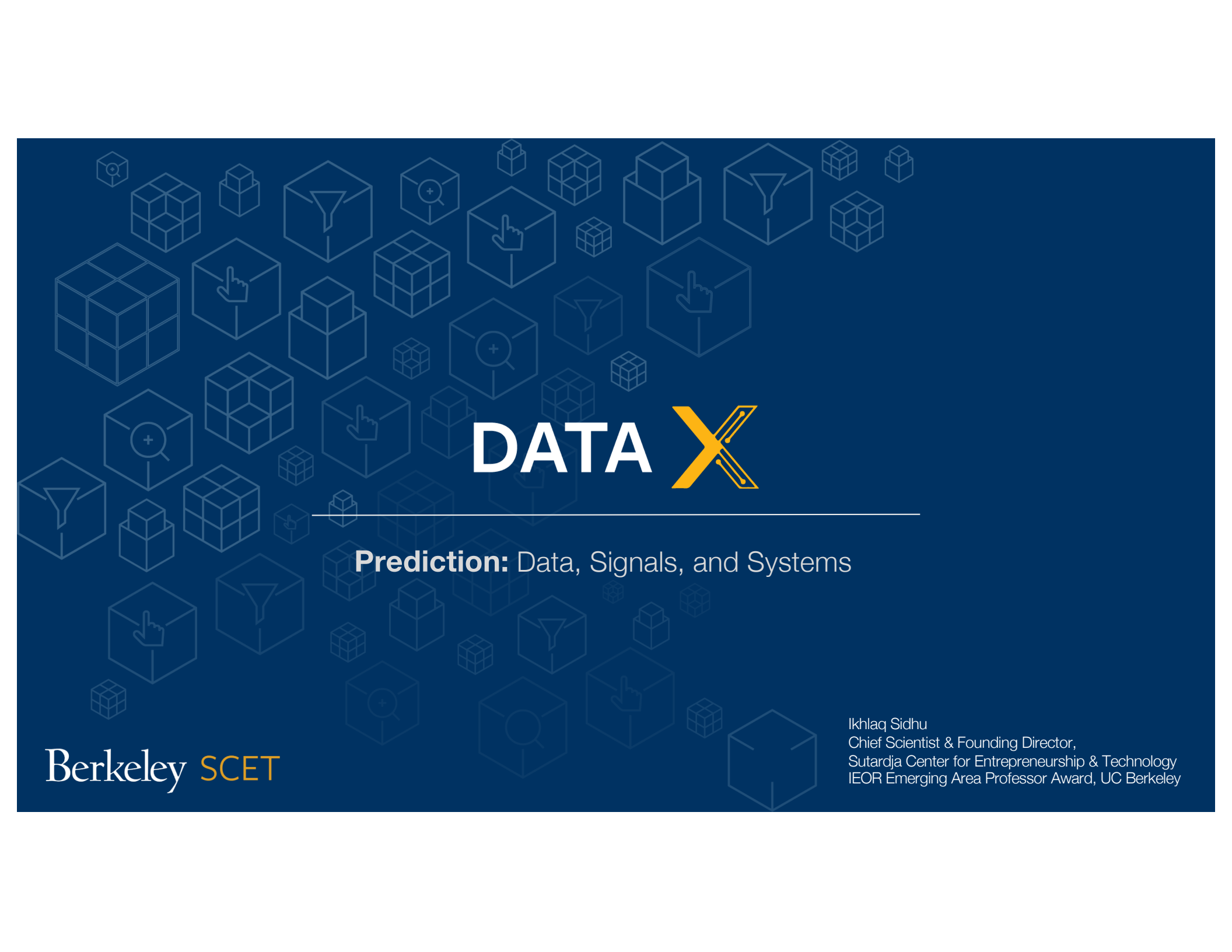




DATA X

Prediction: Data, Signals, and Systems

Berkeley SCET

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



Introduction to Prediction

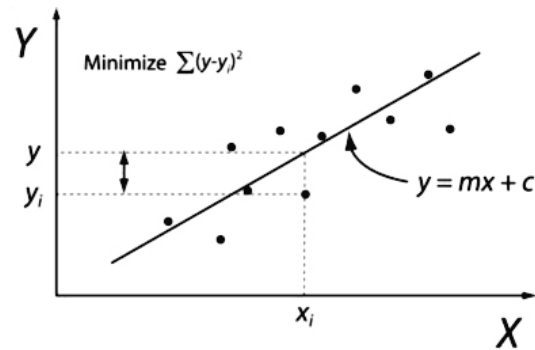
Berkeley SCET

Prediction

