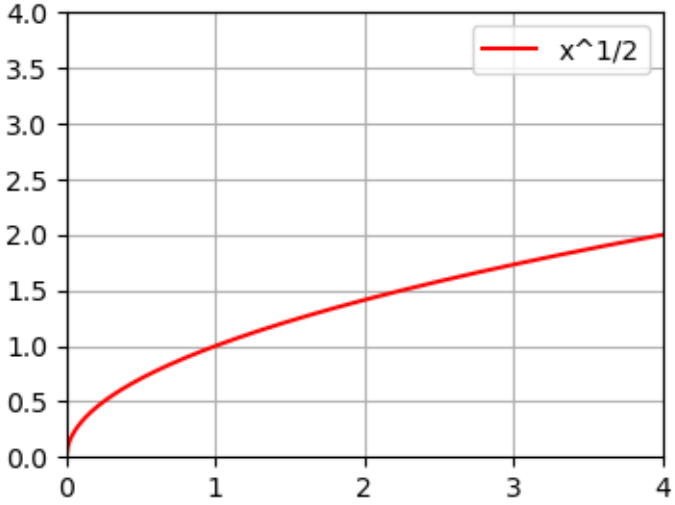
Transforming Powers of x

```
a=1
b=0
c=0

x = np.arange(0,4,0.01)
n = 1/2
f = a*(x-b)**n+c

plt.plot(x,f,color='red',label='x^1/2')
plt.legend()
plt.ylim(0,4)
plt.xlim(0,4)
plt.grid()
plt.show()
```

Transforming Powers of x



Transforming Powers of x

On paper:

Questions:

1. How would you create a square root function that "starts" at the point $(2, 3)$?
2. Why does Python give you an error if you choose $b = 1$ and have your x-values going from $(0, 4)$?
3. How would you fit the following function:

$$y = a\sqrt{x} + c$$

so that it goes through the points $(1, 0)$ and $(4, 4)$?

NOTE: It is interesting that in questions 3 we were solving a linear system of equations even though we were fitting a nonlinear curve!

General Transformations

- If we know the graph of $f(x)$ then
  - we can shift it to the right b units by using $f(x-b)$.
  - we can stretch it vertically by multiplying by a when $a > 1$
  - we can shrink it vertically by multiplying by a when $a < 1$
  - we can move it up an down by adding c to it.

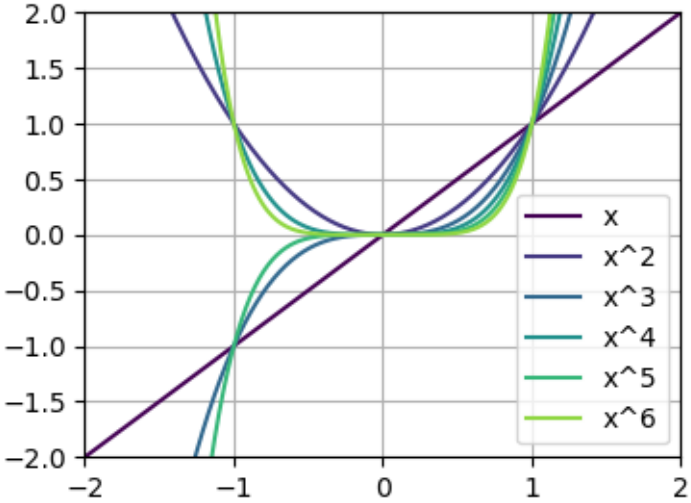
Transforming Powers of x

One paper and in Python

Choose one of the other x^n style functions and do the following:

1. Imagine what you want the graph to look like: shifted, stretched, moved, etc...
2. Draw a rough sketch of what you think your new function will look like.
3. Plot your function using Pyhton. How close was your sketch?

