



Data X

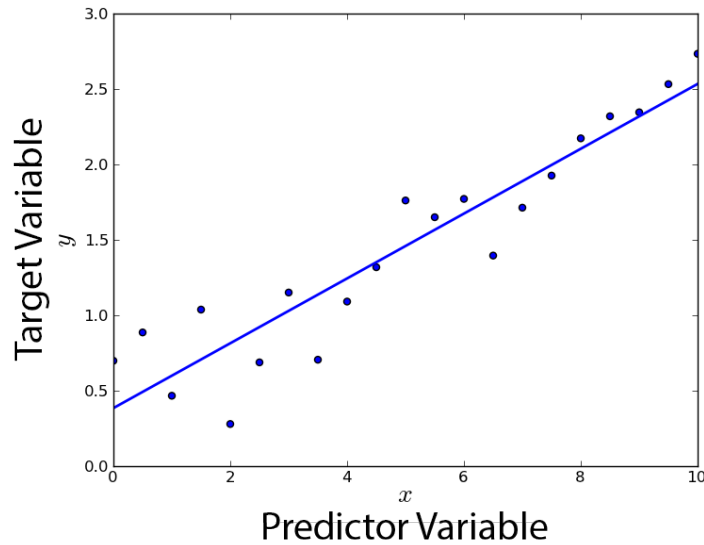
Prediction Data, Signals, and Systems

Ikhtlaq Sidhu
Chief Scientist & Founding Director,
Sutardja Center for Entrepreneurship & Technology
IEOR Emerging Area Professor Award, UC Berkeley

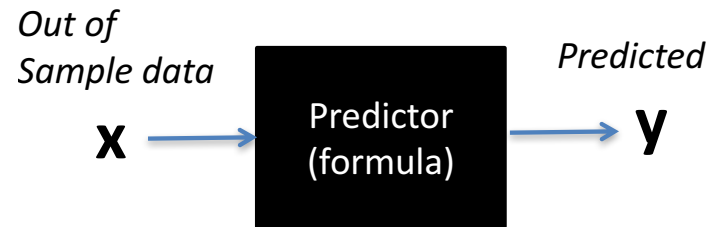
Introduction to Prediction

Data^X

Prediction



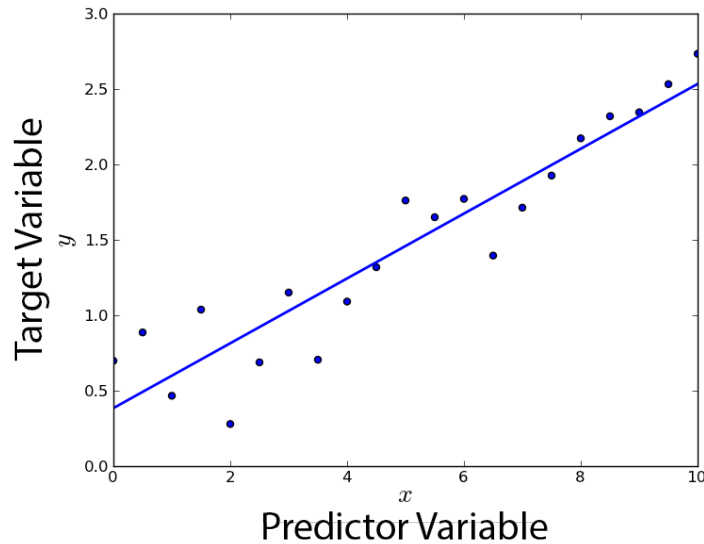
Data We Might Have
(In Sample)



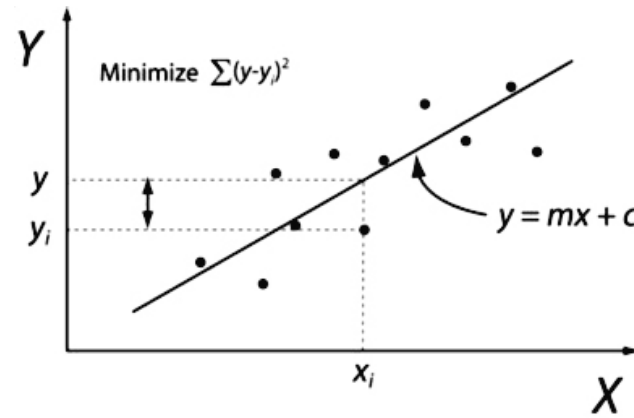
Our Goal: Working with
out of sample data