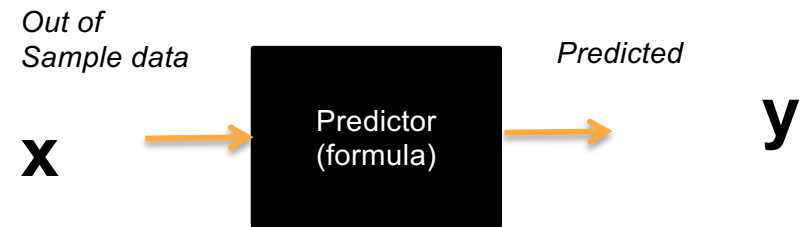
Data We Might Have
(In Sample)

X	Y
2	3
5	9
6	11
8	?
10	?
?	?

Data View



Math View



Our Goal: Working with
out of sample data

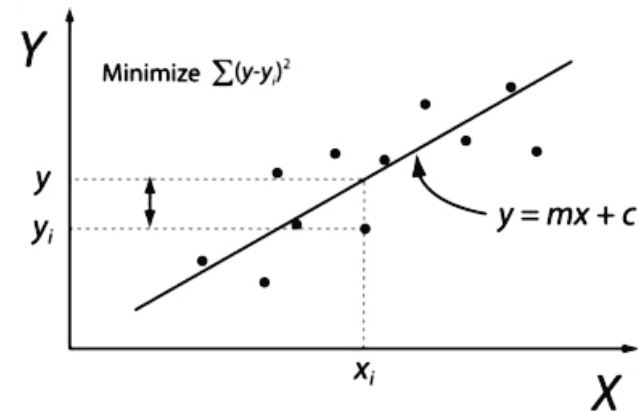
Systems View

Prediction

Data We Might Have
(In Sample)

X	Y
2	3
5	9
6	?
8	?
?	?

