

Uncertain $\langle T \rangle$

A First-Order Type for Uncertain Data

James Bornholt

Australian National University

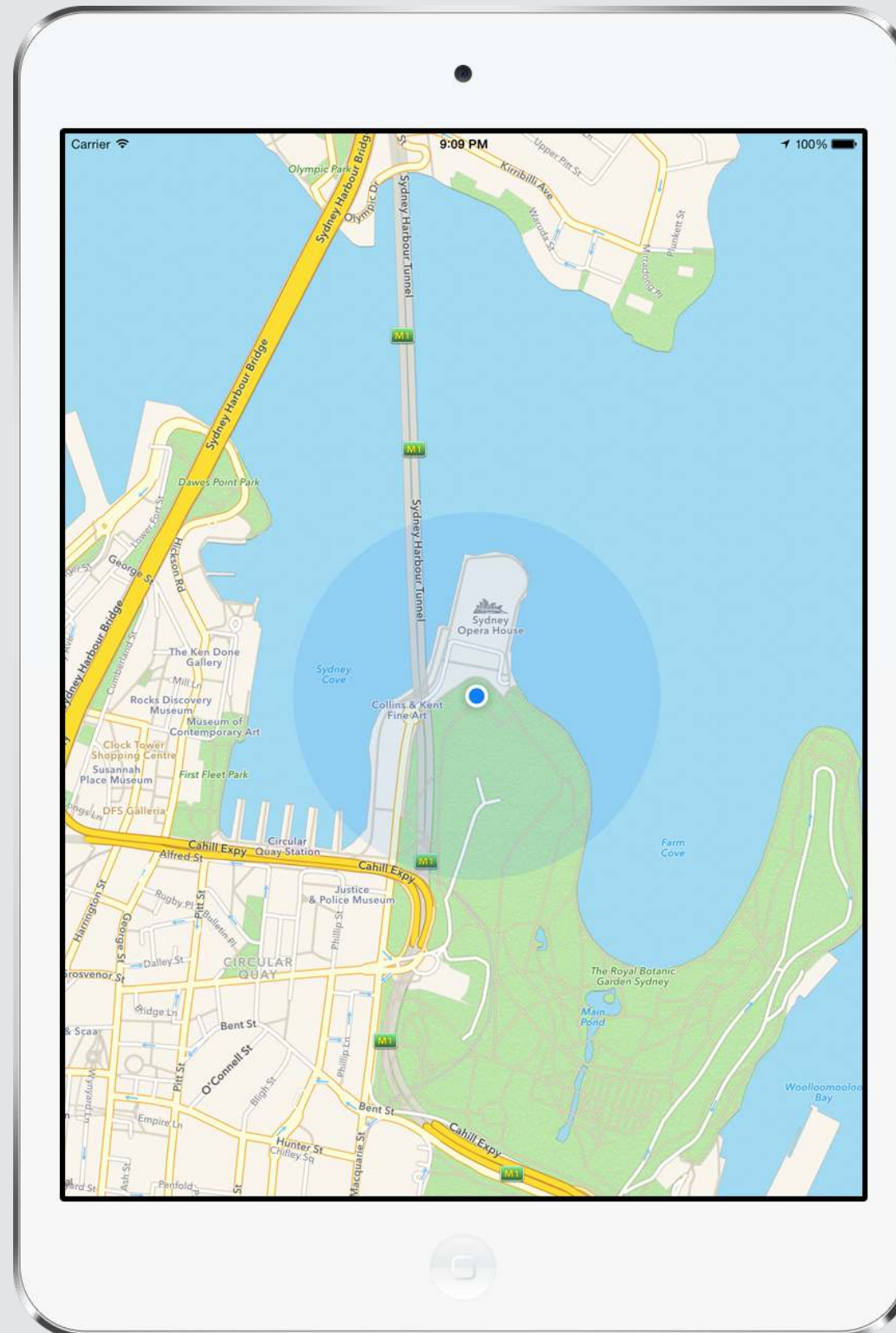
Todd Mytkowicz

Microsoft Research

Kathryn S. McKinley

Microsoft Research