Data x

Prediction



Data We Might Have
(In Sample)



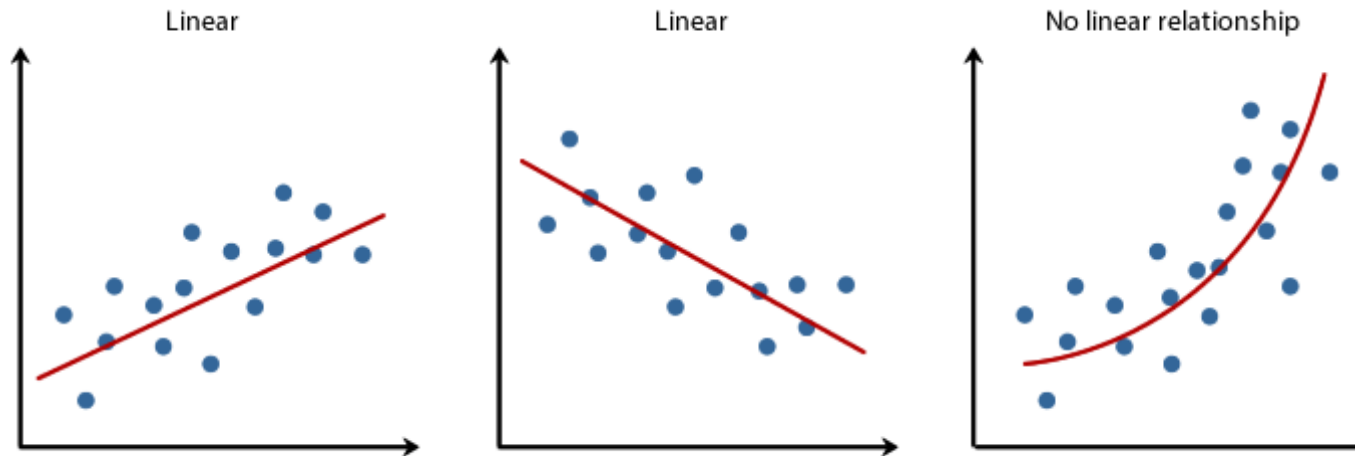
One way to make a prediction:

Choose a line that best fits the sample data

Then $y(x) = mx + c$ is a predictor for a
new out of sample x

Data x

*Of Course, we can not assume that all data can be predicted by a **linear model***



- Model might be a **poor fit** (wrong model)
- Model might be too good of a fit or **over-fit** (only works well on the in sample data)

Image: Laerd Statistics, 2014



Best Linear Predictor (if you just have 2 Variables)

This turns out to be
the **best linear predictor**:

$$L(Y | X) = \mathbb{E}(Y) + \frac{\text{cov}(X, Y)}{\text{var}(X)} [X - \mathbb{E}(X)]$$

It's a line:

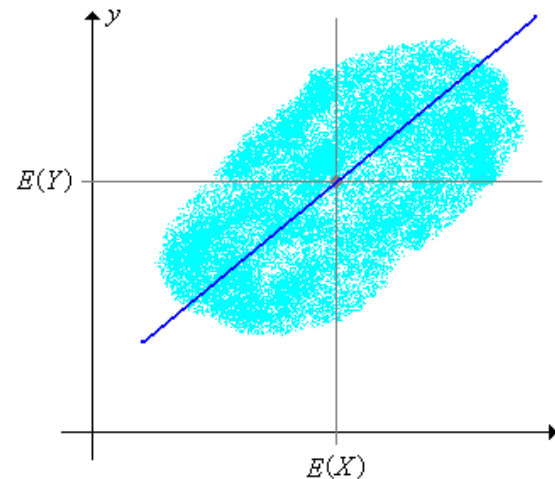
Runs though point: $(\mathbb{E}[X], \mathbb{E}[Y])$

slope: $m = \frac{\text{COV}(X,Y)}{\text{VAR}(X)}$

y-intercept $= \mathbb{E}[Y] - m\mathbb{E}[X]$

$$= \mathbb{E}[Y] - \text{COV}(X, Y) * \frac{\mathbb{E}[X]}{\text{VAR}(X)}$$

*What makes this the the best linear predictor?
How much error does this predictor have?*