Data View

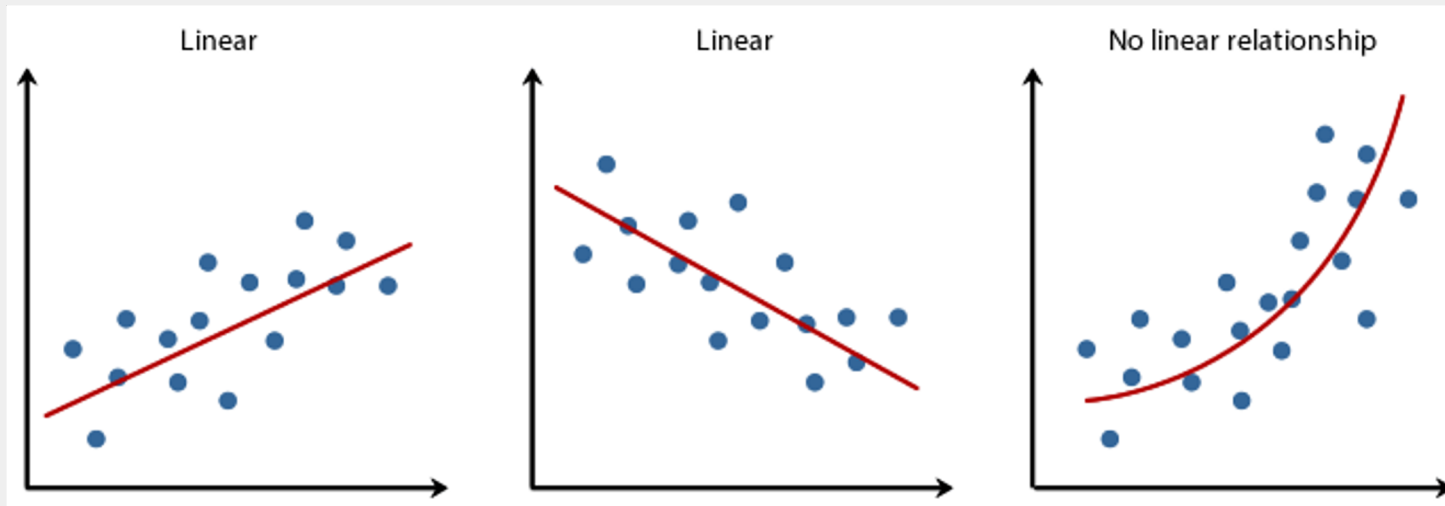


One way to make a prediction:

Choose a line that best fits the sample data

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

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)

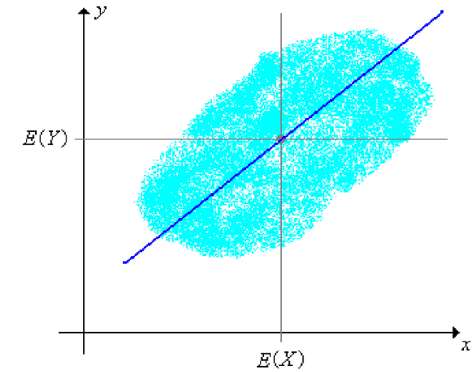
Best Linear Predictor

(if you just have 2 variables)

This turns out to be the best linear predictor:

Berkeley SCET

$$L(Y|X) = E(Y) + \frac{\text{Cov}(X,Y)}{\text{Var}(X)} [X - E(X)]$$



The distribution regression line

It's a line:

runs through point: $(E[X], E[Y])$

slope: $m = \frac{\text{Cov}(X,Y)}{\text{Var}(X)}$

y-intercept $= E[Y] - mE[X]$
 $= E[Y] - \text{Cov}(X, Y) * \frac{E[X]}{\text{Var}(X)}$

Remember:

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

$$\text{Cov}(X,Y) = E[X Y] - E[X] E[Y]$$

$$\text{Cov}(X,X) = \text{Var}(X)$$

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

What makes this the the best linear predictor?

How much error does this predictor have?

Simple Example:

Calculate best
linear predictor

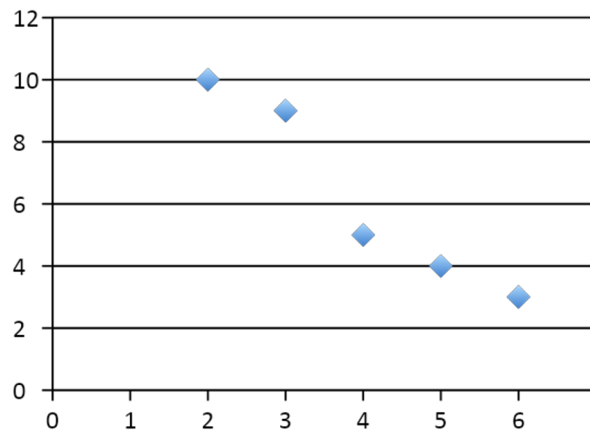
Data Set in a Table, two
variables

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

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	



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

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

$$\text{Var}(X) = 18 - 16 = 2$$

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

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

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

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

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)

