



Great Expectations

Chris Piech (and Dickens)
CS109, Stanford University

Joint Random Variables



Use a joint table, density function or CDF to solve probability question



Think about **conditional** probabilities with joint variables (which might be continuous)



Use and find **expectation** of multiple RVS



Use and find **independence** of multiple RVS



What happens when you **add** random variables?



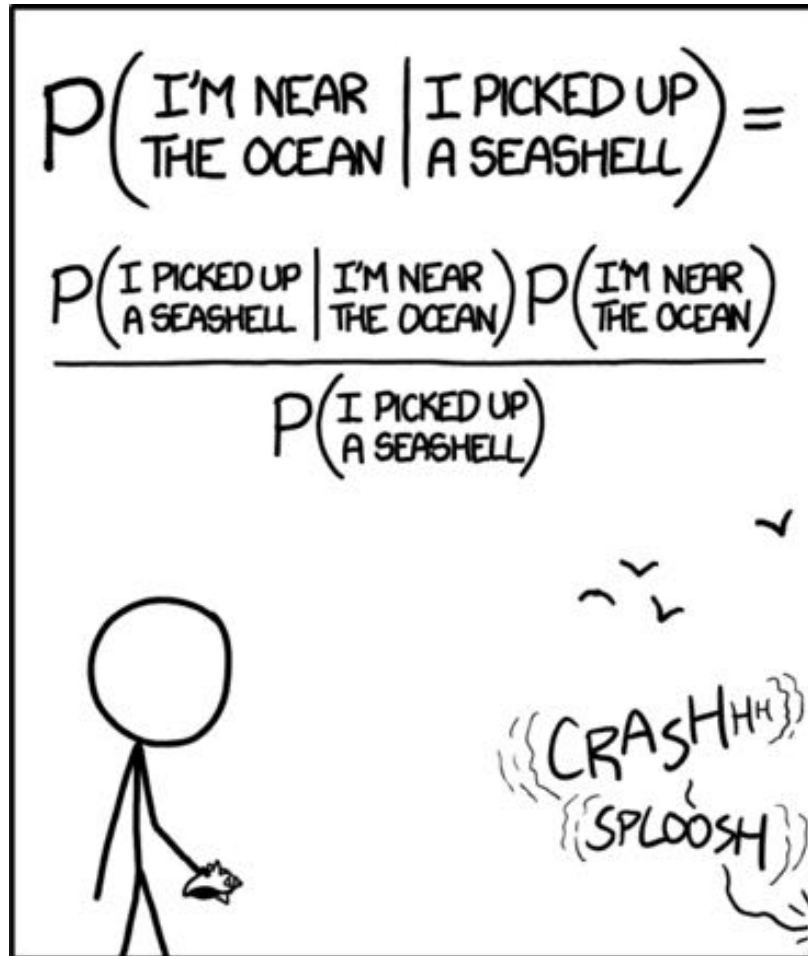
How do multiple variables **covary**?

Course Mean

$E[\text{CS 109}]$

*This is actual midpoint of course
(Just wanted you to know)*

Sea side



$$P\left(\begin{array}{l} \text{I'M NEAR} \\ \text{THE OCEAN} \end{array} \middle| \begin{array}{l} \text{I PICKED UP} \\ \text{A SEASHELL} \end{array}\right) =$$

$$\frac{P\left(\begin{array}{l} \text{I PICKED UP} \\ \text{A SEASHELL} \end{array} \middle| \begin{array}{l} \text{I'M NEAR} \\ \text{THE OCEAN} \end{array}\right) P\left(\begin{array}{l} \text{I'M NEAR} \\ \text{THE OCEAN} \end{array}\right)}{P\left(\begin{array}{l} \text{I PICKED UP} \\ \text{A SEASHELL} \end{array}\right)}$$

$$P\left(\begin{array}{l} \text{I PICKED UP} \\ \text{A SEASHELL} \end{array}\right)$$

STATISTICALLY SPEAKING, IF YOU PICK UP A SEASHELL AND DON'T HOLD IT TO YOUR EAR, YOU CAN PROBABLY HEAR THE OCEAN.

Review

Expected Values of Sums

$$E[X + Y] = E[X] + E[Y]$$

Generalized:
$$E\left[\sum_{i=1}^n X_i\right] = \sum_{i=1}^n E[X_i]$$

Holds regardless of dependency between X_i 's

End Review



Boole Was Cool

- Let E_1, E_2, \dots, E_n be events with indicator RVs X_i
 - If event E_i occurs, then $X_i = 1$, else $X_i = 0$
 - Recall $E[X_i] = P(E_i)$
 - Why?

$$E[X_i] = 0 \cdot (1 - P(E_i)) + 1 \cdot P(E_i)$$

Bernoulli aka Indicator Random Variables were studied extensively by George Boole

Boole died of being too cool



Expectation of Binomial

- Let $Y \sim \text{Bin}(n, p)$
 - n independent trials
 - Let $X_i = 1$ if i -th trial is “success”, 0 otherwise
 - $X_i \sim \text{Ber}(p)$ $E[X_i] = p$

$$Y = X_1 + X_2 + \cdots + X_n = \sum_{i=1}^n X_i$$

$$E[Y] = E\left[\sum_{i=1}^n X_i\right]$$

$$= \sum_{i=1}^n E[X_i]$$

$$= E[X_1] + E[X_2] + \cdots + E[X_n]$$

$$= np$$

Expectation of Negative Binomial

- Let $Y \sim \text{NegBin}(r, p)$
 - Recall Y is number of trials until r “successes”
 - Let $X_i = \#$ of trials to get success after $(i - 1)$ st success
 - $X_i \sim \text{Geo}(p)$ (i.e., Geometric RV) $E[X_i] = \frac{1}{p}$

$$Y = X_1 + X_2 + \cdots + X_r = \sum_{i=1}^r X_i$$

$$E[Y] = E\left[\sum_{i=1}^r X_i\right]$$

$$= \sum_{i=1}^r E[X_i]$$

$$= E[X_1] + E[X_2] + \cdots + E[X_r]$$

$$= \frac{r}{p}$$

Differential Privacy

Aims to provide means to **maximize the accuracy** of probabilistic queries while minimizing the **probability** of identifying its records.



Cynthia Dwork's celebrity lookalike is Cynthia Dwork.

Differential Privacy

100 independent values $X_1 \dots X_{100}$ where $X_i \sim \text{Bern}(p)$

```
# Maximize accuracy, while preserving privacy.
```

```
def calculateYi(Xi):  
    obfuscate = random()  
    if obfuscate:  
        return indicator(random())  
    else:  
        return Xi
```

random() returns
True or False with
equal likelihood

Differential Privacy

100 independent values $X_1 \dots X_{100}$ where $X_i \sim \text{Bern}(p)$

```
# Maximize accuracy, while preserving privacy.
```

```
def calculateYi(Xi):
```

```
    obfuscate = random()
```

```
    if obfuscate:
```

```
        return indicator(random())
```

```
    else:
```

```
        return Xi
```

random() returns
True or False with
equal likelihood

What is $E[Y_i]$?

$$E[Y_i] = P(Y_i = 1) = \frac{p}{2} + \frac{1}{4}$$

Differential Privacy

100 independent values $X_1 \dots X_{100}$ where $X_i \sim \text{Bern}(p)$

```
# Maximize accuracy, while preserving privacy.
```

```
def calculateYi(Xi):
```

```
    obfuscate = random()
```

```
    if obfuscate:
```

```
        return indicator(random())
```

```
    else:
```

```
        return Xi
```

random() returns
True or False with
equal likelihood

Let $Z = \sum_{i=1}^{100} Y_i$

What is the $E[Z]$?

$$E[Z] = E\left[\sum_{i=1}^{100} Y_i\right] = \sum_{i=1}^{100} E[Y_i] = \sum_{i=1}^{100} \left(\frac{p}{2} + \frac{1}{4}\right) = 50p + 25$$

Differential Privacy

100 independent values $X_1 \dots X_{100}$ where $X_i \sim \text{Bern}(p)$

```
# Maximize accuracy, while preserving privacy.
```

```
def calculateYi(Xi):
```

```
    obfuscate = random()
```

```
    if obfuscate:
```

```
        return indicator(random())
```

```
    else:
```

```
        return Xi
```

random() returns
True or False with
equal likelihood

Let $Z = \sum_{i=1}^{100} Y_i$ $E[Z] = 50p + 25$ How do you estimate p ?

$$p \approx \frac{Z - 25}{50}$$

Challenge: What is the probability that our estimate is good?