The distribution regression line

Remember:

$$\text{COV}(X, Y) = \mathbb{E}[(X - \mu_x)(Y - \mu_y)]$$

$$\text{COV}(X, Y) = \mathbb{E}[XY] - \mathbb{E}[X]\mathbb{E}[Y]$$

$$\text{COV}(X, X) = \text{VAR}(X)$$

$$\text{COV}(AX, Y) = A\text{COV}(X, Y)$$

Data^x

Simple Example: Calculate best linear predictor

Data Set in a Table, two variables

X	Y
2	10
4	5
3	9
5	4
6	3

A decorative banner at the bottom of the slide featuring a background of blue and white binary code (0s and 1s). The text "Data X" is overlaid on the left side in a white, serif font.

Data X

$$E[X] = 4, \quad E[Y] = 6.2$$

$$\begin{aligned} Cov(X, Y) &= E[XY] - E[X]E[Y] \\ &= 21 - 4 * 6.2 = -3.8 \end{aligned}$$

$$Var(X) = 18 - 16 = 2$$

$$\begin{aligned} y\text{-int} &= E[Y] - \frac{Cov(X, Y)}{Var(X)} * E[X] \\ &= 6.2 - \left(\frac{-3.8}{2} * 4 \right) = 13.8 \end{aligned}$$

$$m = \frac{Cov(X, Y)}{Var(X)} = \frac{-3.8}{2} = -1.9$$

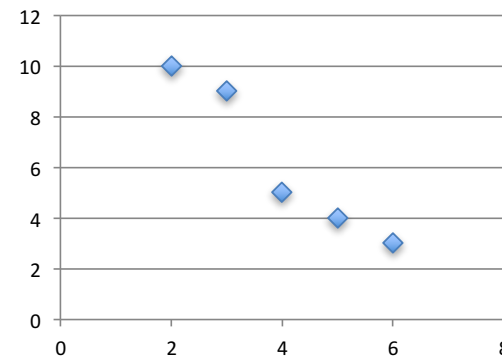
$$y(x) = -1.9x + 13.8$$

$$E[y(x) - y_{actual}]^2 = ?$$

Example: Data Set in a Table 2 variables

X	Y	X*Y	X^2	y(x)
2	10	20	4	6.96
4	5	20	16	3.92
3	9	27	9	5.44
5	4	20	25	2.4
6	3	18	36	0.88

E[X]	E[Y]	E[XY]	E[X^2]
4	6.2	21	18



Data^X

Code Sample

```
import numpy as np

x = np.array([2, 4, 3, 5, 6])
y = np.array([10, 5, 9, 4, 3])

E_x = np.mean(x)
E_y = np.mean(y)

cov_xy = np.mean(x*y) - E_x*E_y

y_0 = E_y - cov_xy/np.var(x)*E_x
m = cov_xy/np.var(x)

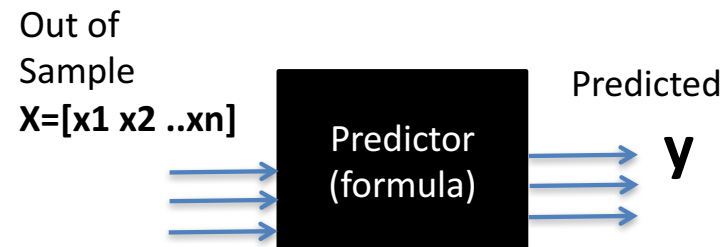
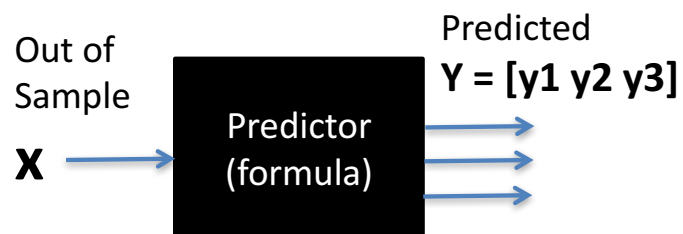
y_pred=m*x+y_0

print "E[(y_pred-y_actual)^2] =", np.mean(np.square(y_pred-y))

E[(y_pred-y_actual)^2] = 0.54
```