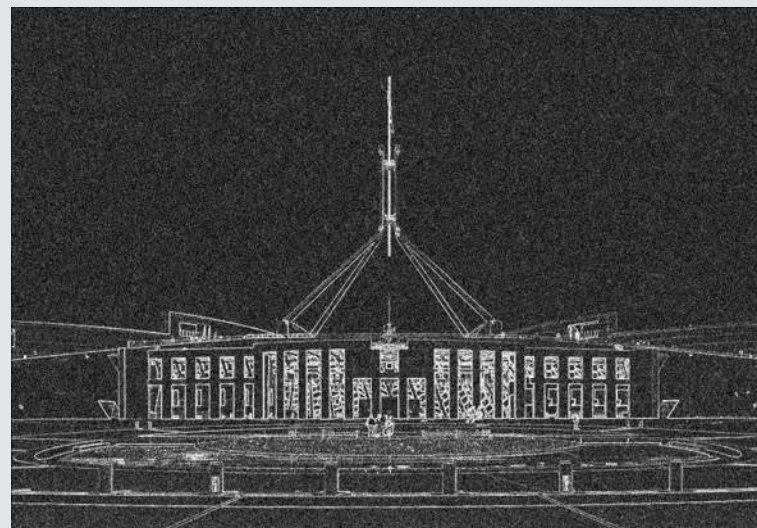
Sensors



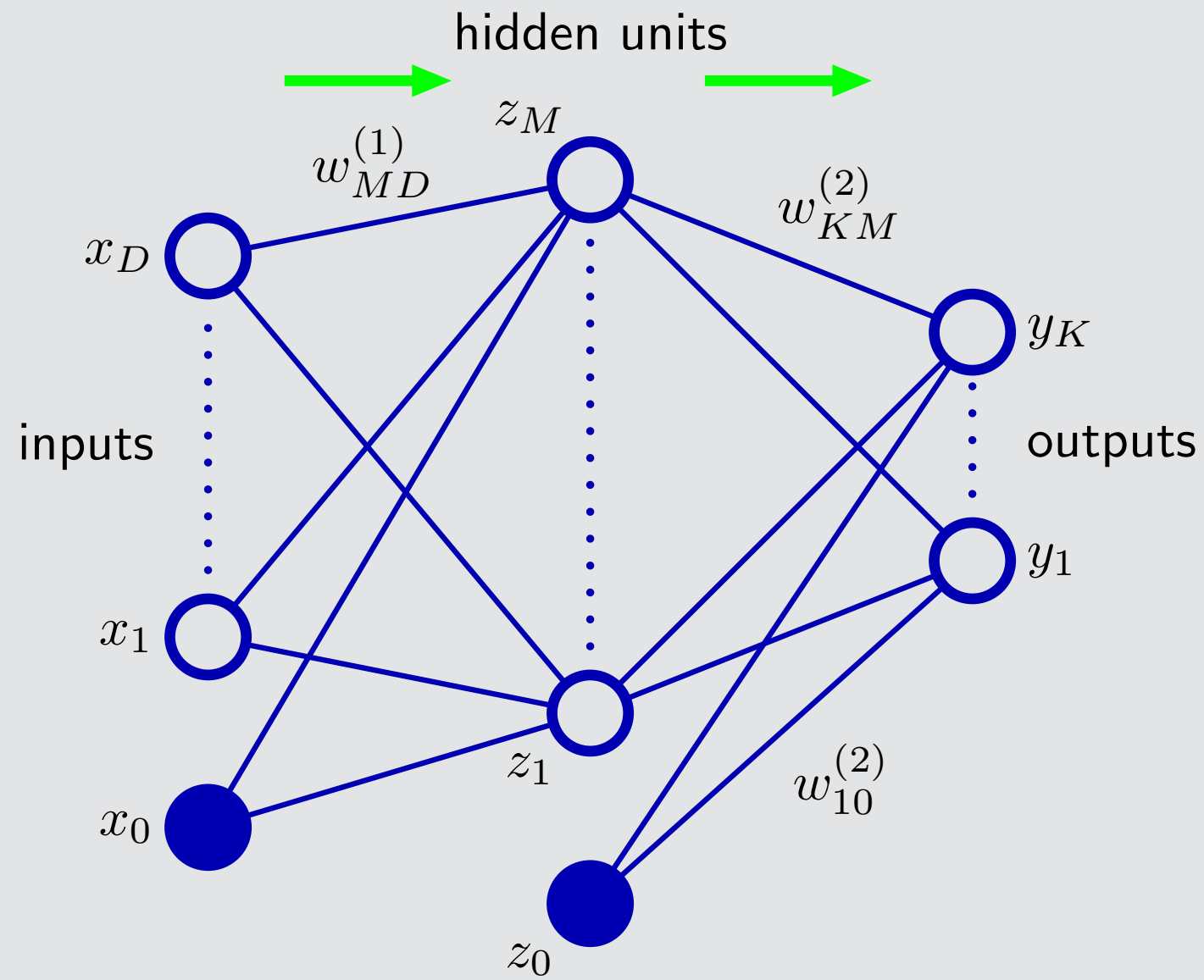
Approximate computing

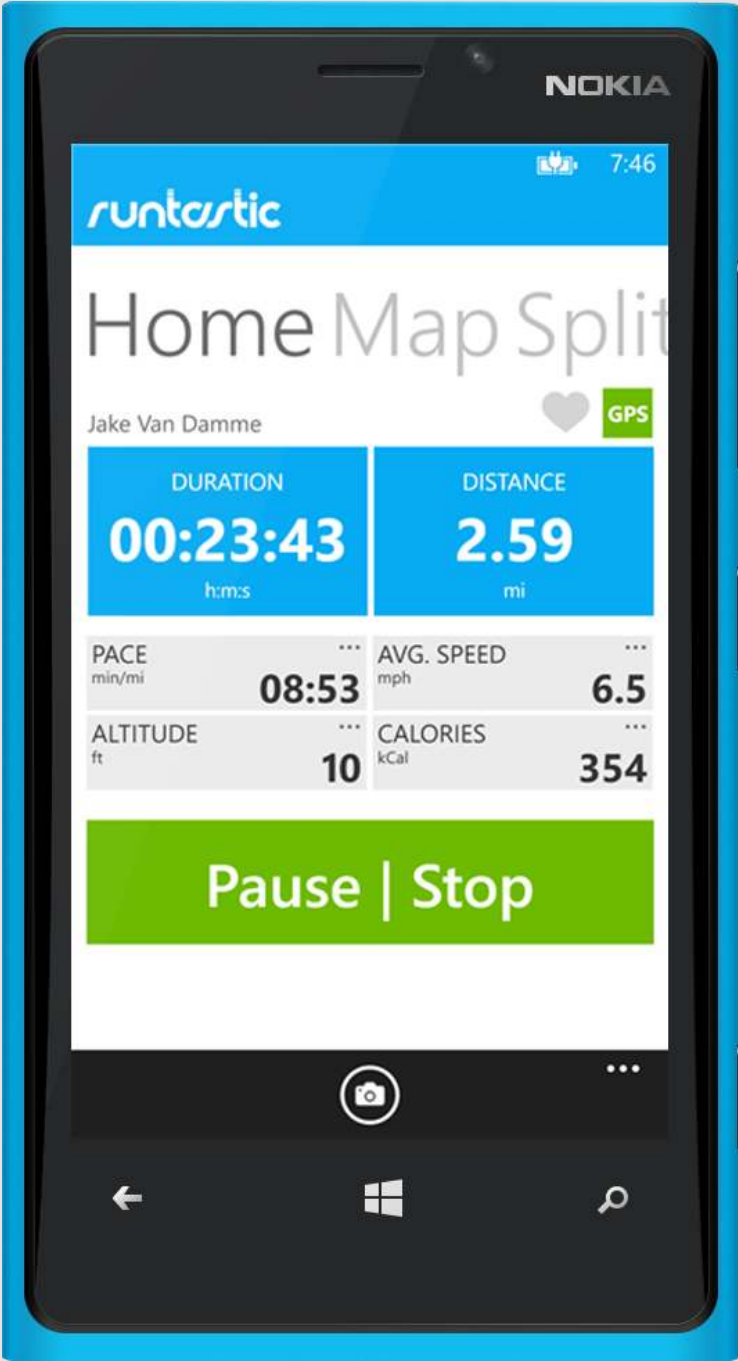


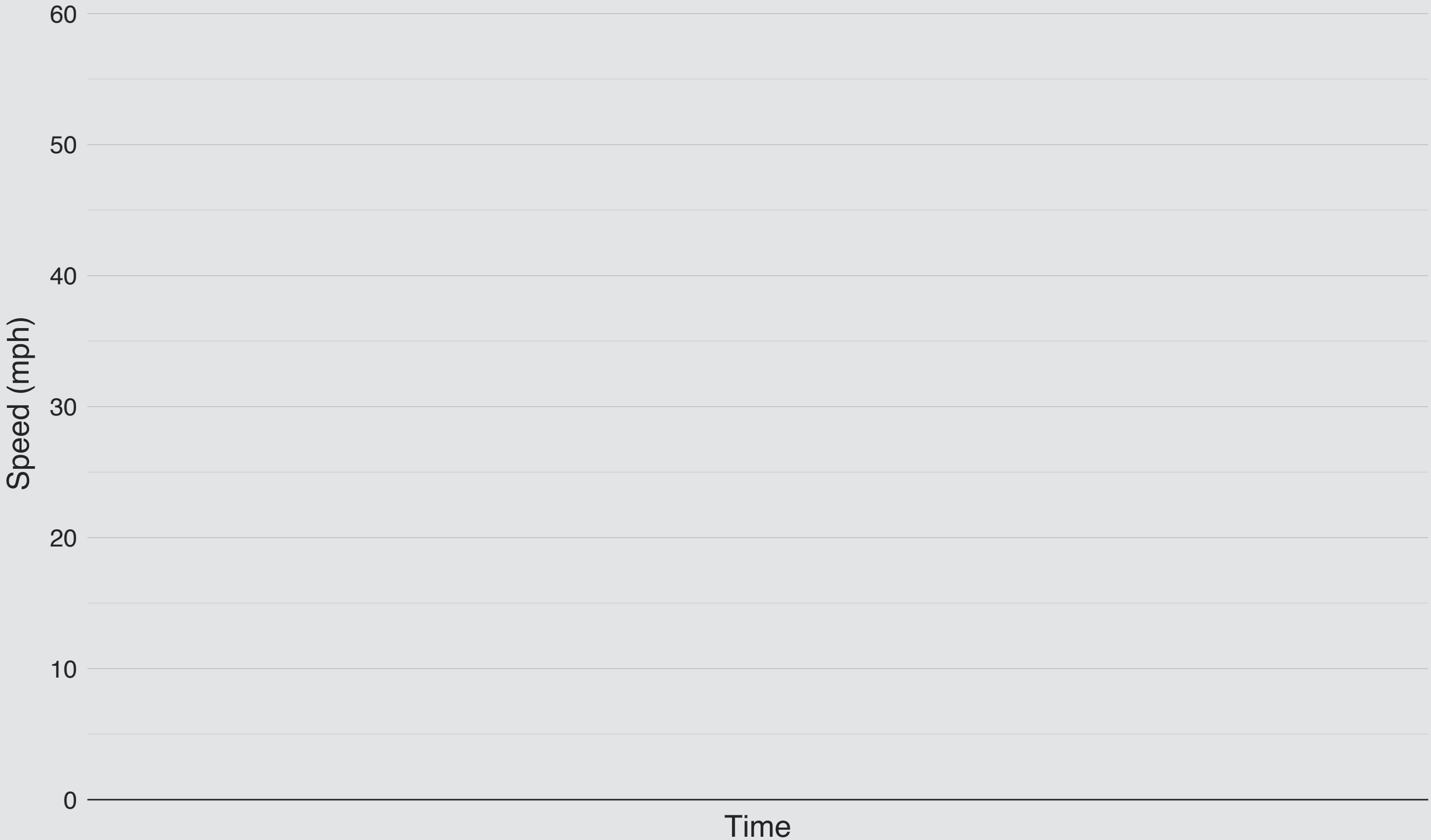
approximate edge detection

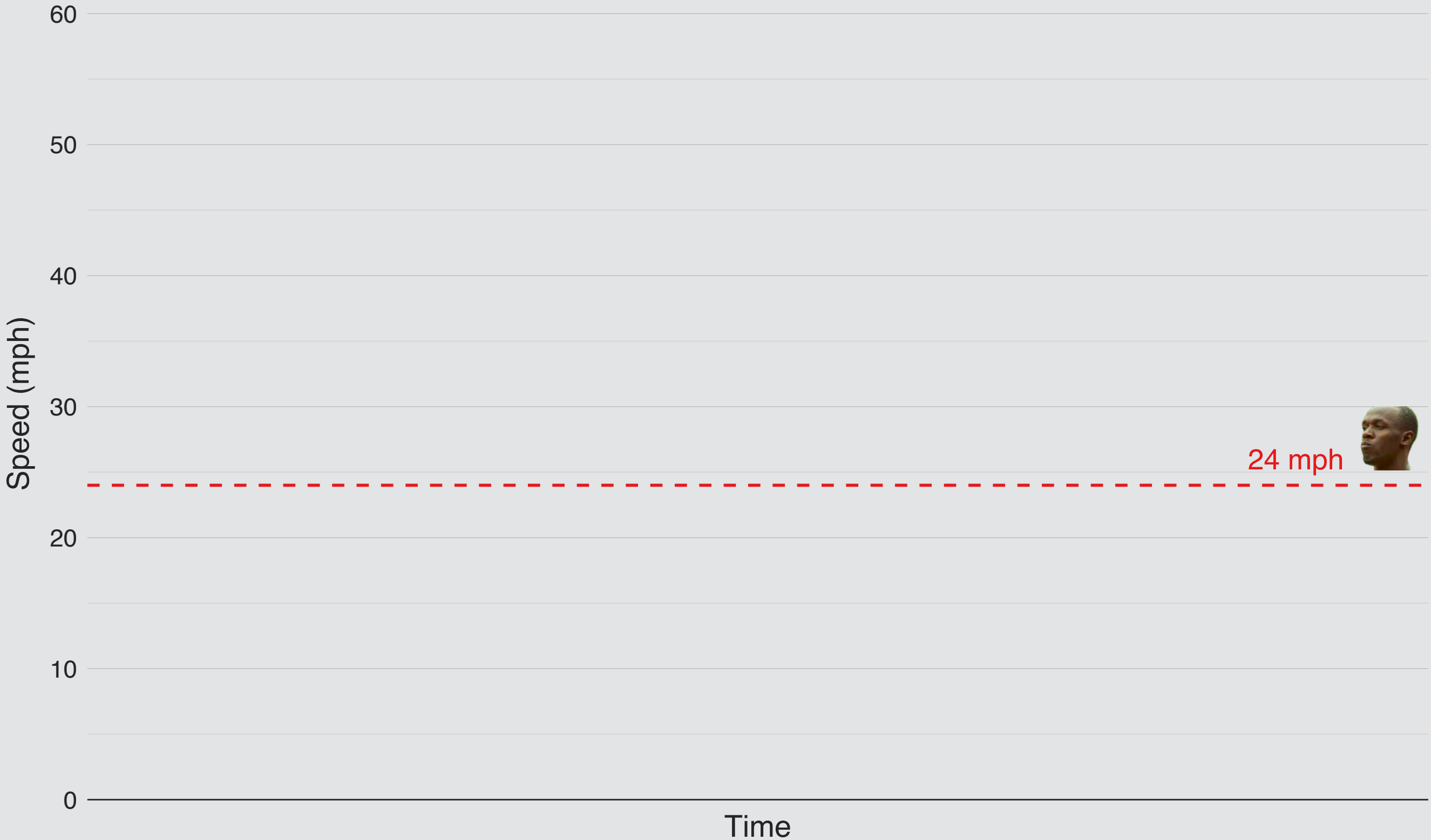