More Practice!

Computer Cluster Utilization

- Computer cluster with k servers
 - Requests independently go to server i with probability p_i
 - Let event $A_i =$ server i receives no requests
 - Let Bernoulli B_i be an indicator for A_i
 - $X =$ # of events A_1, A_2, \dots, A_k that occur
 - $Y =$ # servers that receive ≥ 1 request $= k - X$
 - $E[Y]$ after first n requests?
 - Since requests independent: $P(A_i) = (1 - p_i)^n$

$$X = \sum_{i=1}^k B_i$$

$$E[X] = E\left[\sum_{i=1}^k B_i\right] = \sum_{i=1}^k E[B_i] = \sum_{i=1}^k P(A_i) = \sum_{i=1}^k (1 - p_i)^n$$

$$E[Y] = k - E[X] = k - \sum_{i=1}^k (1 - p_i)^n$$

amazon

The Amazon logo consists of the word "amazon" in a bold, black, lowercase sans-serif font. Below the text is a curved orange arrow that starts under the letter 'a' and points to the right, ending under the letter 'n'.



* 52% of Amazons Earnings

**More profitable than Amazon's North America commerce operations



When stuck, brainstorm
about random variables





Hash Tables (aka Toy Collecting)

- Consider a hash table with n buckets
 - Each string equally likely to get hashed into any bucket
 - Let $X = \#$ strings to hash until each bucket ≥ 1 string
 - What is $E[X]$?
 - Let $X_i = \#$ of trials to get success after i -th success
 - where “success” is hashing string to previously empty bucket
 - After i buckets have ≥ 1 string, probability of hashing a string to an empty bucket is $p = (n - i) / n$
 - $P(X_i = k) = \frac{n - i}{n} \left(\frac{i}{n} \right)^{k-1}$ equivalently: $X_i \sim \text{Geo}((n - i) / n)$
 - $E[X_i] = 1 / p = n / (n - i)$
 - $X = X_0 + X_1 + \dots + X_{n-1} \Rightarrow E[X] = E[X_0] + E[X_1] + \dots + E[X_{n-1}]$
$$E[X] = \frac{n}{n} + \frac{n}{n-1} + \frac{n}{n-2} + \dots + \frac{n}{1} = n \left[\frac{1}{n} + \frac{1}{n-1} + \dots + 1 \right] = O(n \log n)$$

This is your final answer

Break



Conditional Expectation

Conditional Expectation

- X and Y are jointly discrete random variables

- Recall conditional PMF of X given $Y = y$:

$$p_{X|Y}(x | y) = P(X = x | Y = y) = \frac{p_{X,Y}(x, y)}{p_Y(y)}$$

- Define conditional expectation of X given $Y = y$:

$$E[X | Y = y] = \sum_x x P(X = x | Y = y) = \sum_x x p_{X|Y}(x | y)$$

- Analogously, jointly continuous random variables:

$$f_{X|Y}(x | y) = \frac{f_{X,Y}(x, y)}{f_Y(y)} \quad E[X | Y = y] = \int_{-\infty}^{\infty} x f_{X|Y}(x | y) dx$$

Rolling Dice

- Roll two 6-sided dice D_1 and D_2
 - $X = \text{value of } D_1 + D_2$ $Y = \text{value of } D_2$
 - What is $E[X | Y = 6]$?

$$\begin{aligned} E[X | Y = 6] &= \sum_x xP(X = x | Y = 6) \\ &= \left(\frac{1}{6}\right)(7 + 8 + 9 + 10 + 11 + 12) = \frac{57}{6} = 9.5 \end{aligned}$$

- Intuitively makes sense: $6 + E[\text{value of } D_1] = 6 + 3.5$

Properties of Conditional Expectation

- X and Y are jointly distributed random variables

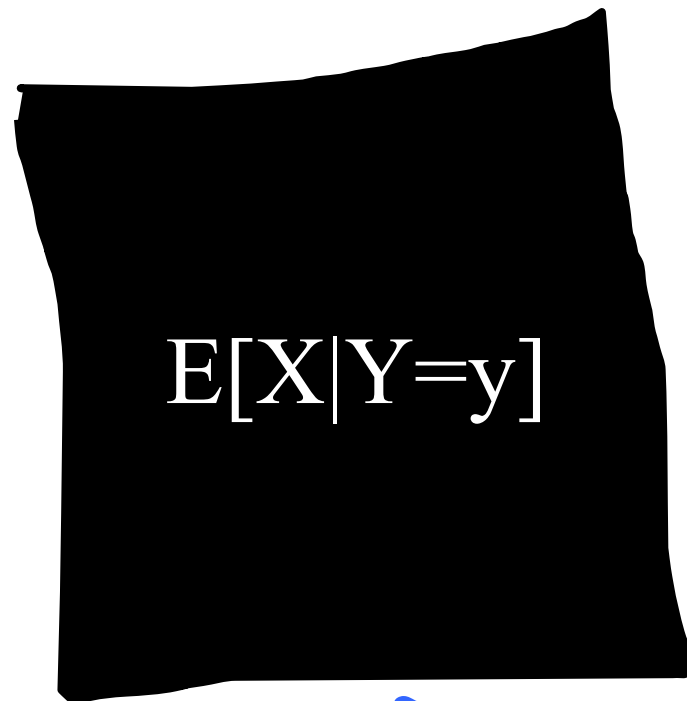
$$E[g(X) | Y = y] = \sum_x g(x) p_{X|Y}(x | y) \quad \text{or} \quad \int_{-\infty}^{\infty} g(x) f_{X|Y}(x | y) dx$$