DataX

Prediction: Multiple Inputs and Outputs



No problem,
we use multiple predictors.

$$y_1 = g_1(x)$$

$$y_2 = g_2(x)$$

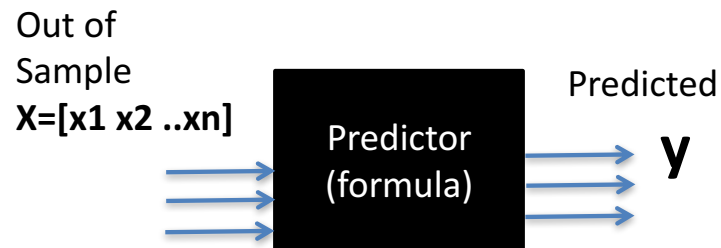
$$y_3 = g_3(x)$$

?

Data \mathbf{X}

A decorative horizontal bar at the bottom of the slide featuring a background of binary code (0s and 1s) in blue and white.

Prediction: Multiple Inputs and Outputs



In this case, for linear prediction, we use a matrix format:

$$\begin{pmatrix} X \\ (N \times d) \\ 1 \ 3 \ 4 \ 6 \ 2 \end{pmatrix} \cdot \begin{pmatrix} W \\ (d \times 1) \end{pmatrix} = \begin{pmatrix} Y \\ (N \times 1) \end{pmatrix}$$

$$\begin{aligned}
 x_{1,1}w_1 + x_{1,2}w_2 + x_{1,3}w_3 &= y_1 \\
 x_{2,1}w_1 + x_{2,2}w_2 + x_{2,3}w_3 &= y_2 \\
 \dots & \\
 \dots &
 \end{aligned}$$

X_i

X is the in-sample data. We only need to figure out W .

With W , we can estimate y for new x , i.e. out of sample data



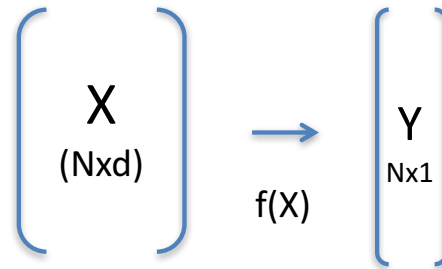
An ML Framework

In Sample Data

which we have

- d features
- N samples

Use for training



Results in Y

(which we know for N samples)

Sometimes measured
Y includes error or noise

$$Y = f(X) + e()$$

We don't know:

$P(X)$, the distribution of X

Function f:
 $f: X \rightarrow Y$

Features: (d columns)

Age, income, zip code, ..

	Sex	Age	Marital...	Occupation	Job Time	Checking	Savings	Good/Bad Mark
Person 1	Female	27.17	Married	Semi-professional	0	No	Yes	Good
Person 2	Male	25.92	Married	Blue Collar	0.375	No	Yes	Good
Person 3	Male	23.08	Married	Blue Collar	1	No	Yes	Good
	Male	39.59	Married	Semi-professional	0	No	Yes	Good
	Male	38.59	Single	Blue Collar	0.125	No	No	Good
	Male	17.25	Married	Blue Collar	0.04	No	No	Good
	Female	17.67	Single	Semi-professional	0	No	No	Bad
	Male	16.5	Married	Blue Collar	0.165	No	No	Good
	Female	27.33	Married	Semi-professional	0	No	No	Good
	Male	31.25	Married	Semi-professional	0	No	Yes	Good
	Male	20	Married	Blue Collar	0	No	No	Bad
	Male	39.5	Married	Blue Collar	0	No	No	Good
	Male	36.5	Married	Blue Collar	3.5	No	No	Good
	Male	52.42	Married	Blue Collar	3.75	No	No	Good