Machine learning

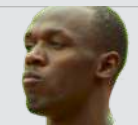


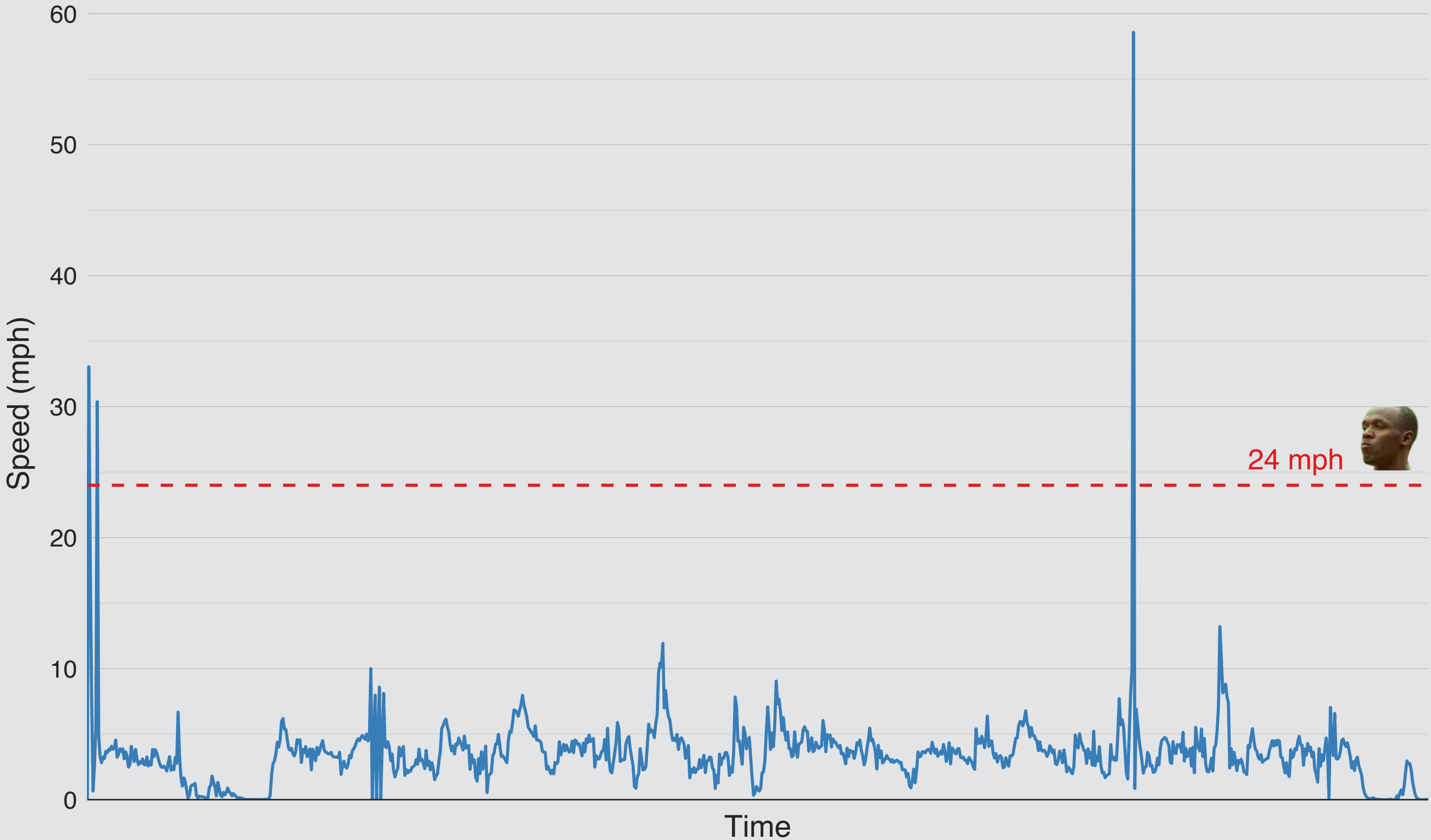




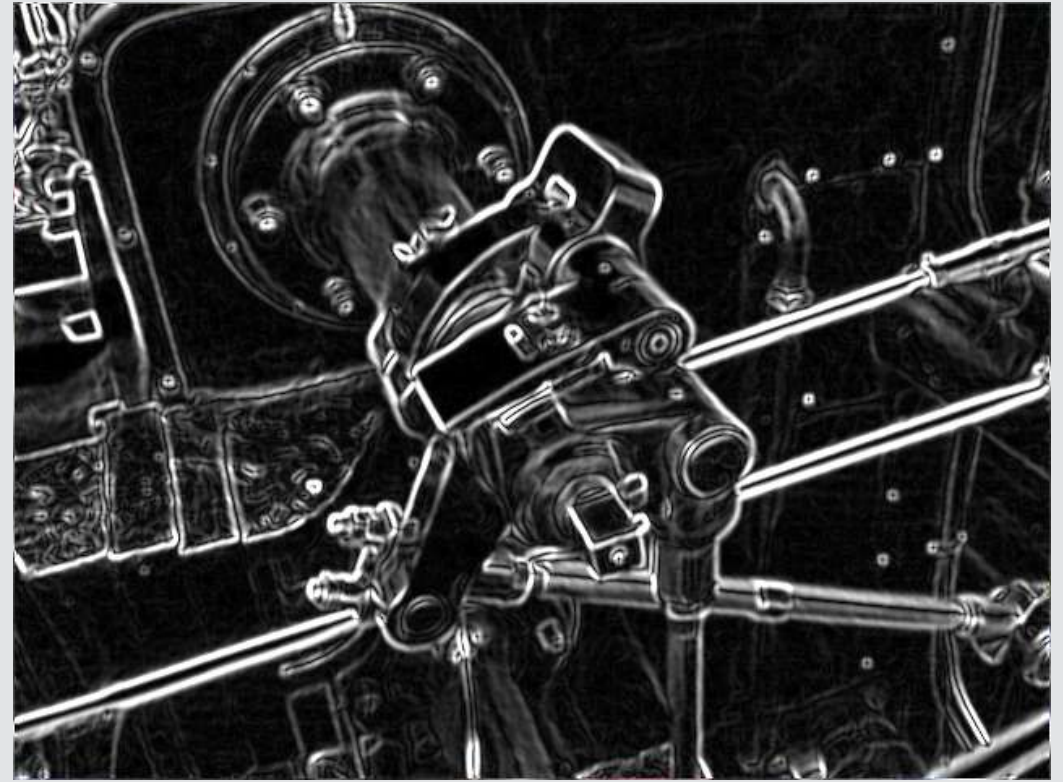
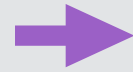
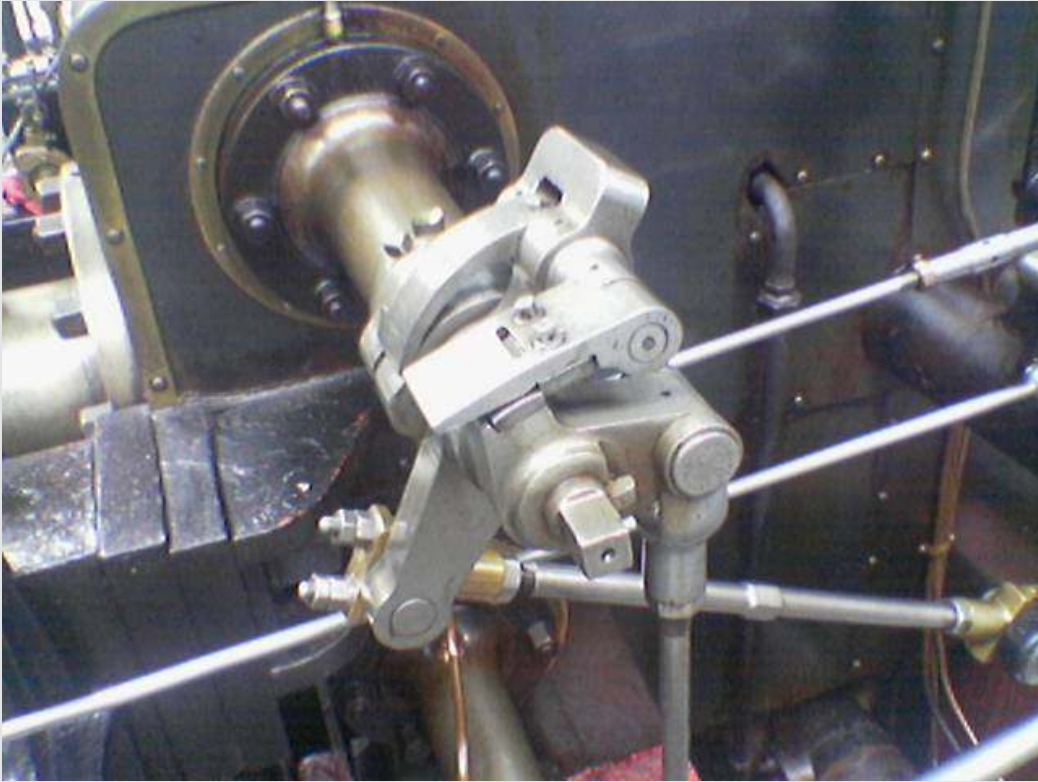