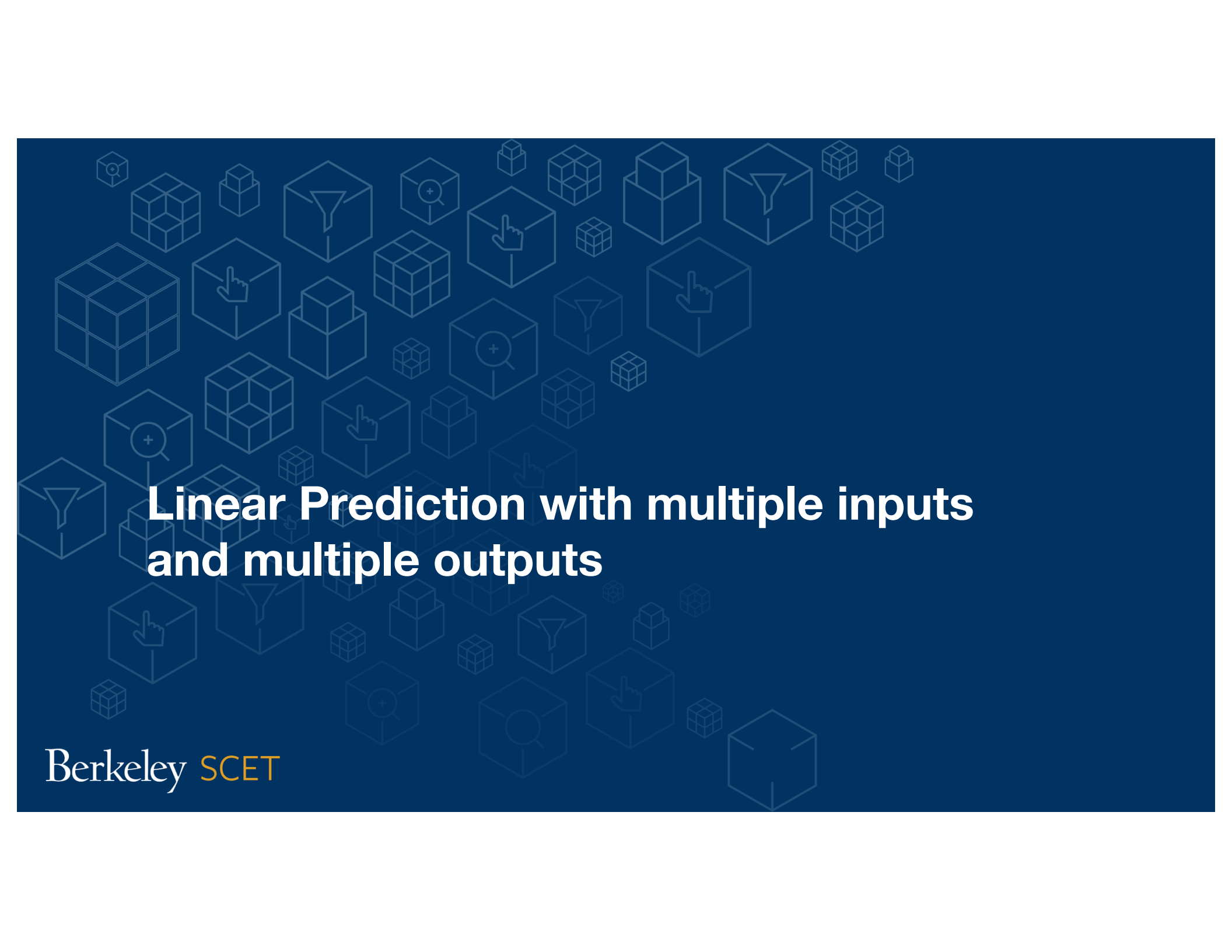
cov_xy = np.mean(x*y) - E_x*E_y

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
```

The background is a dark blue field filled with a repeating pattern of white line-art icons. These icons include 3D cubes of various sizes, some of which are stacked. Other icons include a hand cursor pointing at a cube, a cube with a magnifying glass and a plus sign, and a cube with a funnel shape. The pattern is dense and covers the entire slide area.

Linear Prediction with multiple inputs and multiple outputs

Prediction:

Multiple Inputs and Outputs



No problem, we use multiple predictors.

$$\mathbf{y1} = g1(\mathbf{x}) \quad \mathbf{y2} = g2(\mathbf{x}) \quad \mathbf{y3} = g3(\mathbf{x})$$



What can be done in this case?

Prediction:

Multiple Inputs and Outputs

Out of Sample

$\mathbf{X} = [\mathbf{x}_1 \ \mathbf{x}_2 \ \dots \mathbf{x}_n]$



Predictor
(formula)



Predicted
 $\mathbf{y}$

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

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

$\mathbf{X}_i$

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

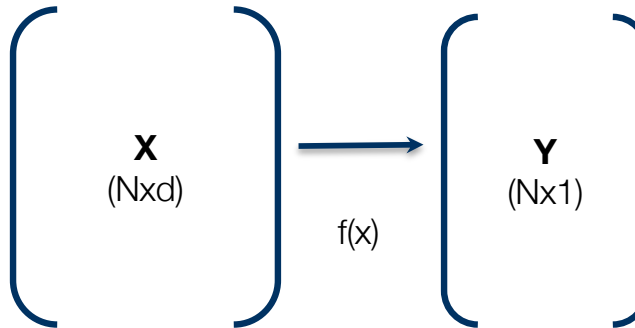
$\mathbf{X}$ is the in-sample data. We only need to figure out $\mathbf{W}$.

With $\mathbf{W}$, we can estimate $\mathbf{y}$ for new $\mathbf{x}$, i.e. out of sample data

An ML Framework

We don't know:
 $P(X)$, the distribution of X
 Function f :
 $f: X \rightarrow Y$

Berkeley SCET



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
Person N	male	39.58	Married	Semi-professional	0	No	Yes	Good
	male	30.58	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	0.5	No	No	Bad
	male	39.5	Married	Blue Collar	3.5	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

Out of Sample

In Sample Data:

which we have

- d features
- N samples

Use for training

Results in Y (which we know for N examples)

Sometimes measured Y includes error or noise

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

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”

Example: Prediction with Regression

Data: ID	Name	$x(i,1)$ Age	$x(i,2)$ years w employer	$x(i,3)$ Income	$y(i,1)$ 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

$$\begin{pmatrix} \mathbf{X} \\ (N \times d+1) \\ 1 \ 25 \ 3 \ 50 \\ 1 \ 23 \ 2 \ 60 \\ 1 \ 28 \ 1 \ 80 \\ 1 \ 25 \ .5 \ 59 \end{pmatrix} \cdot \begin{pmatrix} \mathbf{W} \\ w_0 \\ w_1 \\ w_2 \\ w_3 \end{pmatrix} = \begin{pmatrix} \mathbf{Y} \\ N \times 1 \\ 660 \\ 580 \\ 425 \\ 320 \end{pmatrix}$$