Integer Powers of x



Integer Powers of x

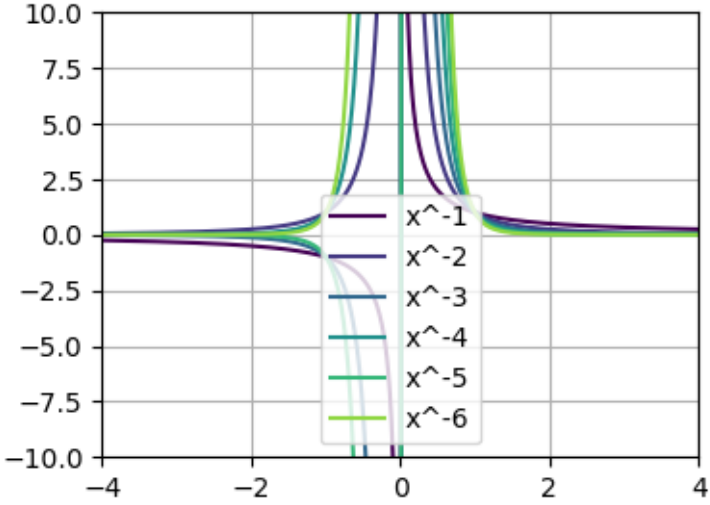
Positive n

Even Functions are functions that are equal when reflected across $x = 0$. You could fold them in half along $x = 0$ and they would overlap.
Odd Functions are functions that are opposite when reflected across $x = 0$. You could fold them in half along $x = 0$ and they would NOT overlap.

Questions:

1. Which of the functions above are even? Which are odd?
2. Do you see a pattern?

Integer Powers of x



Integer Powers of x

Negative n

Questions:

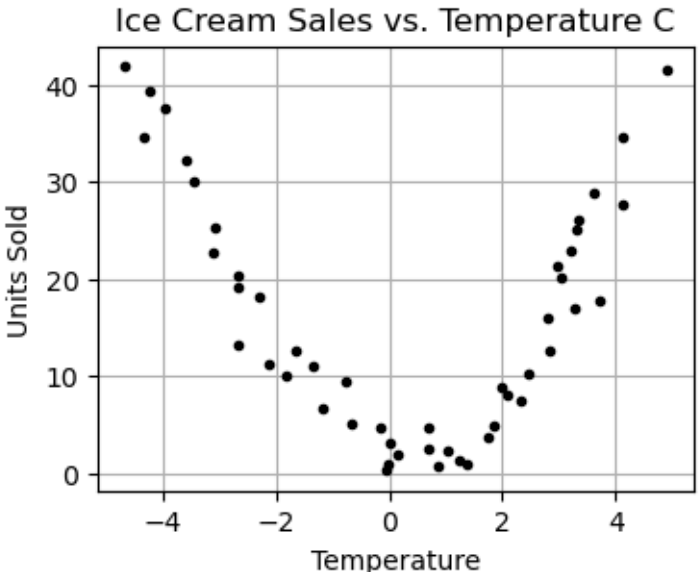
1. Can we talk about even and odd functions here?
2. What happens as x gets really small?
3. What is happening across $x = 0$?

Lets look at some data

Here is data that records the daily temperature and the number of units of ice cream sold for that day.

	Temperature (°C)	Ice Cream Sales (units)
0	-4.662263	41.842986
1	-4.316559	34.661120
2	-4.213985	39.383001
3	-3.949661	37.539845
4	-3.578554	32.284531
5	-3.455712	30.001138
6	-3.108440	22.635401
7	-3.081303	25.365022
8	-2.672461	19.226970
9	-2.652287	20.279679

Lets look at some data



Lets look at some data

Questions:

After looking at the graph answer the following questions:

1. Can we use a linear regression?
2. Should we use a linear regression?
3. What function seems to kindof fit this data? Can you come up with a guess of the function of best fit?

Lets try a linear regression (brute force)