- Expectation of conditional sum:

$$E\left[\sum_{i=1}^n X_i | Y = y\right] = \sum_{i=1}^n E[X_i | Y = y]$$

Conditional Expectation Functions

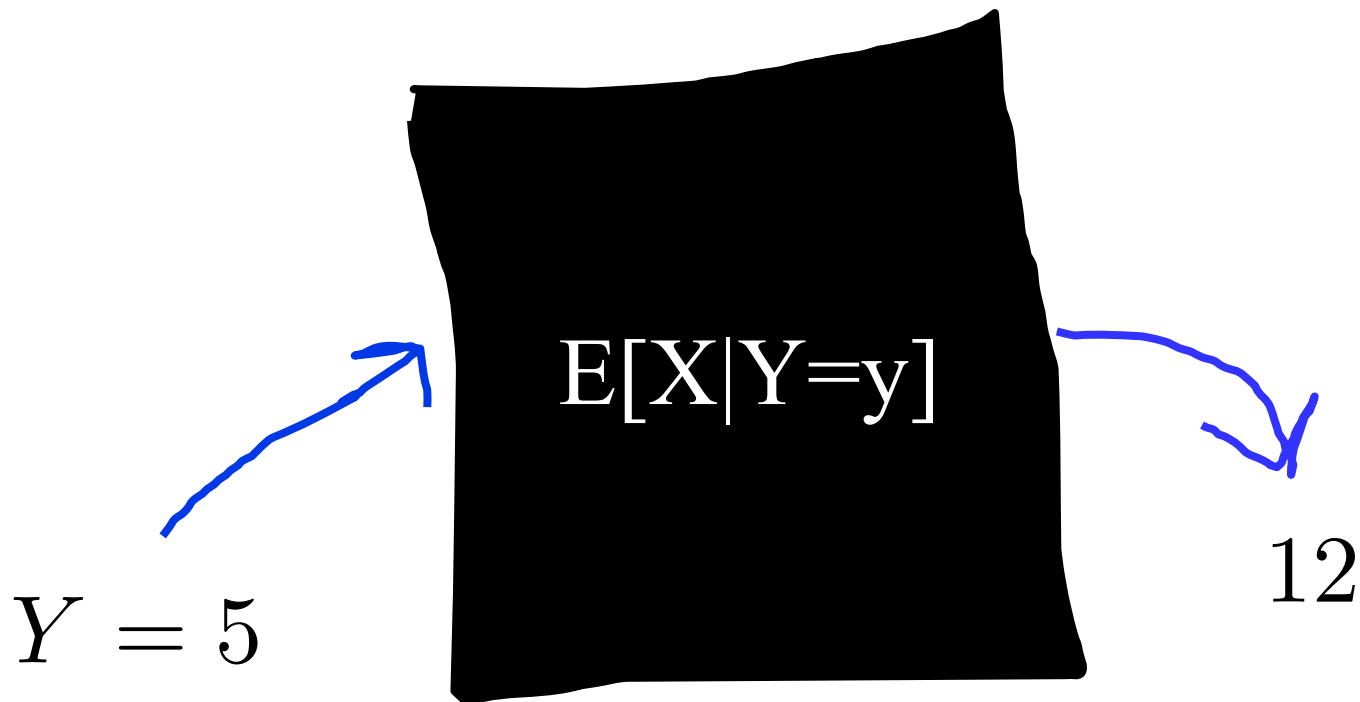
- Define $g(Y) = E[X | Y]$
- This is just function of Y


$$E[X|Y=y]$$

This is a function with Y as input

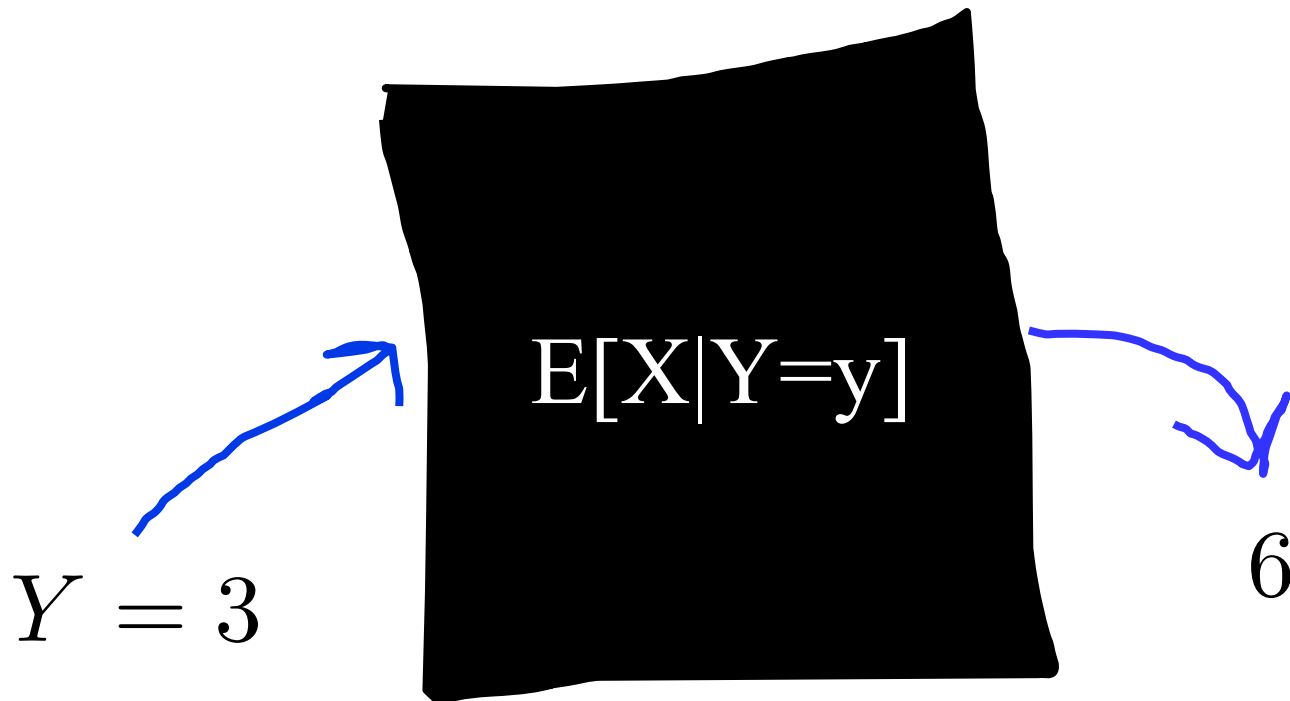
Conditional Expectation Functions

- Define $g(Y) = E[X | Y]$
- This is just function of Y



Conditional Expectation Functions

- Define $g(Y) = E[X | Y]$
- This is just function of Y



Conditional Expectation Functions

This is a number:

$$E[X]$$



This is a function of y :