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?

Example: Prediction with Regression

$$\mathbf{X}_2 \begin{pmatrix} \mathbf{x} \\ (N \times d+1) \\ 1 \ 25 \ 3 \ 50 \\ 1 \ 23 \ 2 \ 60 \\ 1 \ 28 \ 1 \ 80 \\ 1 \ 25 \ .5 \ 59 \end{pmatrix} \cdot \begin{pmatrix} \mathbf{w} \\ w_0 \\ w_1 \\ w_2 \\ w_3 \end{pmatrix} = \begin{pmatrix} \mathbf{Y} \\ N \times 1 \\ 660 \\ 580 \\ 425 \\ 320 \end{pmatrix}$$

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

Train with this to “calculate” **W**



Then predict (or classify) with **W**

The Math:

Linear Regression

Berkeley SCET

Predictor:
 $g(x) = XW$

OK, but better
to measure
squared error

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

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

The Math:

Linear Regression

Berkeley SCET

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

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

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

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

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

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

N = # of X data rows
d = # of features
m = # of outputs in Y

$$X^T = dxN$$

$$X = N \times d$$

$$(X^T X) =$$

$$dx \times d$$

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

$$dx \times d$$

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

$$=$$

$$dx \times d$$

$$\cdot dx \times N$$

$$\cdot$$

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

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

$$dx \times m$$

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

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

$$\cdot$$

$$W = dx \times m$$

$$=$$

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

Example:

Prediction with
Regression
(to be
continued)

$$X = \begin{pmatrix} \mathbf{X} \\ (N \times d+1) \\ 1 \ 25 \ 3 \ 50 \\ 1 \ 23 \ 2 \ 60 \\ 1 \ 28 \ 1 \ 80 \\ 1 \ 25 \ .5 \ 59 \end{pmatrix} \cdot \begin{pmatrix} \mathbf{W} \\ w_0 \\ w_1 \\ w_2 \\ w_3 \end{pmatrix} = \begin{pmatrix} \mathbf{Y} \\ N \times 1 \\ 660 \\ 580 \\ 425 \\ 320 \end{pmatrix}$$

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

Train with this to “calculate”

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

W

Out of
Sample data
(a row of numbers)
X_i



Predictor:
g(x) = XW



Estimated y
(a real number)

Then predict (or classify) with **W**

Code Example