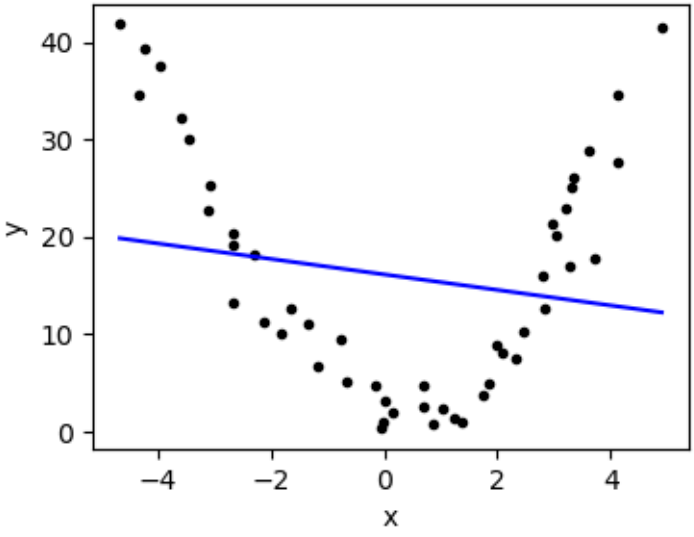
Correlation:

```
-0.17518429270784366
```

Beta values

```
array([-0.79645711, 16.12174939])
```

Lets try a linear regression (brute force)



Lets try a linear regression (brute force)

It worked!!! BUT....

Questions:

1. Are we happy with this fit?
2. What do all those numbers/graphs mean to you?
3. What went wrong?

Nonlinear Models - Linearization

Even though the model above is nonlinear we can still technically use linear regression. BUT HOW?

We think of linear regression in one variable (1D) as using just a straight line:

$$y = \beta_0 + \beta_1 x$$

but what we have above is something more like

$$y = \beta_0 + \beta_1 x^2$$

Nonlinear Models - Linearization

What if we just imagine that our data had another column that was just the temperature squared:

	Temperature (°C)	Ice Cream Sales (units)	Temperature Squared (°C^2
0	-4.662263	41.842986	21.736693
1	-4.316559	34.661120	18.632685
2	-4.213985	39.383001	17.757668
3	-3.949661	37.539845	15.599823
4	-3.578554	32.284531	12.806047
5	-3.455712	30.001138	11.941943
6	-3.108440	22.635401	9.662400
7	-3.081303	25.365022	9.494430
8	-2.672461	19.226970	7.142047
9	-2.652287	20.279679	7.034625

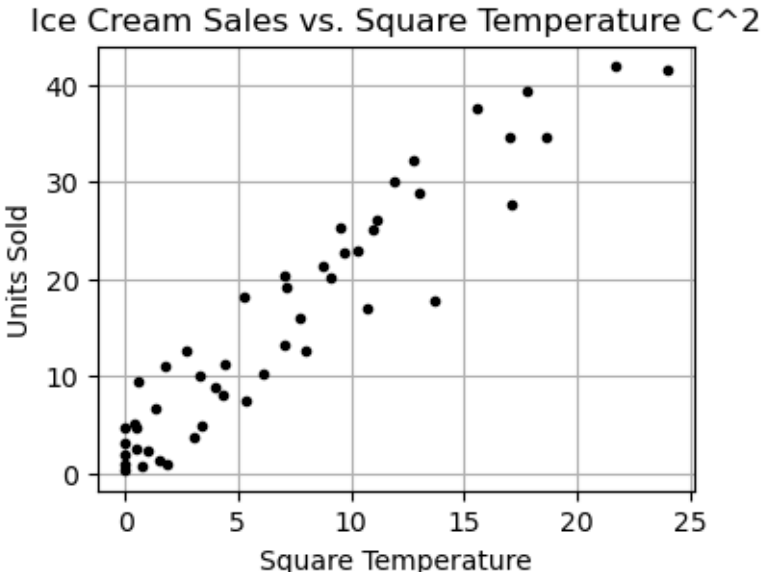
Nonlinear Models - Linearization

Are y and x^2 correlated?

```
0.948267846768457
```

What does a scatter plot of y vs x^2 look like?