$$E[X | Y = y]$$

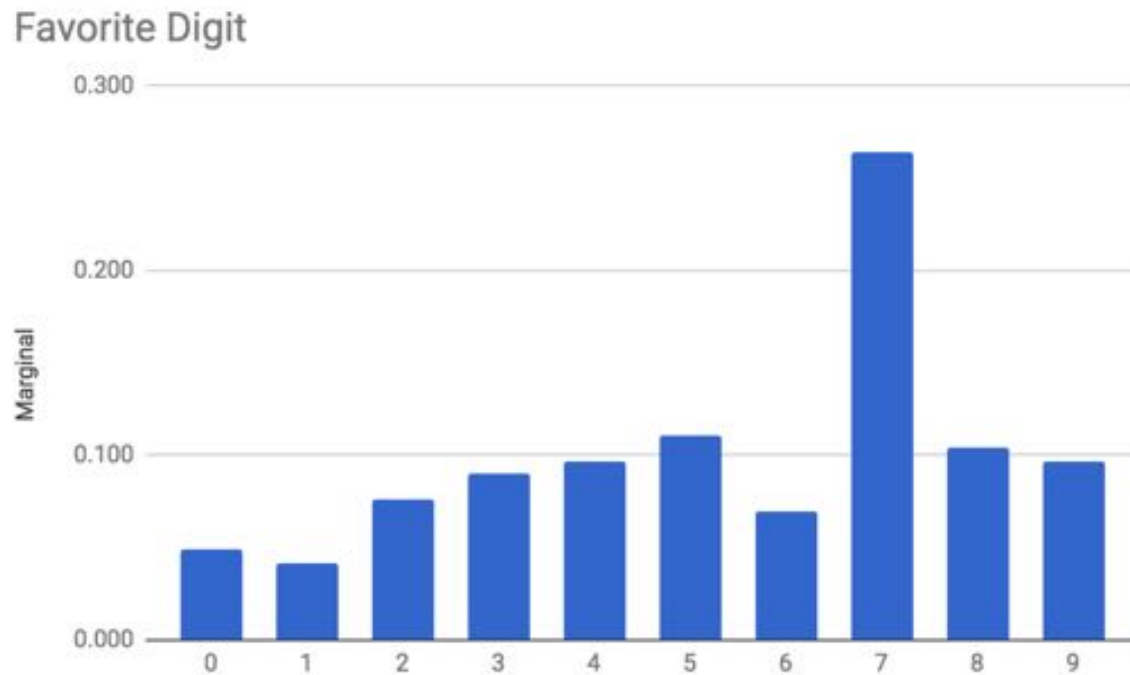
$$E[X = 5]$$

Doesn't make sense. Take expectation of random variables, not events

Conditional Expectation Functions

X = favorite number

Y = year in school



$$E[X] = 0 * 0.05 + \dots + 9 * 0.10 = 5.38$$

Conditional Expectation Functions

X = favorite number

Y = year in school

$E[X | Y]$?

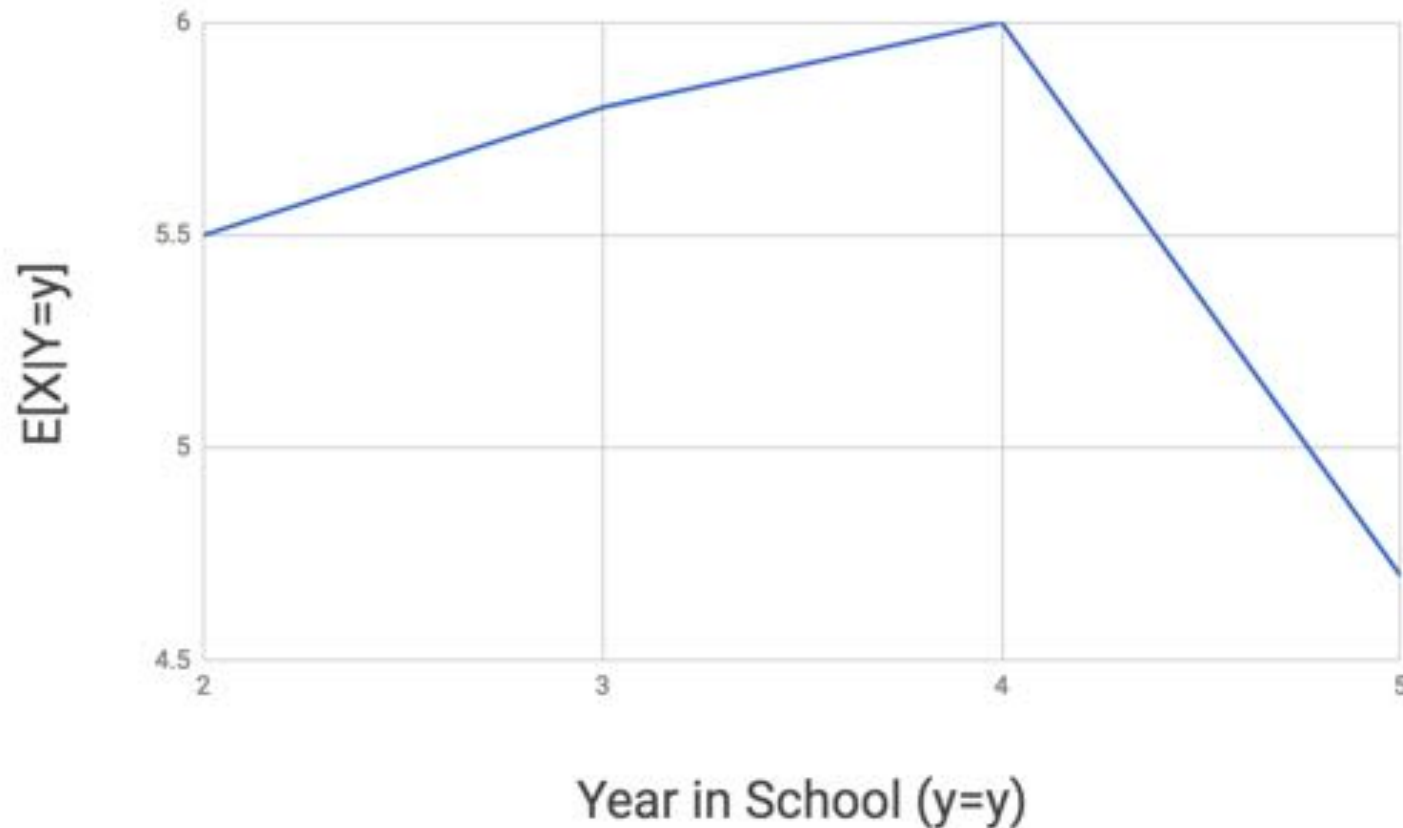
Year in school, $Y = y$	$E[X Y = y]$
2	5.5
3	5.8
4	6.0
5	4.7

Conditional Expectation Functions

X = favorite number

Y = year in school

$E[X | Y]$?