```
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])

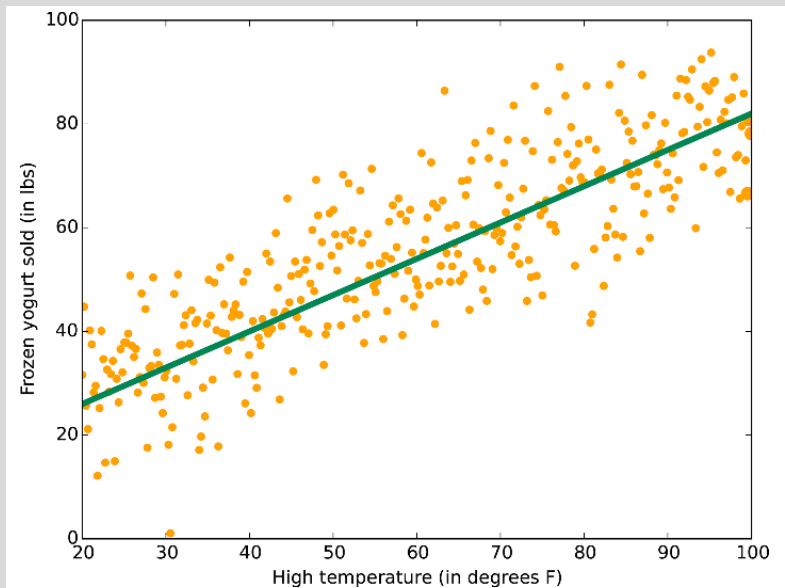
print "W = ", np.linalg.inv(x.T.dot(x)).dot(x.T).dot(y)

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

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

Data: ID	Name		x(i,1) Age	x(i,2) years w employer	x(i,3) Income		y(i,1) Credit Score
1	John	1	25	3	50	W0	660
2	Alice	1	23	2	60	W1	580
3	Bill	1	28	1	80	W2	425
4	Rahul	1	25	5	59	W3	320

Linear Regression Illustration with Scikit Learn



```
#Setting Linear Regression in sklearn  
from sklearn import linear_model
```

```
model= linear_model.LinearRegression()  
model.fit(X_train, Y_train)
```

```
Y_pred_train = model.predict(X_train)  
Y_pred_test = model.predict(X_test)
```

```
# Compare Y_pred_test with Y_test for  
error.
```

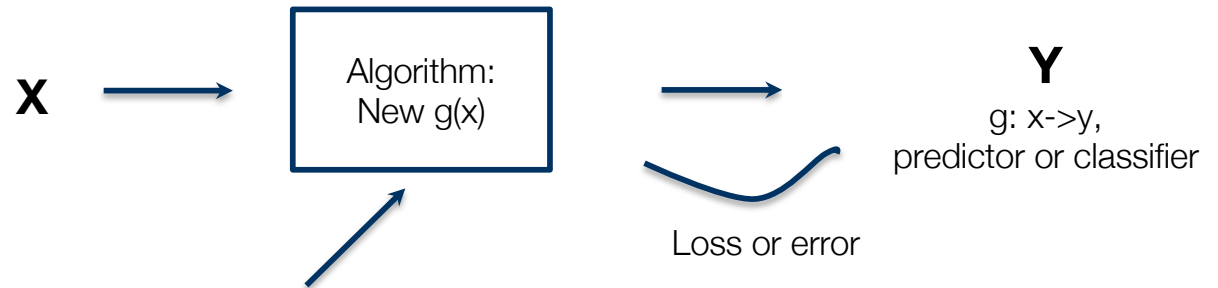
Illustration Source:

<https://docs.microsoft.com/en-us/azure/machine-learning/machine-learning-algorithm-choice>

An ML Framework

Berkeley SCET

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 (or different parameters)

- $g(x)$ is close to $f(x)$

- For any out of sample x , we can predict Y or classify it



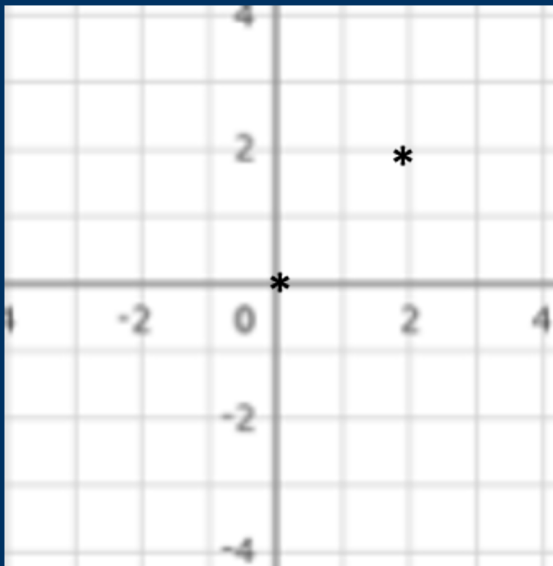
End of Section



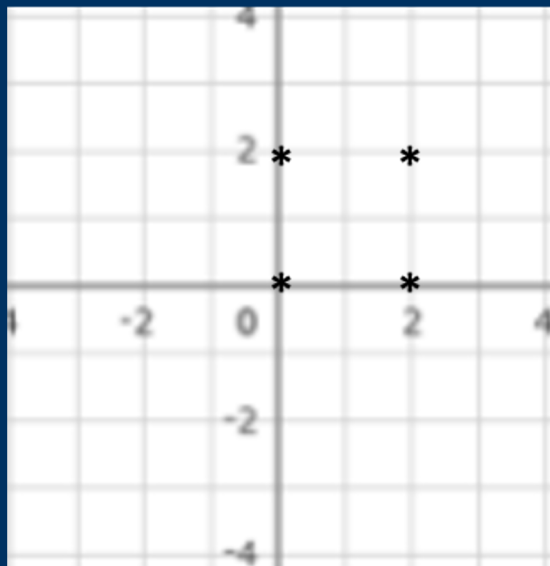
Test your understanding

Covariance

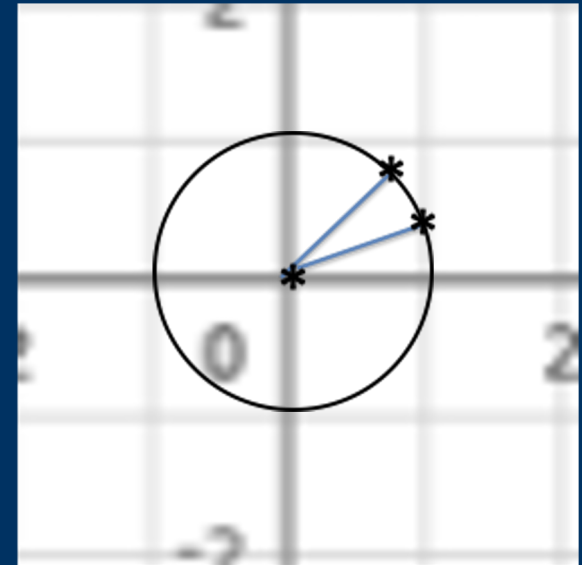
- What is the COV (X,Y) for these points:



(0,0) and (2,2)



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



Points on a unit circle that are

- 30 degrees and 45 degrees, and the origin
- Or, Every point on the circle

Multiple inputs and outputs

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

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