Nonlinear Models - Linearization



Nonlinear Models - Linearization

The linearized model

So lets look at the nonlinear model:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2$$

This thing is linear in β. Instead we could try to predict the Ice Cream Sales using both the temperature and the temperature squared? Just treat them each as separate variables (even though we secretly know they are related). This would be more like

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

Nonlinear Models - Linearization

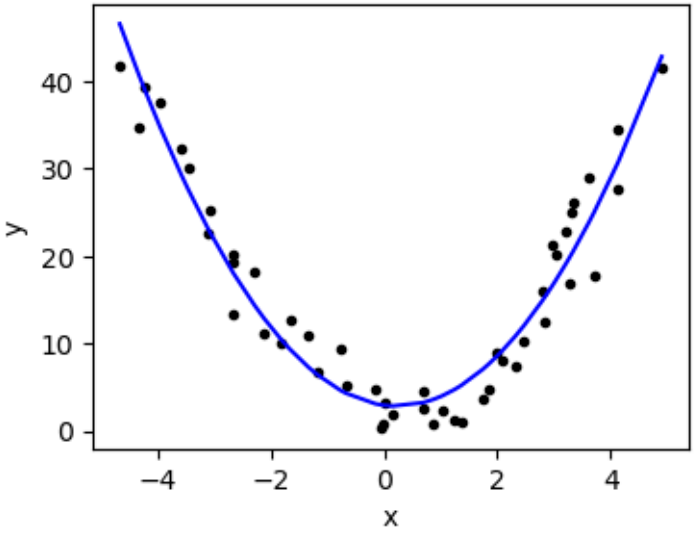
We could "trick" linear regression into doing nonlinear regression! This is what polyfit does when we tell it we want a second order polynomial.

```
{python] np.polyfit(x,y,2)
```

beta values

```
array([ 1.82952623, -0.82468167,  2.95177416])
```

Nonlinear Models - Linearization



Pause for a second - Summation Notation

Often mathematicians try to write things down in beautiful and compact notation. When we are doing general linear regression we could have as many variables as we want (aka as many dimensions). It gets really annoying to write out

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \beta_5 x_5 + ...$$

Imagine if you had 100 variables! Summation notation makes this much easier to write (okay a little harder to understand at first)

$$y = \beta_0 + \sum_{i=1}^{100} \beta_1 x_i$$

This is how we might write out the idea of a polynomial regression:

Pause for a second - Summation Notation

Questions:

1. What is this sum if your write it out long hand?
2. Try writing out the terms in this sum

$$\sum_{n=2}^{5}(-1)^n x^n$$

3. Can you change this from long hand to summation notation?

$$y = 2 + 4x + 8x^2 + 16x^3 + 32x^4$$

We will see more of this as we go on in class... but I wanted to start working in small doses!

Another measure of fit - Mean Squared Error

R^2 measures the proportion of the total variation in the data that the model can explain. The values range from zero to one with higher being netter.

MSE or mean squared error measures the average squared difference between the predicted and actual values. Lower MSE values indicate better prediction accuracy.

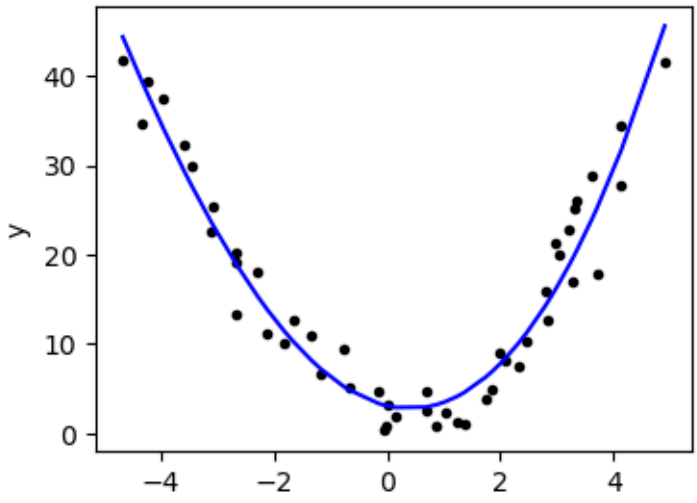
$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

we saw this before! This is the residual sum of squares divided by the number of data points.

`10.003220594982494`

Higher order Polynomials

Why stop at x^2 we have a computer we can go crazy!!!!



Higher order Polynomials

Questions;

1. Rerun the code above for higher and higher order polynomials, do the results get better?
2. Do you reach a point where making the model more complicated doesn't actually improve the results that much?