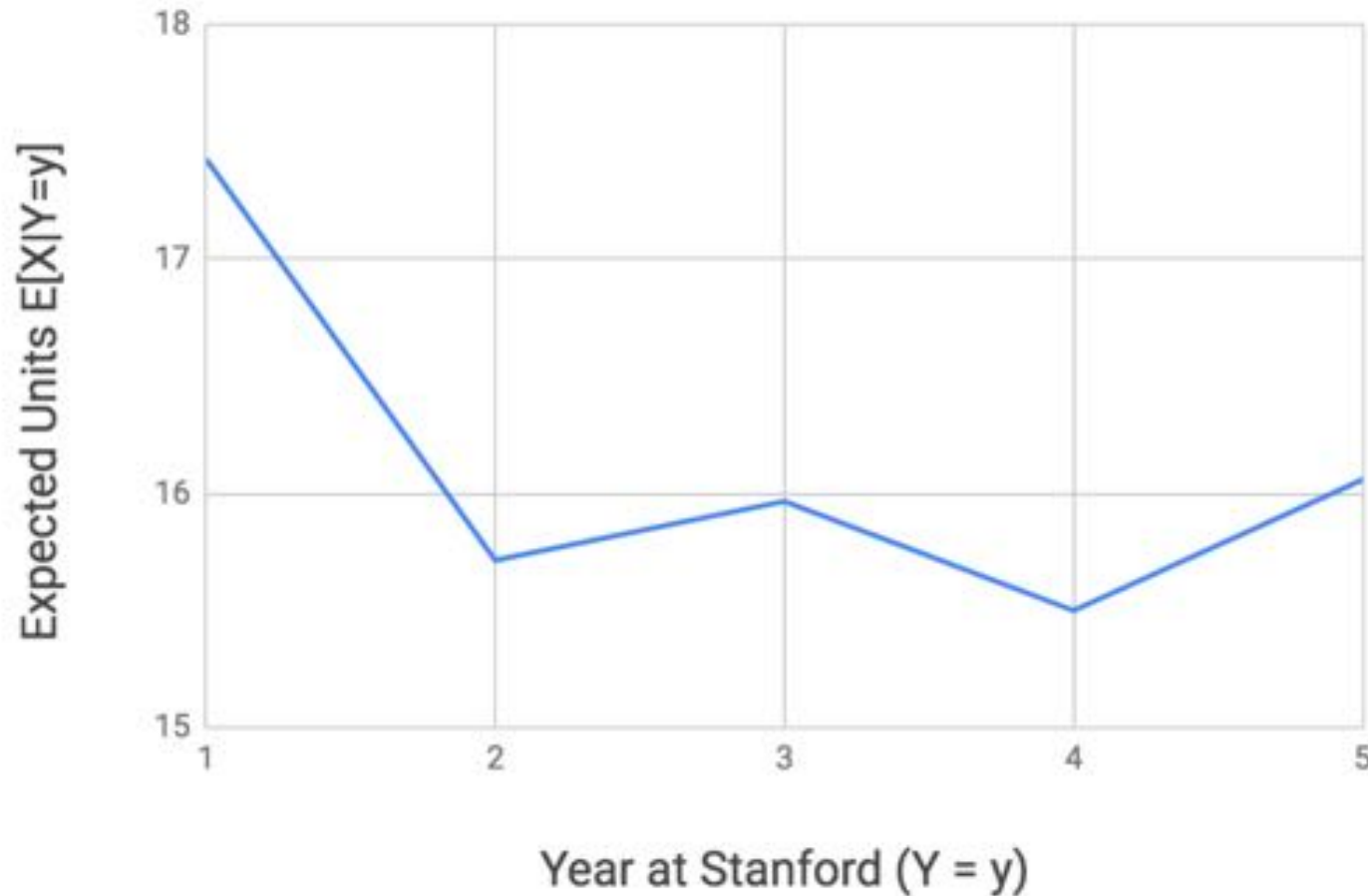


Conditional Expectation Functions

X = units in fall quarter

Y = year in school

$E[X | Y]$?



Law of Total Expectation

$$E[E[X|Y]] = E[X]$$

$$E[E[X|Y]] = \sum_y E[X|Y = y]P(Y = y) \quad g(Y) = E[X|Y]$$

$$= \sum_y \sum_x xP(X = x|Y = y)P(Y = y) \quad \text{Def of } E[X|Y]$$

$$= \sum_y \sum_x xP(X = x, Y = y) \quad \text{Chain rule!}$$

$$= \sum_x \sum_y xP(X = x, Y = y) \quad \text{I switch the order of the sums}$$

$$= \sum_x x \sum_y P(X = x, Y = y) \quad \text{Move that } x \text{ outside the } y \text{ sum}$$

$$= \sum_x xP(X = x) \quad \text{Marginalization}$$

$$= E[X] \quad \text{Def of } E[X]$$

Law of Total Expectation

For any random variable X and any **discrete** random variable Y



$$E[X] = \sum_y E[X|Y = y]P(Y = y)$$

Analyzing Recursive Code

```
int Recurse() {  
    int x = randomInt(1, 3); // Equally likely values  
    if (x == 1) return 3;  
    else if (x == 2) return (5 + Recurse());  
    else return (7 + Recurse());  
}
```

- Let Y = value returned by `Recurse()`. What is $E[Y]$?

$$E[Y] = E[Y | X = 1]P(X = 1) + E[Y | X = 2]P(X = 2) + E[Y | X = 3]P(X = 3)$$

$$E[Y | X = 1] = 3$$

$$E[Y | X = 2] = E[5 + Y] = 5 + E[Y]$$

$$E[Y | X = 3] = E[7 + Y] = 7 + E[Y]$$

$$E[Y] = 3(1/3) + (5 + E[Y])(1/3) + (7 + E[Y])(1/3) = (1/3)(15 + 2E[Y])$$

$$E[Y] = 15$$

Protip: do this in CS161

If we have time...

Hiring and Engineer

```
DEFINE JOBINTERVIEWQUICKSORT(LIST):
  OK SO YOU CHOOSE A PIVOT
  THEN DIVIDE THE LIST IN HALF
  FOR EACH HALF:
    CHECK TO SEE IF IT'S SORTED
    NO, WAVE, IT DOESN'T MATTER
    COMPARE EACH ELEMENT TO THE PIVOT
    THE BIGGER ONES GO IN A NEW LIST
    THE EQUAL ONES GO INTO UH
    THE SECOND LIST FROM BEFORE
  HANG ON, LET ME NAME THE LISTS
  THIS IS LIST A
  THE NEW ONE IS LIST B
  PUT THE BIG ONES INTO LIST B
  NOW TAKE THE SECOND LIST
  CALL IT LIST, UH, A2
  WHICH ONE WAS THE PIVOT IN?
  SCRATCH ALL THAT
  IT JUST RECURSIVELY CALLS ITSELF
  UNTIL BOTH LISTS ARE EMPTY
  RIGHT?
  NOT EMPTY, BUT YOU KNOW WHAT I MEAN
  AM I ALLOWED TO USE THE STANDARD LIBRARIES?
```

Your company has one job opening for a software engineer.

You have n candidates. But you have to say yes/no **immediately** after each interview!

Proposed algorithm: reject the first k and accept the next one who is better than all of them.

What's the best value of k ?

Hiring and Engineer

n candidates, must say yes/no **immediately** after each interview.
Reject the first k , accept the next who is better than all of them.
What's the best value of k ?

B: event that you hire the best engineer

X: position of the best engineer on the interview schedule

What is the $P(B|X = i)$?

Hiring and Engineer

n candidates, must say yes/no **immediately** after each interview.
Reject the first k , accept the next who is better than all of them.
What's the best value of k ?

B: event that you hire the best engineer

X: position of the best engineer on the interview schedule

What is the $P(B|X = i)$?



Hiring and Engineer

n candidates, must say yes/no **immediately** after each interview.
Reject the first k , accept the next who is better than all of them.
What's the best value of k ?

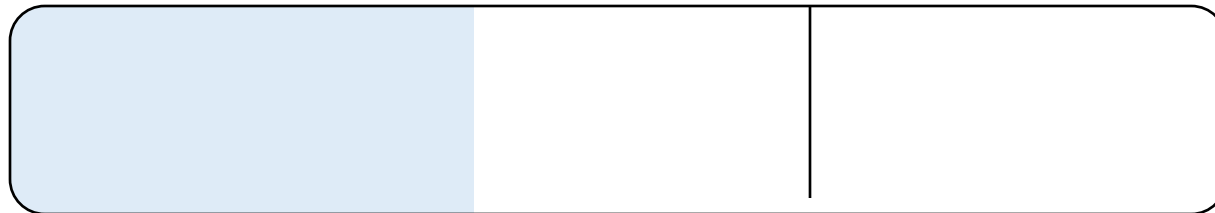
B: event that you hire the best engineer

X: position of the best engineer on the interview schedule

What is the $P(B|X = i)$?

k

i



Hint: where is the
best among the first
 $i - 1$ candidates?



Hiring and Engineer

n candidates, must say yes/no **immediately** after each interview.
Reject the first k , accept the next who is better than all of them.
What's the best value of k ?

B: event that you hire the best engineer

X: position of the best engineer on the interview schedule

What is the $P(B|X = i)$?

Here?



Hint: where is the
best among the first
 $i - 1$ candidates?