24 mph

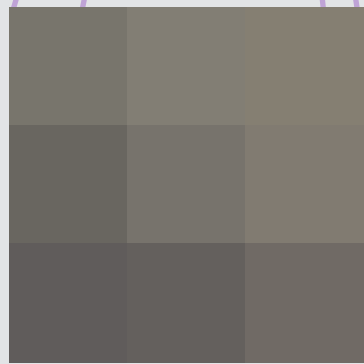
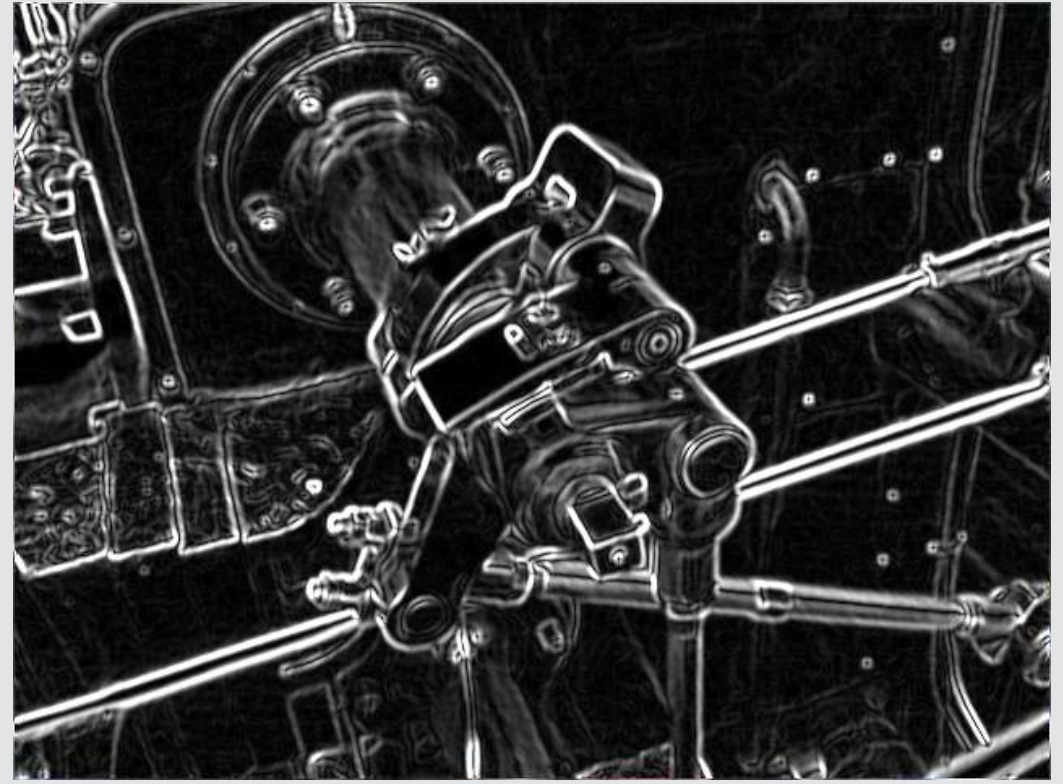
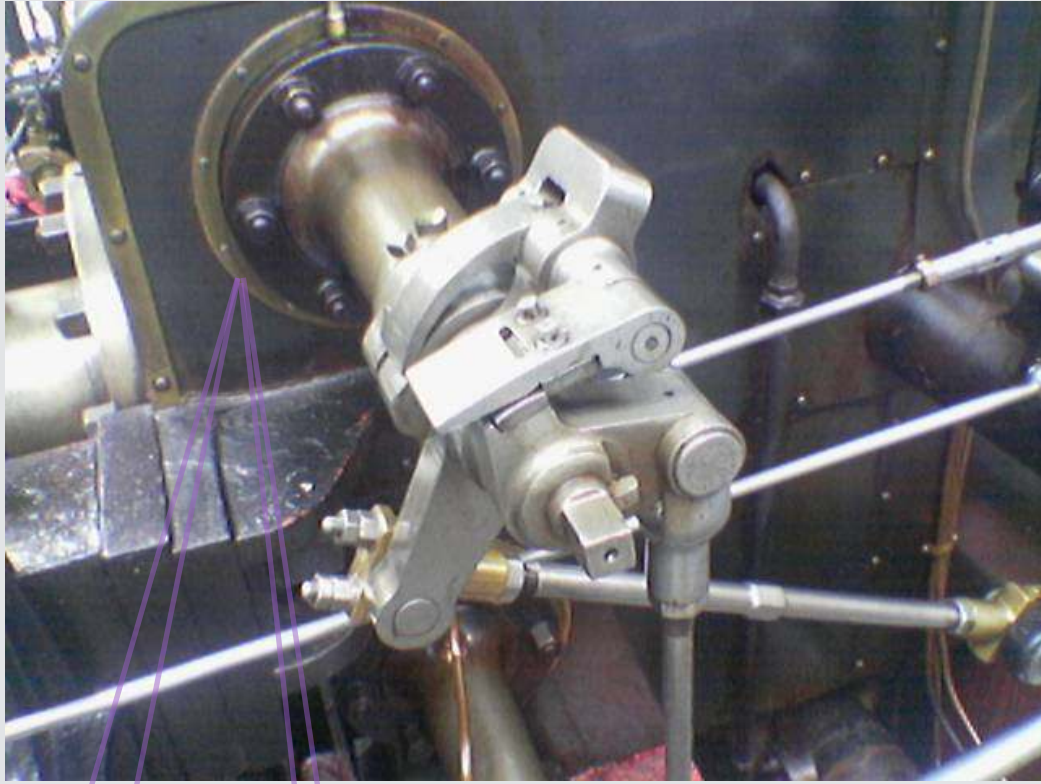




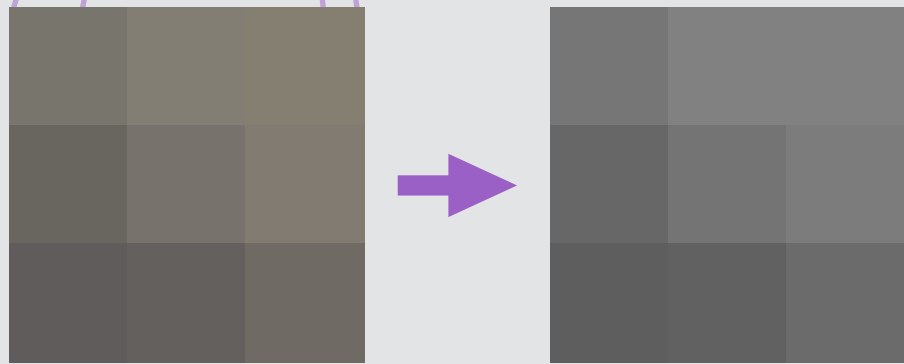
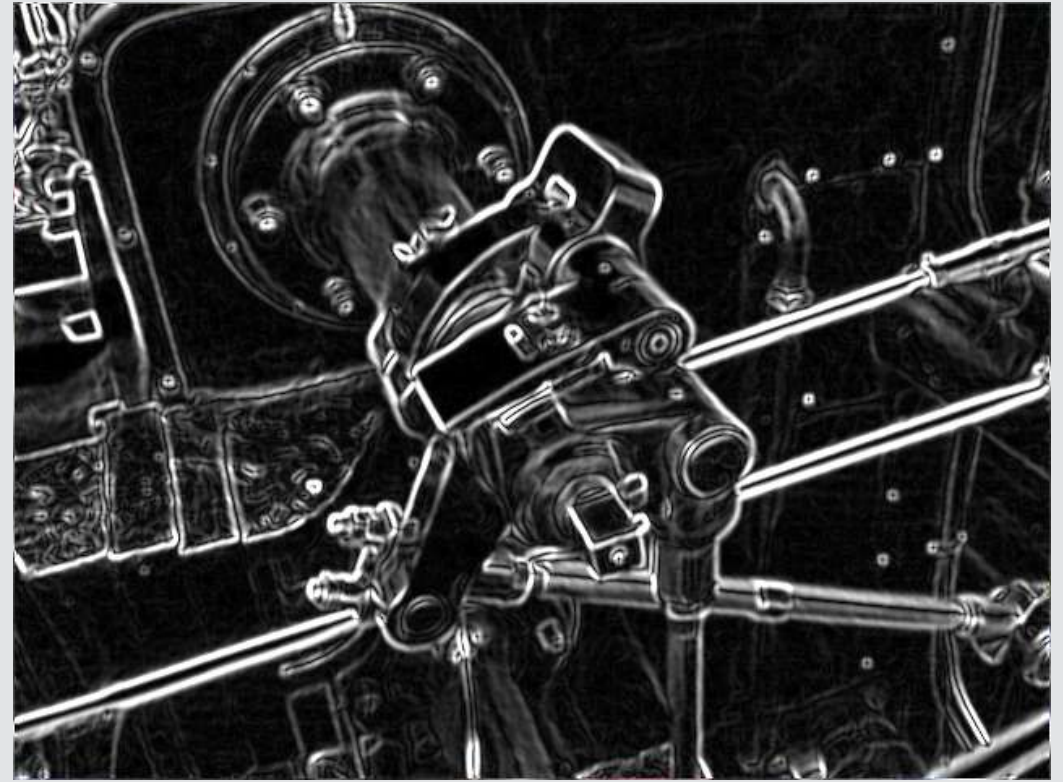
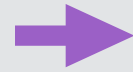
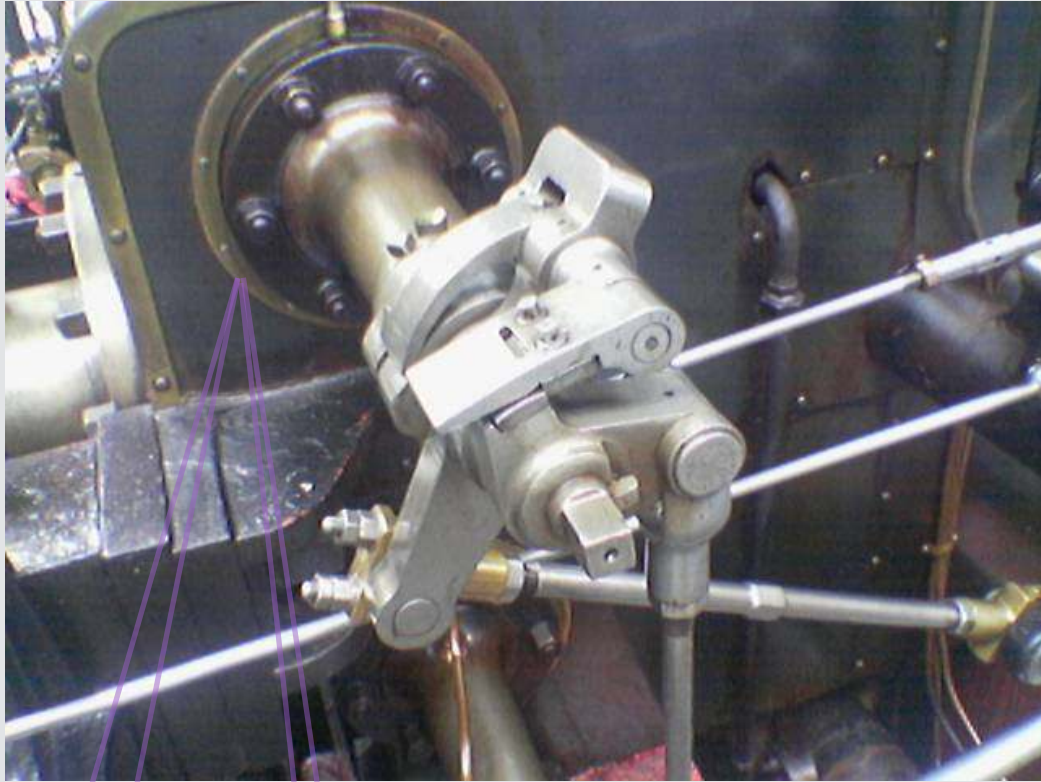
Edge detection