Copyright © Plug&Score

In Sample

Out of Sample

N rows:

- each row has customer information
- d features
- **Y has a value of interest**
 - *Credit score*: a number
 - *Classification*: "good customer" vs "poor customer"

Data X

We use this to train:
 $(x_1, y_1), (x_2, y_2) \dots x_N, y_N$

An ML Framework

X

Algorithm:
New $g(x)$
like $[X] [w]$

Y

$g: x \rightarrow y$,
predictor or classifier

H = Hypothesis Set
All Possible Algorithms Candidate Formulas

Error measure: **Error(g, x)**

for example: **$E[f(x) - g(x)]$**

Actual

Estimated

or any loss function: **$L(y, g(x))$**

- $g(x)$ is our estimate of $f(x)$. Note $f(x) = y$
- Error_in_sample(g) is expected to be similar to Error_out of sample(g)
- Many ways to measure error (difference, square, etc)
- Loss function is like an error function but more general

Data X

Example: Prediction with Regression

Data:		$x(i,1)$	$x(i,2)$	$x(i,3)$	$y(i,1)$
ID	Name	Age	years w employer	Income	Credit Score
1	John	25	3	50	660
2	Alice	23	2	60	580
3	Bill	28	1	80	425
4	Rahul	25	.5	59	320

$$\begin{matrix} \mathbf{X} \\ (N \times d+1) \end{matrix} \cdot \begin{matrix} \mathbf{W} \\ \begin{matrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{matrix} \end{matrix} = \begin{matrix} \mathbf{Y} \\ N \times 1 \end{matrix}$$

$$\begin{pmatrix} 1 & 25 & 3 & 50 \\ 1 & 23 & 2 & 60 \\ 1 & 28 & 1 & 80 \\ 1 & 25 & .5 & 59 \end{pmatrix} \cdot \begin{pmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{pmatrix} = \begin{pmatrix} 660 \\ 580 \\ 425 \\ 320 \end{pmatrix}$$

We are now going to
choose a W that gives us a predictor
 for Y

This time, Y is the actual value we want to estimate

Notice: we added an extra column of 1s?

Data^X

Example: Prediction with Regression

$$\begin{matrix} \mathbf{X} \\ (N \times d+1) \end{matrix} \quad \begin{matrix} \mathbf{W} \\ \begin{matrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{matrix} \end{matrix} = \begin{matrix} \mathbf{Y} \\ N \times 1 \end{matrix}$$

x_2 $\begin{pmatrix} 1 & 25 & 3 & 50 \\ 1 & 23 & 2 & 60 \\ 1 & 28 & 1 & 80 \\ 1 & 25 & .5 & 59 \end{pmatrix} \cdot$

This time, \mathbf{Y} is the actual value we want to estimate.?

Train with this to “calculate” \mathbf{W}

Out of sample
(a row of numbers)

x_i

Predictor:
 $g(x) = XW$

Estimated y
(a real number)