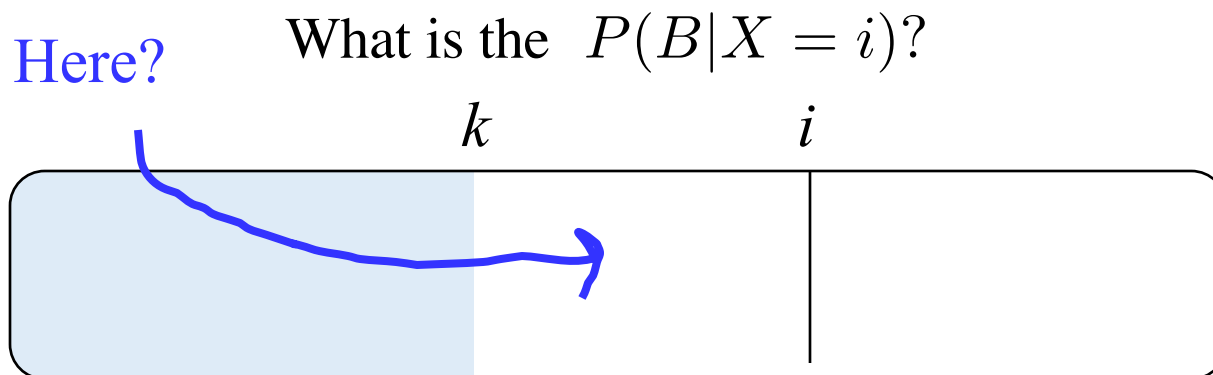


Hiring and Engineer

n candidates, must say yes/no **immediately** after each interview.
Reject the first k , accept the next who is better than all of them.
What's the best value of k ?

B: event that you hire the best engineer

X: position of the best engineer on the interview schedule



Hint: where is the
best among the first
 $i - 1$ candidates?



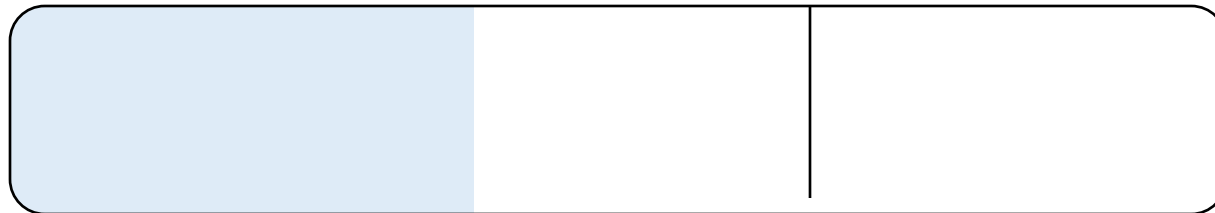
Hiring and Engineer

n candidates, must say yes/no **immediately** after each interview.
Reject the first k , accept the next who is better than all of them.
What's the best value of k ?

B: event that you hire the best engineer

X: position of the best engineer on the interview schedule

$$P(B|X = i) = \frac{k}{i-1} \text{ if } i > k$$



Hint: where is the
best among the first
 $i - 1$ candidates?



Hiring and Engineer

n candidates, must say yes/no **immediately** after each interview.
Reject the first k , accept the next who is better than all of them.
What's the best value of k ?

B: event that you hire the best engineer

X: position of the best engineer on the interview schedule

$$P_k(B) = \sum_{i=1}^n P_k(B|X = i)P(X = i)$$

By the law of total expectation

$$= \frac{1}{n} \sum_{i=1}^n P_k(B|X = i)$$

$$= \frac{1}{n} \sum_{i=k+1}^n \frac{k}{i-1}$$

since we know $P_k(\text{Best}|X = i)$

$$\approx \frac{1}{n} \int_{i=k+1}^n \frac{k}{i-1} di$$

By Riemann Sum approximation

$$= \frac{k}{n} \ln(i-1) \Big|_{k+1}^n = \frac{k}{n} \ln \frac{n-1}{k} \approx \frac{k}{n} \ln \frac{n}{k}$$

Hiring and Engineer

n candidates, must say yes/no **immediately** after each interview.
Reject the first k , accept the next who is better than all of them.
What's the best value of k ?

B: event that you hire the best engineer

X: position of the best engineer on the interview schedule

$$P_k(B) = \sum_{i=1}^n P_k(B|X = i)P(X = i)$$
$$\approx \frac{k}{n} \ln \frac{n}{k}$$

By the law of total expectation

Fun fact. Optimized when: $k = \frac{n}{e}$

That's all folks!

Let's Do Some Sorting!

5	3	7	4	8	6	2	1
---	---	---	---	---	---	---	---

QuickSort

5	3	7	4	8	6	2	1
---	---	---	---	---	---	---	---



select
“pivot”

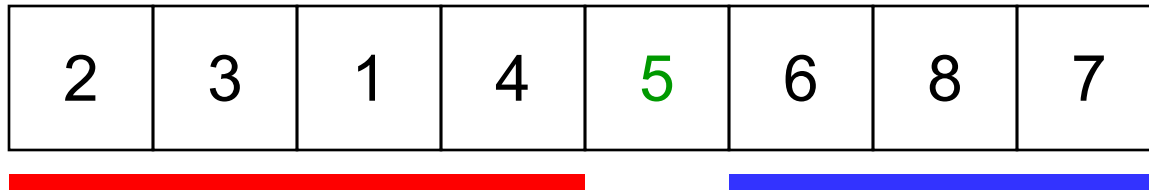
Recursive Insight

5	3	7	4	8	6	2	1
---	---	---	---	---	---	---	---

Partition array so:

- everything smaller than pivot is on left
- everything greater than or equal to pivot is on right
- pivot is in-between

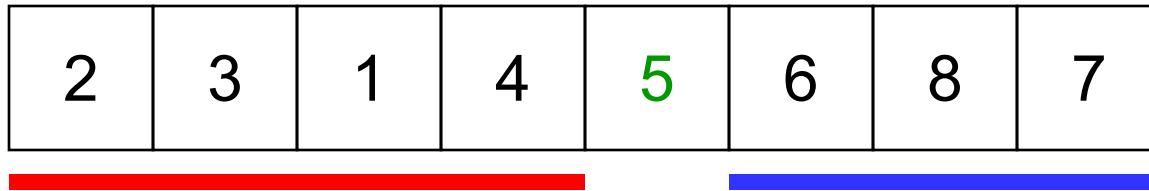
Recursive Insight



Partition array so:

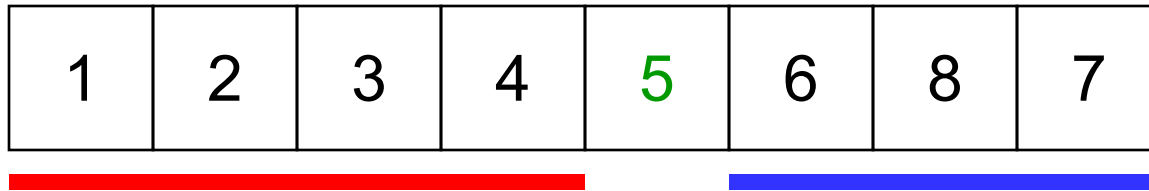
- everything smaller than pivot is on left
- everything greater than or equal to pivot is on right
- pivot is in-between