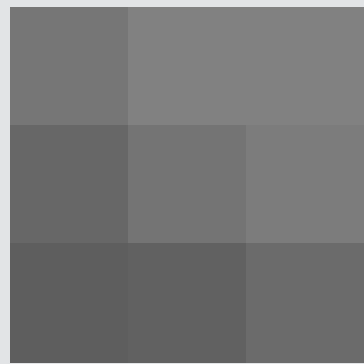
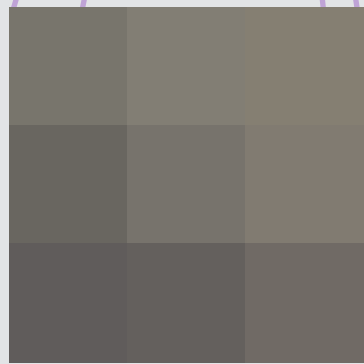
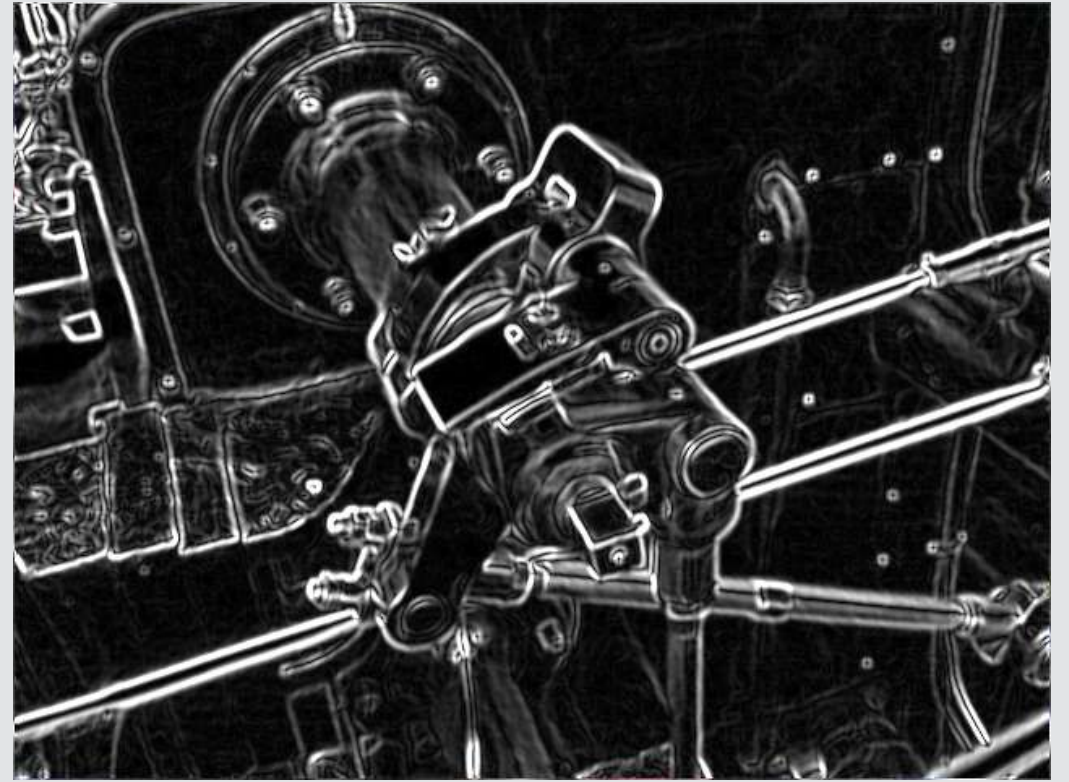
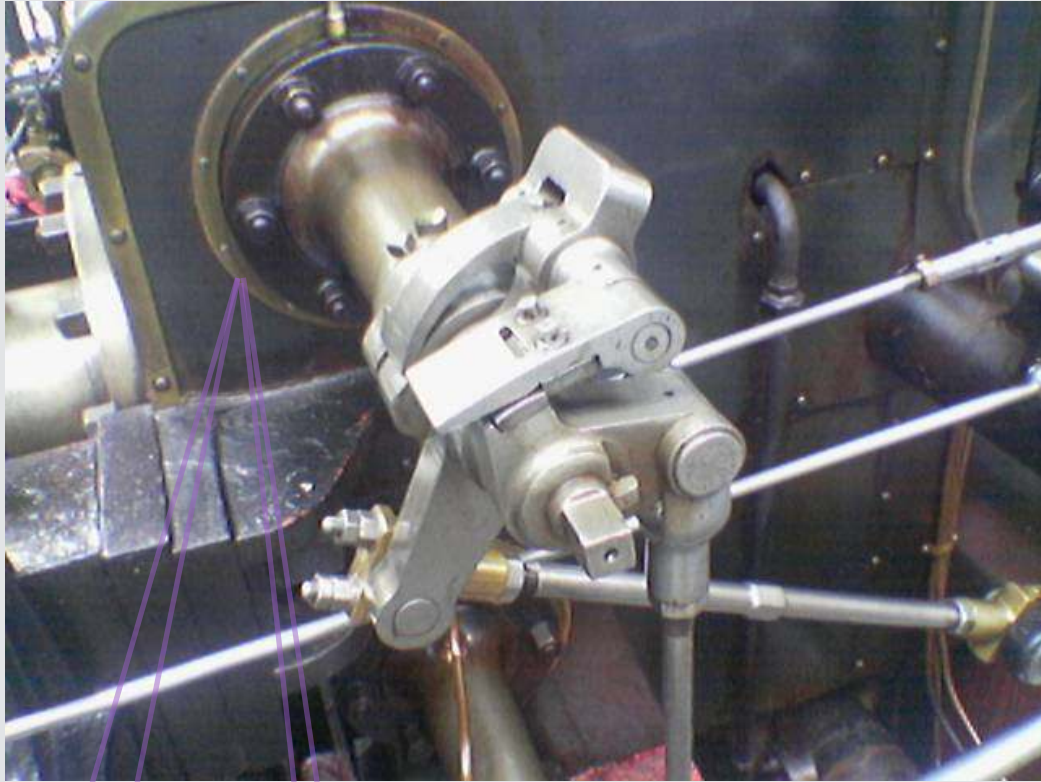
Edge detection