Then predict (or classify) with \mathbf{W}

Data \mathbf{X}

The Math: Linear Regression

Predictor:
 $g(x) = XW$

OK, but better to
measure squared error

$$E_{in}(w) = E[Xw - Y]$$

$$E_{in}(w) = \frac{1}{N} \sum_{i=1}^N (w^T x_i - y_i)^2$$

Data^X

$$E_{in}(w) = \frac{1}{N} \|Xw - y\|^2$$

$$\nabla E_{in}(w) = \frac{2}{N} X^T (Xw - y) = 0.$$

$$X^T X w = (X^T X)^{-1} Y$$

$$X^T X W = X^T Y$$

$$W = (X^T X)^{-1} X^T Y$$

$$Y_{\text{estimated}} = X_{\text{out of sample}} W$$

$$X^T = dxN$$

$$X = N \times d$$

$$(X^T X) =$$

$$dx \times d$$

$$(X^T X)^{-1} =$$

$$dx \times d$$

$$W = (X^T X)^{-1} X^T Y =$$

$$=$$

$$dx \times d$$

$$\cdot$$

$$dx \times N$$

$$\cdot$$

$$Y = N \times m \text{ outputs}$$

$$= [dx \times N] \times [N \times m] =$$

$$dx \times m$$

$$Y_{\text{estimated}} = X W =$$

$$X_{\text{out is}} = N_{\text{out}} \times d$$

$$\cdot$$

$$W = dx \times m$$

$$=$$

$$N_{\text{out}} \times m$$

Data^X

Example 2: Prediction with Regression

$$\begin{array}{c} \mathbf{X} \\ (N \times d+1) \end{array} \quad \begin{array}{c} \mathbf{W} \\ \begin{matrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{matrix} \end{array} = \begin{array}{c} \mathbf{Y} \\ N \times 1 \end{array}$$

$$\mathbf{X} = \begin{pmatrix} 1 & 25 & 3 & 50 \\ 1 & 23 & 2 & 60 \\ 1 & 28 & 1 & 80 \\ 1 & 25 & .5 & 59 \end{pmatrix} \cdot \begin{pmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{pmatrix} = \begin{pmatrix} 660 \\ 580 \\ 425 \\ 320 \end{pmatrix}$$

This time, \mathbf{Y} is the actual value we want to estimate.?

Train with this to “calculate” \mathbf{W}

$$\mathbf{W} = \mathbf{X}^T \mathbf{Y} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{Y}$$

Out of sample
(a row of numbers)

\mathbf{X}_i

Predictor:
 $g(x) = \mathbf{XW}$

Estimated y
(a real number)

Then predict (or classify) with \mathbf{W}

Data \mathbf{X}

Code Sample

```
import numpy as np

x = np.array([
    [1,25,3,50],
    [1,23,2,60],
    [1,28,1,80],
    [1,25,0.5,59]
])
y = np.array([660,580,425,320])

print "W = ", np.linalg.inv(x.T.dot(x)).dot(x.T).dot(y)
```

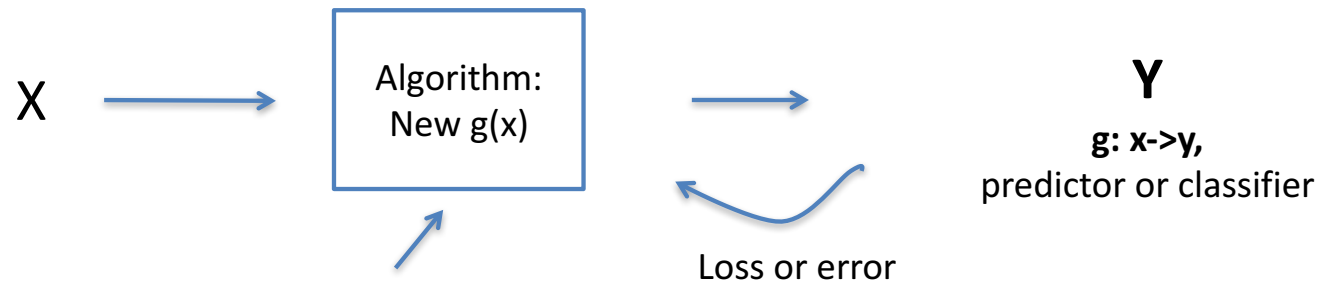
```
W = [ 426.17283951 -16.04938272 149.44444444  3.7345679 ]
```

$$x_0w_0 + x_1w_1 + x_2w_2 + x_3w_3 = y_i$$

Data:		x(i,1)	x(i,2)	x(i,3)	y(i,1)
ID	Name	Age	years w employer	Income	Credit Score
1	John	25	3	50	660
2	Alice	23	2	60	580
3	Bill	28	1	80	425
4	Rahul	25	.5	59	320

Data^x

In the **ML framework**, there is no limit to the predictors or classifiers that can be used.