Recursive Insight



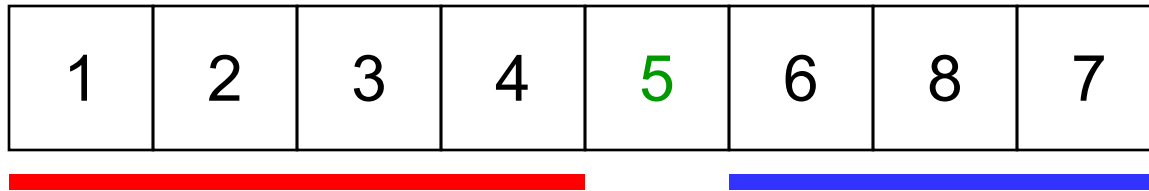
Now recursive sort “red” sub-array

Recursive Insight



Now recursive sort “red” sub-array

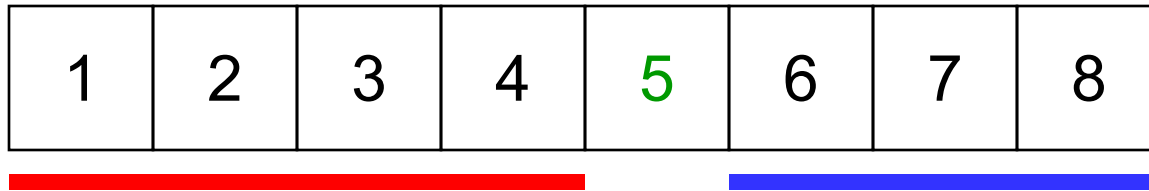
Recursive Insight



Now recursive sort “red” sub-array

Then, recursive sort “blue” sub-array

Recursive Insight



Now recursive sort “red” sub-array

Then, recursive sort “blue” sub-array

Recursive Insight

1	2	3	4	5	6	7	8
---	---	---	---	---	---	---	---

Everything is sorted!

```
void Quicksort(int arr[], int n)
{
    if (n < 2) return;

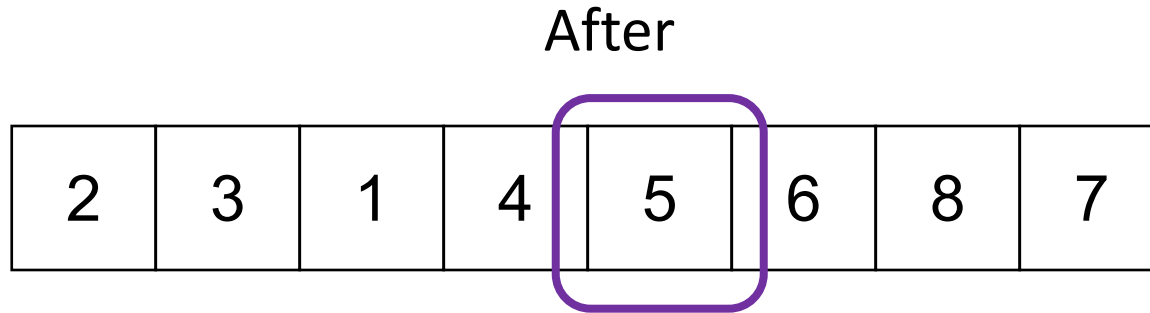
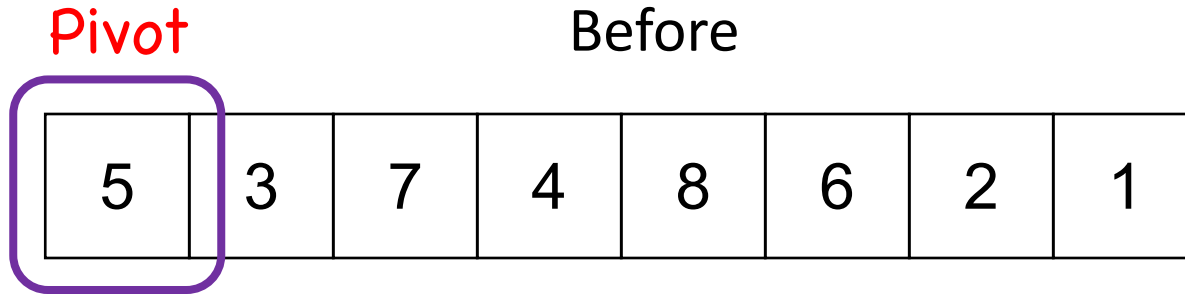
    int boundary = Partition(arr, n);

    // Sort subarray up to pivot
    Quicksort(arr, boundary);

    // Sort subarray after pivot to end
    Quicksort(arr + boundary + 1, n - boundary - 1);
}
```

“boundary” is the index of the pivot

Partition



Does one comparison for every element in the array and the pivot.

Complexity of quicksort is determined by number of comparisons made to pivot

Complexity QuickSort

- QuickSort is $O(n \log n)$, where $n = \#$ elems to sort
 - But in “worst case” it can be $O(n^2)$
 - Worst case occurs when every time pivot is selected, it is maximal or minimal remaining element

Expected Running Time of QuickSort

- Let $X = \#$ comparisons made when sorting n elems
 - $E[X]$ gives us expected running time of algorithm
 - Given V_1, V_2, \dots, V_n in random order to sort
 - Let Y_1, Y_2, \dots, Y_n be V_1, V_2, \dots, V_n in sorted order

When are Y_a and Y_b are compared?

Lets Imagine Our Array in Sorted Order

		Y_a		Y_b	
1	3	5	7	9	11
Y_1	Y_2	Y_3	Y_4	Y_5	Y_6

Whether or not they are compared
depends on pivot choice

Lets Imagine Our Array in Sorted Order

		Y_a		Y_b	
1	3	5	7	9	11

Whether or not they are compared
depends on pivot choice

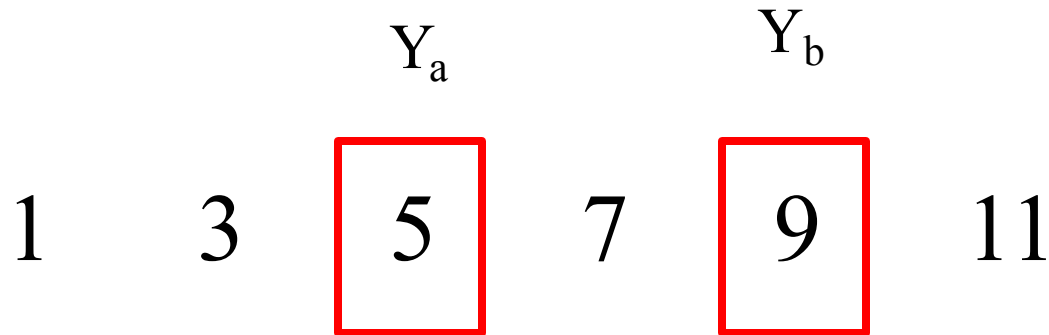
$P(Y_a \text{ and } Y_b \text{ ever compared})$

		Y_a		Y_b	
1	3	5	7	9	11

Consider pivot choice: Y_a

They are compared

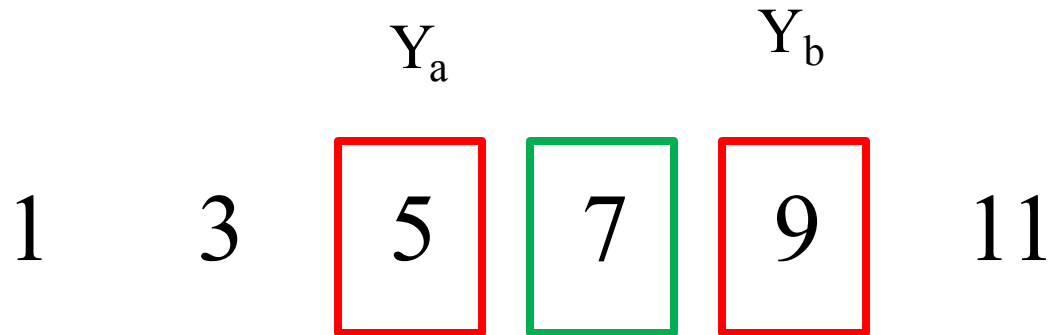
$P(Y_a \text{ and } Y_b \text{ ever compared})$