Higher order Polynomials

Occams Razor -

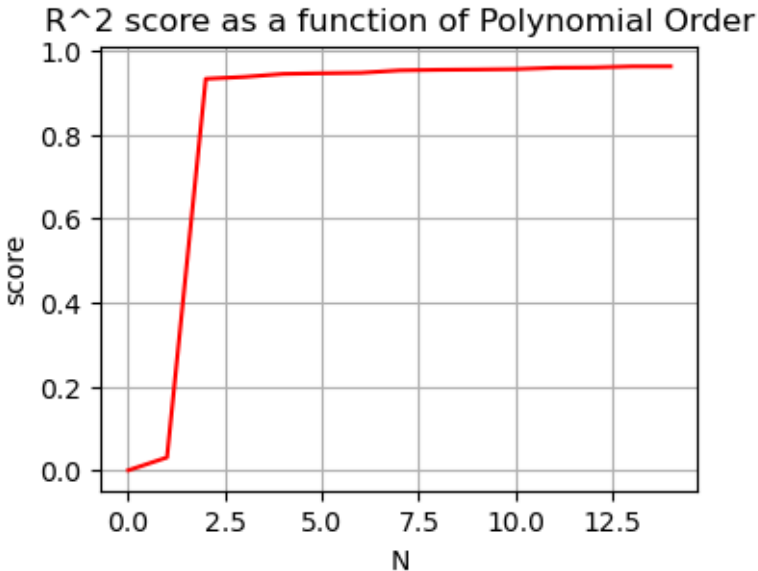
from Wikipedia:
In philosophy, Occam's razor (also spelled Ockham's razor or Ocham's razor; Latin: novacula Occami) is the problem-solving principle that recommends searching for explanations constructed with the smallest possible set of elements. It is also known as the principle of parsimony or the law of parsimony (Latin: lex parsimoniae). Attributed to William of Ockham, a 14th-century English philosopher and theologian, it is frequently cited as Entia non sunt multiplicanda praeter necessitatem, which translates as "Entities must not be multiplied beyond necessity", although Occam never used these exact words. Popularly, the principle is sometimes paraphrased as "of two competing theories, the simpler explanation of an entity is to be preferred."

Higher order Polynomials

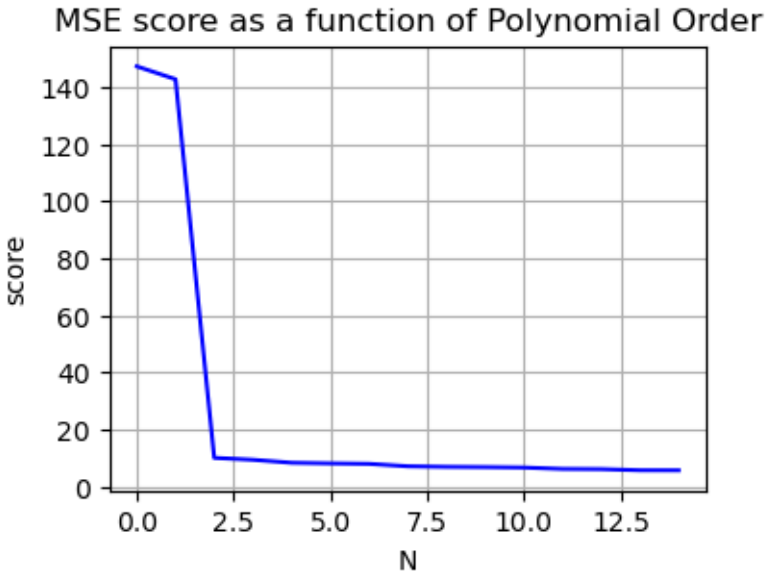
Occams Razor -

Our goal in modeling should be to increase the R^2 value to be as close as possible to 1 and decrease the mean squared error to as close as possible to zero, without making our model ridiculously complicated. Stop before you get diminishing returns!

Higher order Polynomials



Higher order Polynomials



What do we have so far?

- We can apply the ideas of linear regression to higher order polynomials.
- Higher order will always give you lower MSE and R^2 closer to 1.
- BUT we should not complicate our models more then necessary!

You Try

You should start working on Homework 2. Parts 1-4 have linear and polynomial regression applied to vaccine data.