Edge detection

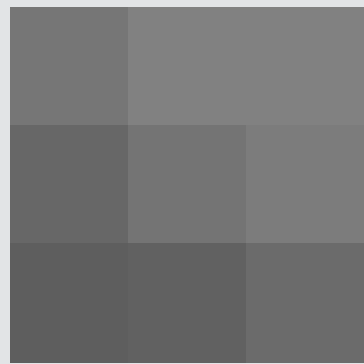
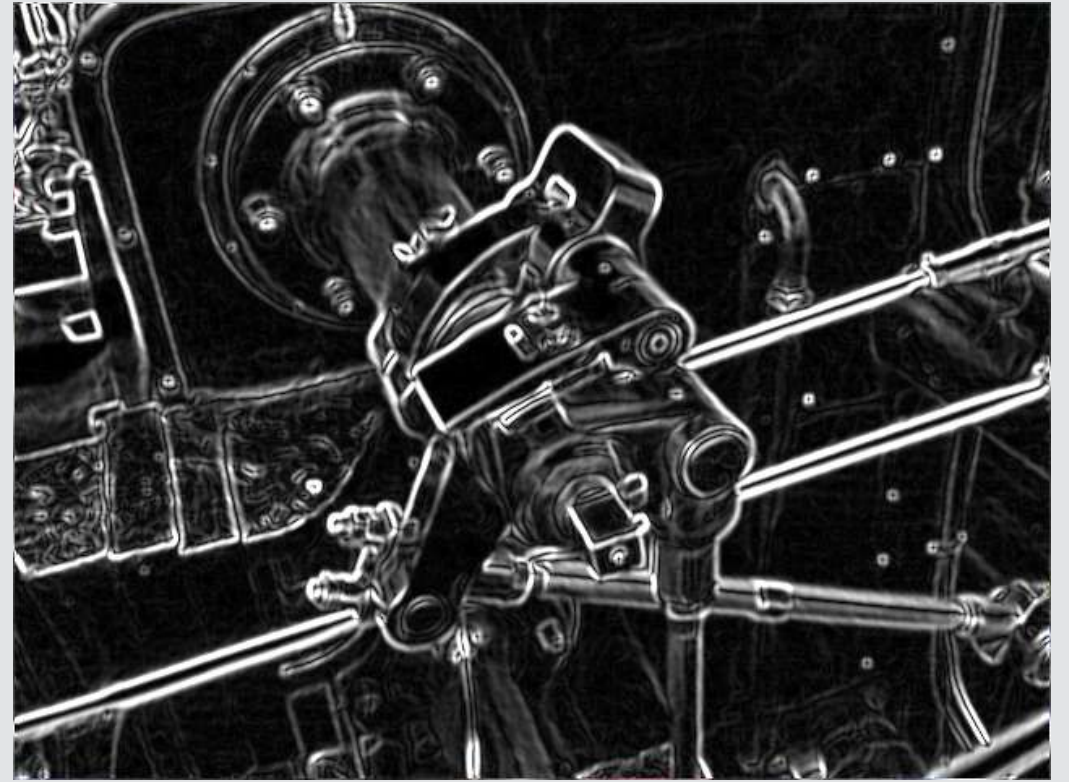
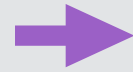
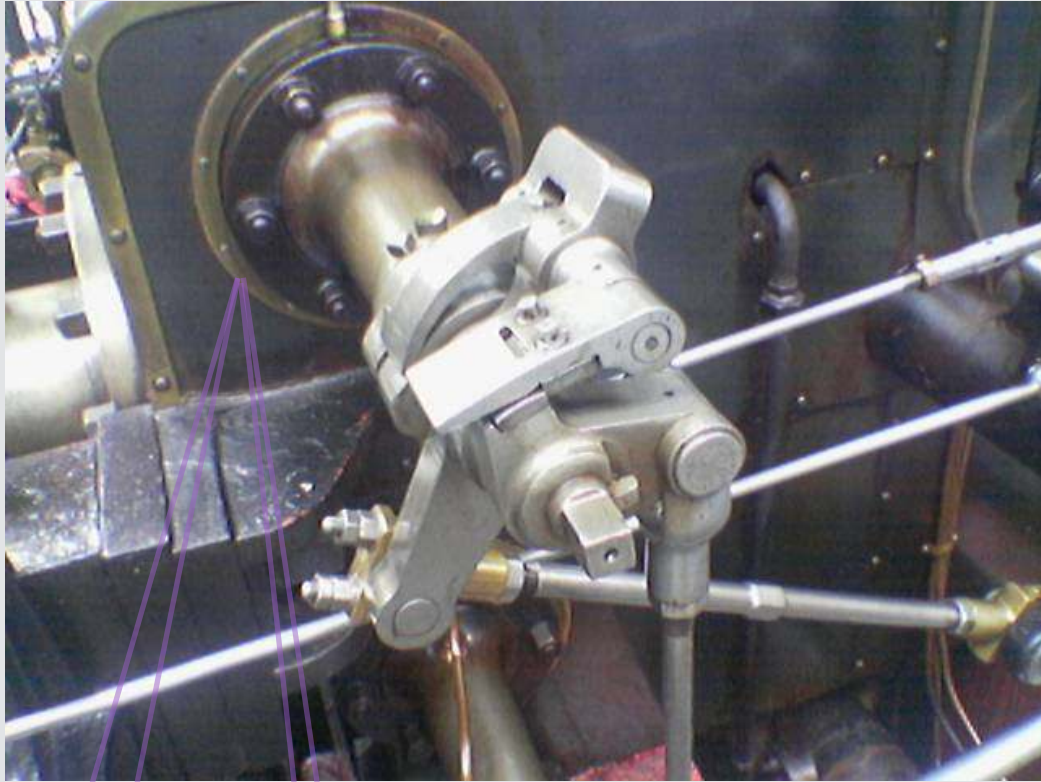


Edge detection



Sobel(p)

Edge detection

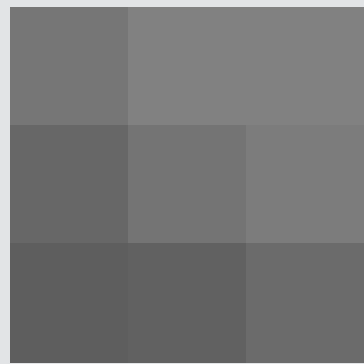
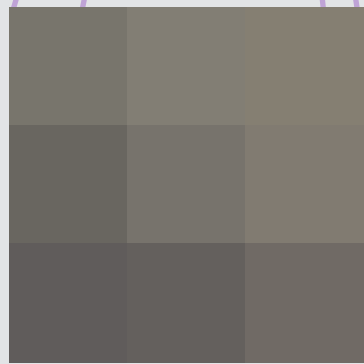
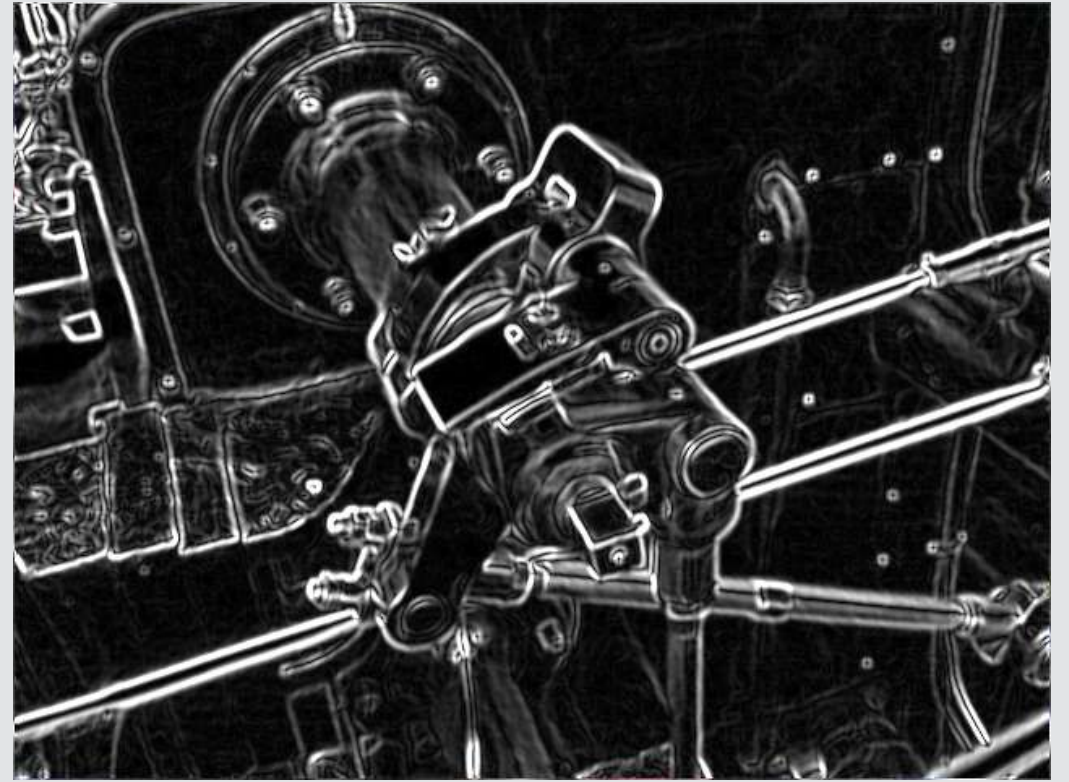
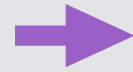
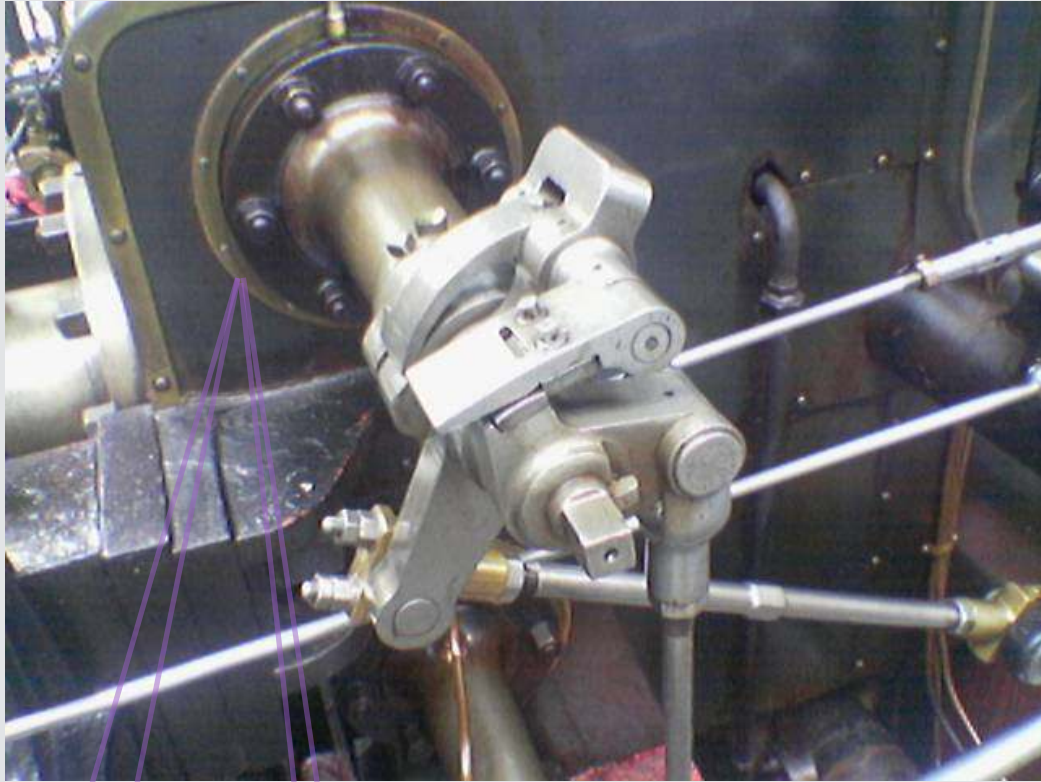