Consider pivot choice: Y_b

They are compared

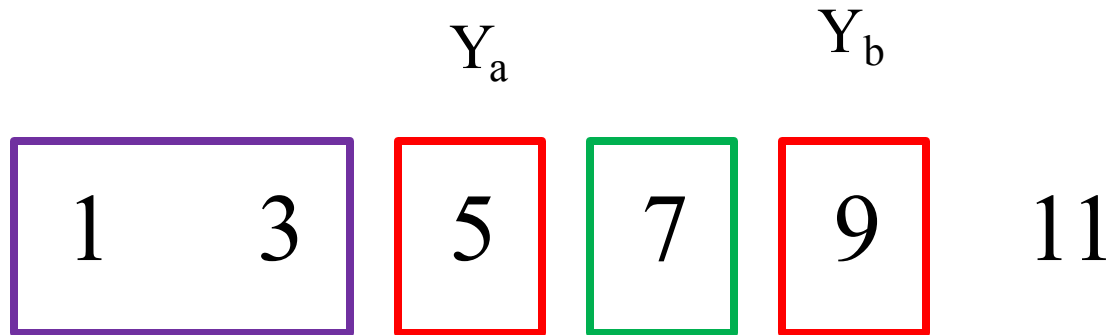
$P(Y_a \text{ and } Y_b \text{ ever compared})$



Consider pivot choice: 7

They are **not** compared

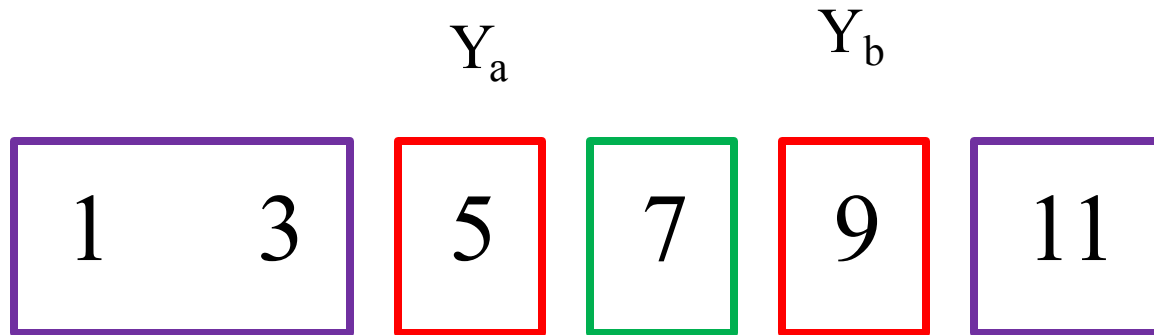
$P(Y_a \text{ and } Y_b \text{ ever compared})$



Consider pivot choice: $< Y_a$

Whether or not they are compared
depends on future pivots

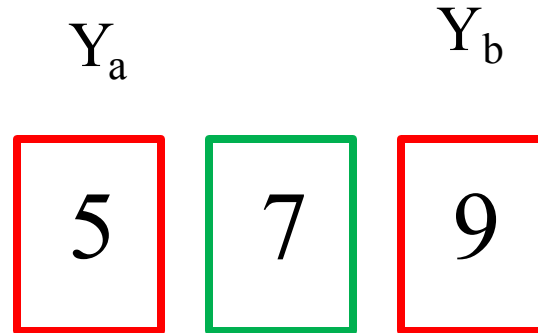
$P(Y_a \text{ and } Y_b \text{ ever compared})$



Consider pivot choice: $> Y_b$

Whether or not they are compared
depends on future pivots

P(Y_a and Y_b ever compared)



Are Y_a and Y_b compared?

Keep repeating pivot choice until you get a pivot
In the range $[Y_a, Y_b]$ inclusive

Expected Running Time of QuickSort

- Let $X = \#$ comparisons made when sorting n elems
 - $E[X]$ gives us expected running time of algorithm
 - Given V_1, V_2, \dots, V_n in random order to sort
 - Let Y_1, Y_2, \dots, Y_n be V_1, V_2, \dots, V_n in sorted order
 - Let $I_{a,b} = 1$ if Y_a and Y_b are compared, 0 otherwise
 - Order where $Y_b > Y_a$, so we have:
$$X = \sum_{a=1}^{n-1} \sum_{b=a+1}^n I_{a,b}$$

Expected Running Time of QuickSort

Aside:
$$X = \sum_{a=1}^{n-1} \sum_{b=a+1}^n I_{a,b}$$

When $a = 1$ $I_{1,2} + I_{1,3} + \dots + I_{1,n}$

When $a = 2$ $+ I_{2,3} + \dots + I_{2,n}$

When $a = n-1$ $+ I_{n-1,n}$

Contains a comparison between each i and j
(where i does not equal j)
exactly once

Expected Running Time of QuickSort

- Let $X = \#$ comparisons made when sorting n elems
 - $E[X]$ gives us expected running time of algorithm
 - Given V_1, V_2, \dots, V_n in random order to sort
 - Let Y_1, Y_2, \dots, Y_n be V_1, V_2, \dots, V_n in sorted order
 - Let $I_{a,b} = 1$ if Y_a and Y_b are compared, 0 otherwise
 - Order where $Y_b > Y_a$, so we have: $X = \sum_{a=1}^{n-1} \sum_{b=a+1}^n I_{a,b}$

$$E[X] = E \left[\sum_{a=1}^{n-1} \sum_{b=a+1}^n I_{a,b} \right] = \sum_{a=1}^{n-1} \sum_{b=a+1}^n E[I_{a,b}] = \sum_{a=1}^{n-1} \sum_{b=a+1}^n P(Y_a \text{ and } Y_b \text{ ever compared})$$

P(Y_a and Y_b ever compared)

- Consider when Y_a and Y_b are directly compared
 - We only care about case where pivot chosen from set:
 $\{Y_a, Y_{a+1}, Y_{a+2}, \dots, Y_b\}$
 - From that set either Y_a and Y_b must be selected as pivot (with equal probability) in order to be compared
 - So,

$$P(Y_a \text{ and } Y_b \text{ ever compared}) = \frac{2}{b-a+1}$$

$$E[X] = \sum_{a=1}^{n-1} \sum_{b=a+1}^n P(Y_a \text{ and } Y_b \text{ ever compared}) = \sum_{a=1}^{n-1} \sum_{b=a+1}^n \frac{2}{b-a+1}$$

Bring it on Home (i.e. Solve the Sum)

$$E[X] = \sum_{a=1}^{n-1} \sum_{b=a+1}^n \frac{2}{b-a+1}$$
$$\sum_{b=a+1}^n \frac{2}{b-a+1} \approx \int_{a+1}^n \frac{2}{b-a+1} db \quad \text{Recall: } \int \frac{1}{x} dx = \ln(x)$$

Thanks
Riemann

$$= 2 \ln(b-a+1) \Big|_{a+1}^n = 2 \ln(n-a+1) - 2 \ln(2)$$

$$\approx 2 \ln(n-a+1) \quad \text{for large } n$$

$$E[X] \approx \sum_{a=1}^{n-1} 2 \ln(n-a+1) \approx 2 \int_{a=1}^{n-1} \ln(n-a+1) da \quad \text{Let } y = n-a+1$$

$$= -2 \int_{y=n}^2 \ln(y) dy$$

$$= -2(y \ln(y) - y) \Big|_n^2$$

Recall:
 $\int \ln(x) dx = x \ln(x) - x$

$$= -2[(2 \ln(2) - 2) - (n \ln(n) - n)] \approx 2n \ln(n) - 2n = O(n \log n)$$

Ahhh 😊