g can be chosen from,

Linear estimators:

- Any weighted sum
- The best fit line or plane

Non-linear functions:

- Neural Networks
- MLE
- Any function

- We try different functions g until
- $g(x)$ is close to $f(x)$
- For any out of sample x , we can predict Y or classify it

Data^x

End of Section

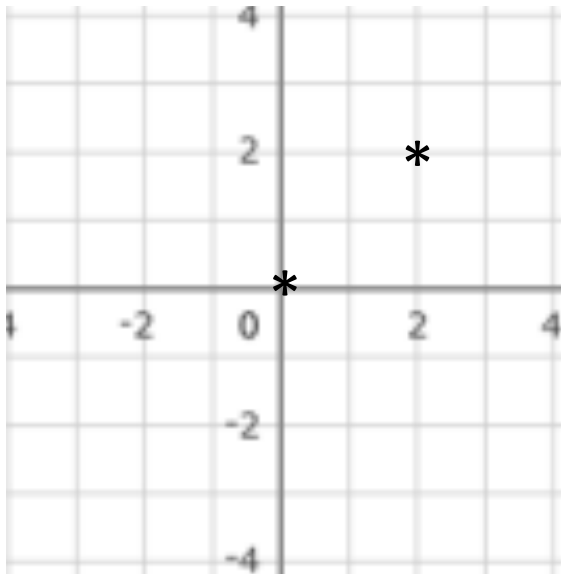


Test Your Understanding

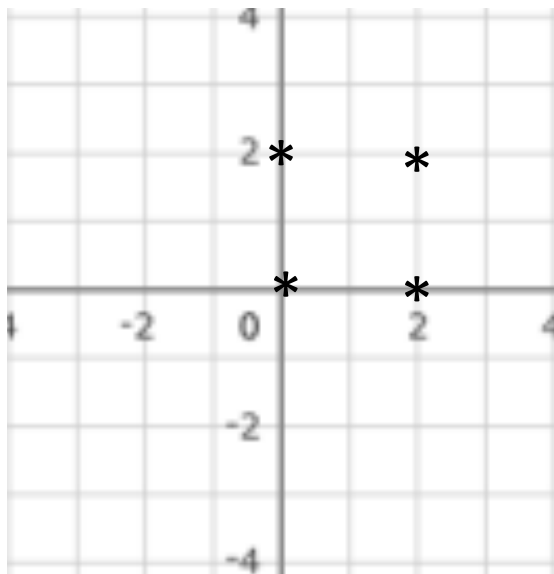


Covariance

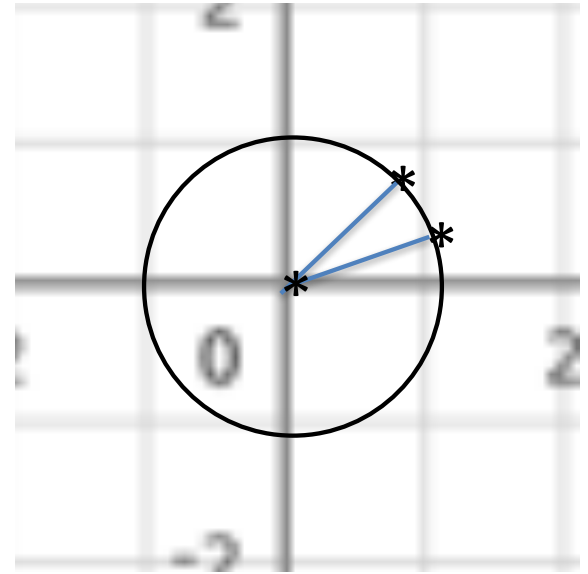
- What is the COV (X,Y) for these points:



(0,0) and (2,2)



(0,0), (2,2), (2,0), (0,2)



- Points on a unit circle that are
- 30 degrees or
 - 45 degrees
 - Every point on the circle

Data^x

Multiple inputs and outputs

$$\text{If } Y_{\text{predicted}} = f(X, W) = X W$$

And W is a 4×3 matrix, then how many inputs (n) are in X and how many outputs (m) are in Y ?