Sobel(p)



0.4940

Edge detection



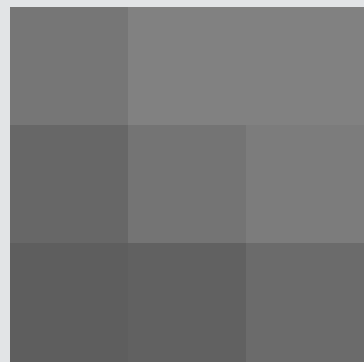
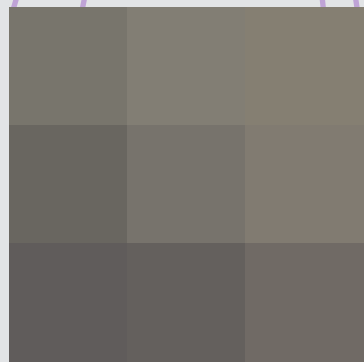
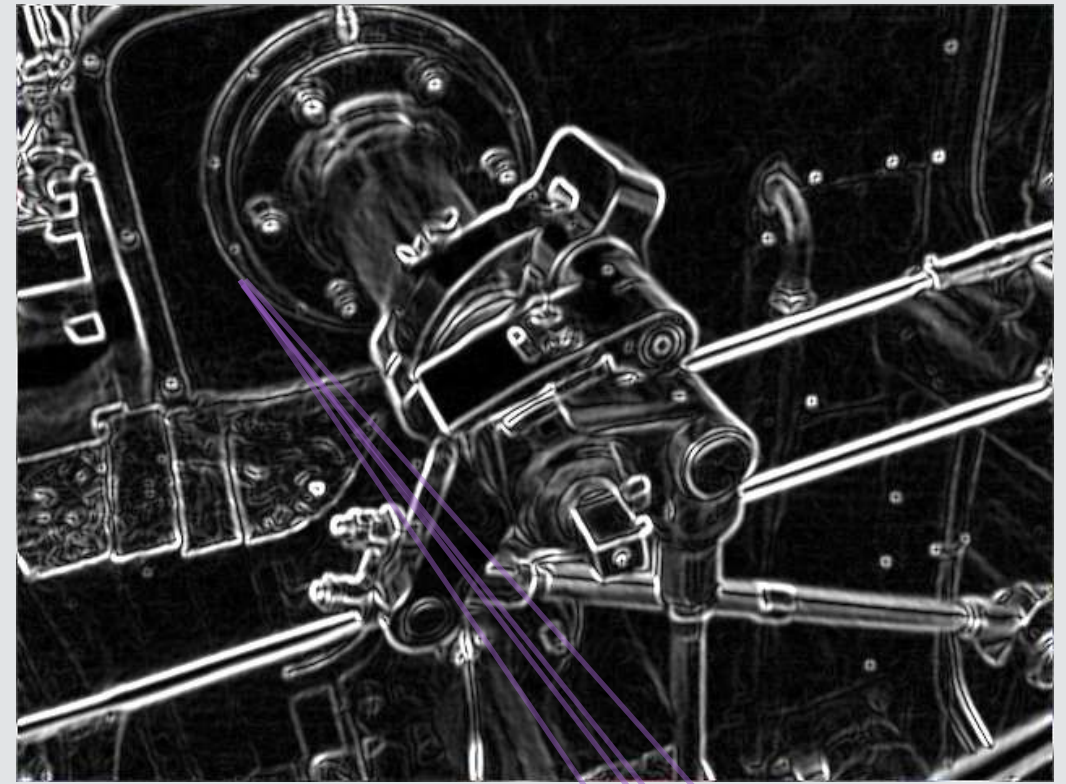
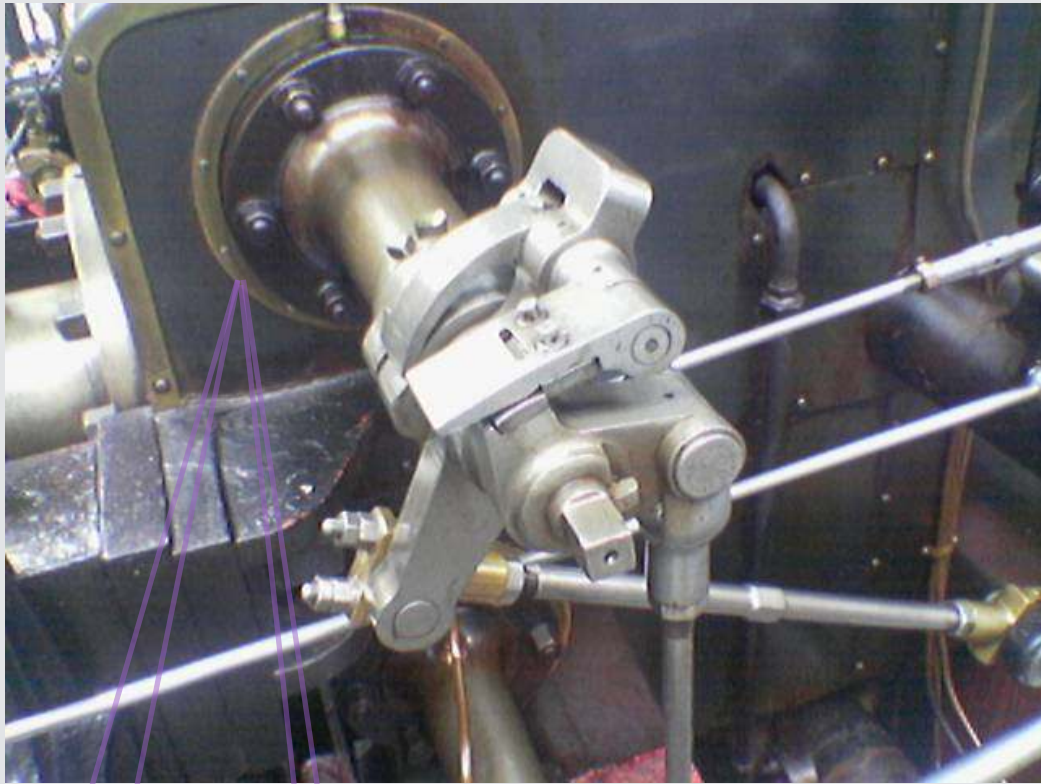
Sobel(p)



0.4940



Edge detection



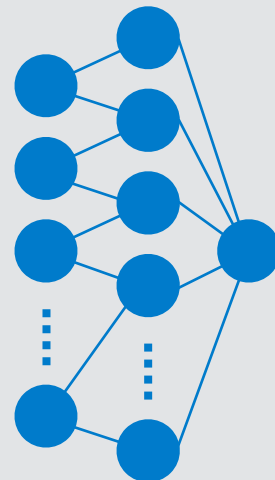
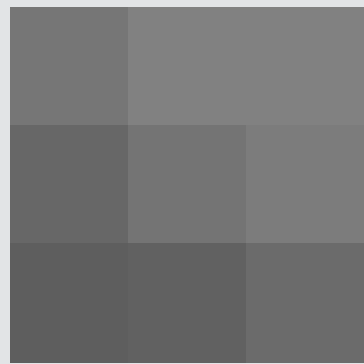
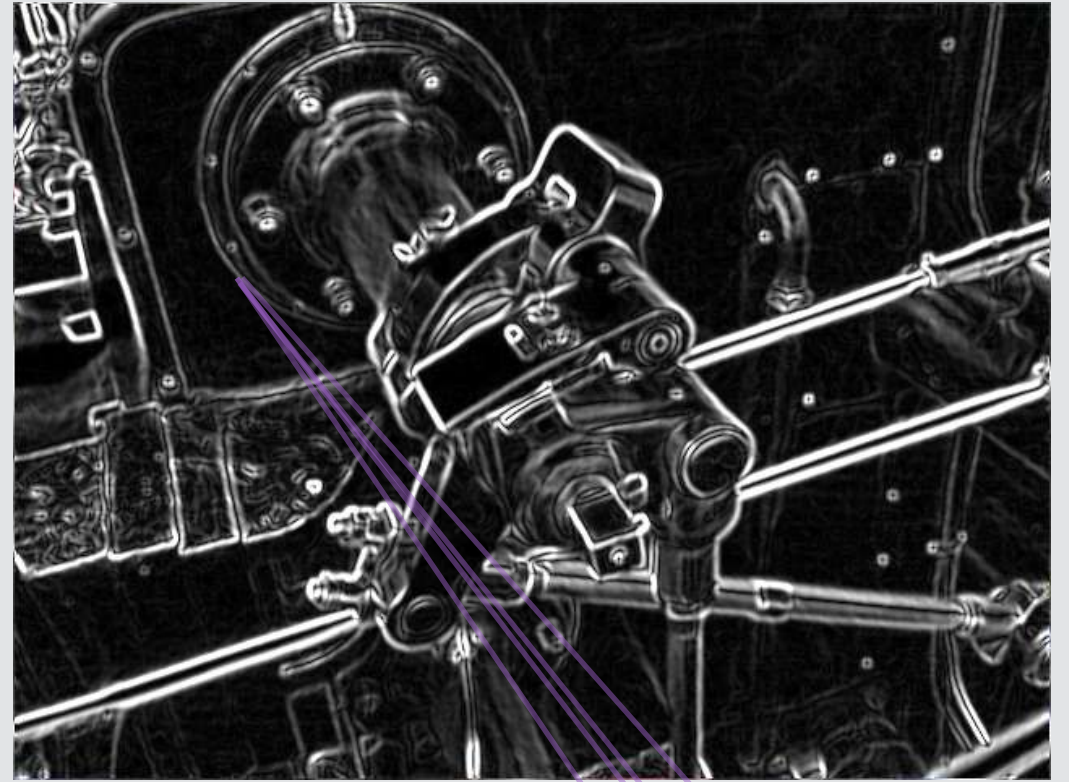
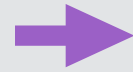
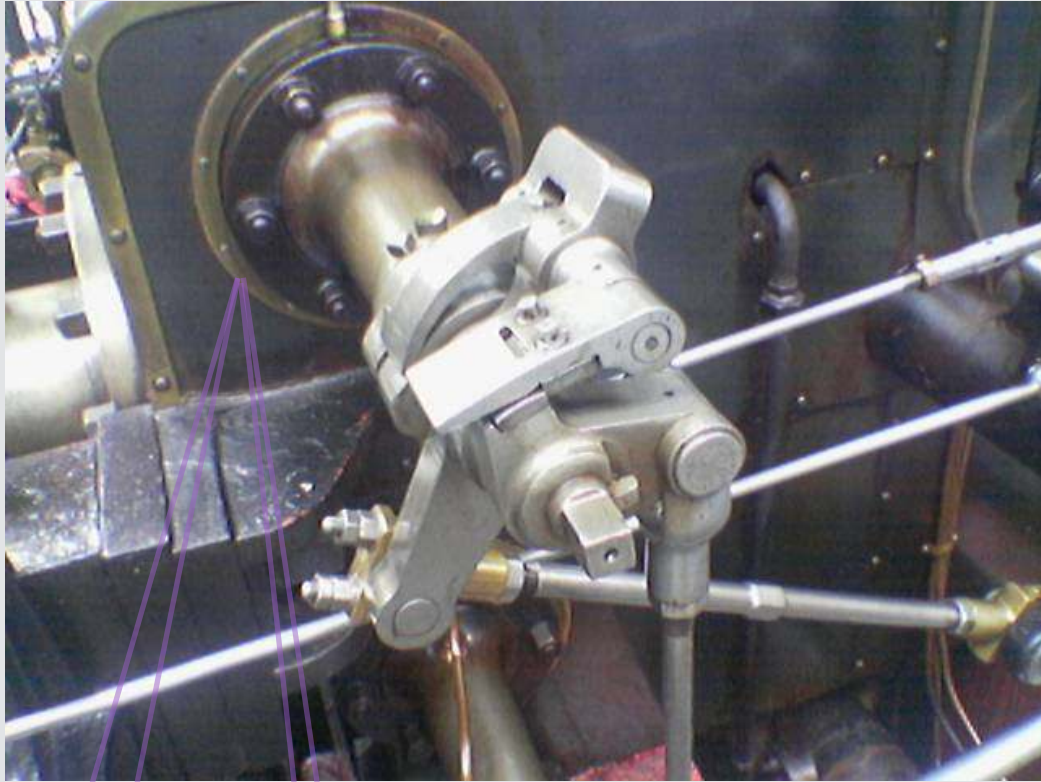
$\text{Sobel}(p)$



0.4940



Approximate edge detection



0.4940



3.4% average error

Approximate edge detection

What is the gradient at pixel p ?

Sobel1(p)

3.4% average
training error

Approximate edge detection

What is the gradient at pixel p ?

`Sobel(p)`

3.4% average
training error

Is there an edge at pixel p ?

```
if (Sobel(p) > 0.1)  
    EdgeFound();
```

Approximate edge detection

What is the gradient at pixel p ?

`Sobel(p)`

3.4% average
training error

Is there an edge at pixel p ?

```
if (Sobel(p) > 0.1)  
    EdgeFound();
```

36% false positives
on the same data!

Approximate edge detection

What is the gradient at pixel p ?

`Sobel(p)`

3.4% average
training error

Is there an edge at pixel p ?

```
if (Sobel(p) > 0.1)  
    EdgeFound();
```

36% false positives
on the same data!

Computation compounds uncertainty!

```
GeoCoordinate Location;
```

```
double Grad = Sobel(p);
```

```
Uncertain<GeoCoordinate> Location;
```

```
Uncertain<double> Grad = Sobel(p);
```


Uncertain<T> is an uncertain type abstraction.

It encourages **non-expert developers** to explicitly reason about uncertainty.


```
Uncertain<GeoCoordinate> LastLoc =  
    GPS.GetLocation();  
Sleep(5);  
Uncertain<GeoCoordinate> Loc =  
    GPS.GetLocation();
```

```
Uncertain<GeoCoordinate> LastLoc =  
    GPS.GetLocation();  
Sleep(5);  
Uncertain<GeoCoordinate> Loc =  
    GPS.GetLocation();  
  
Uncertain<double> Dist =  
    GPS.Distance(Loc, LastLoc);  
Uncertain<double> Speed = Dist / 5;
```

```
Uncertain<GeoCoordinate> LastLoc =  
    GPS.GetLocation();  
Sleep(5);  
Uncertain<GeoCoordinate> Loc =  
    GPS.GetLocation();  
  
Uncertain<double> Dist =  
    GPS.Distance(Loc, LastLoc);  
Uncertain<double> Speed = Dist / 5;  
  
if (Speed > 4) print("Great job!");
```

```
Uncertain<GeoCoordinate> LastLoc =  
    GPS.GetLocation();  
Sleep(5);  
Uncertain<GeoCoordinate> Loc =  
    GPS.GetLocation();  
  
Uncertain<double> Dist =  
    GPS.Distance(Loc, LastLoc);  
Uncertain<double> Speed = Dist / 5;  
  
if (Speed > 4) print("Great job!");  
  
print("Your speed: " + Speed.E());
```

```
Uncertain<GeoCoordinate> LastLoc =  
    GPS.GetLocation();  
Sleep(5);  
Uncertain<GeoCoordinate> Loc =  
    GPS.GetLocation();  
  
Uncertain<double> Dist =  
    GPS.Distance(Loc, LastLoc);  
Uncertain<double> Speed = Dist / 5;  
  
if (Speed > 4) print("Great job!");  
  
print("Your speed: " + Speed.E());
```

Just \$24.99



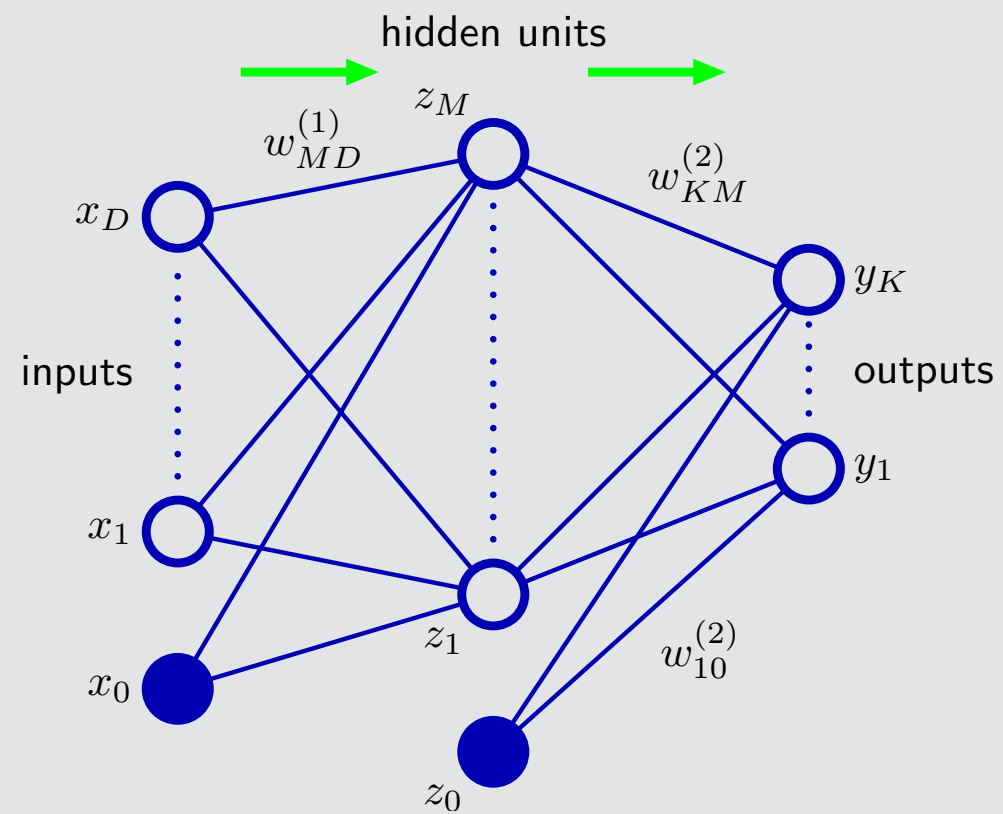
Download from
Windows Phone Store

Probabilistic programming

BUGS, Church, Infer.NET, ...

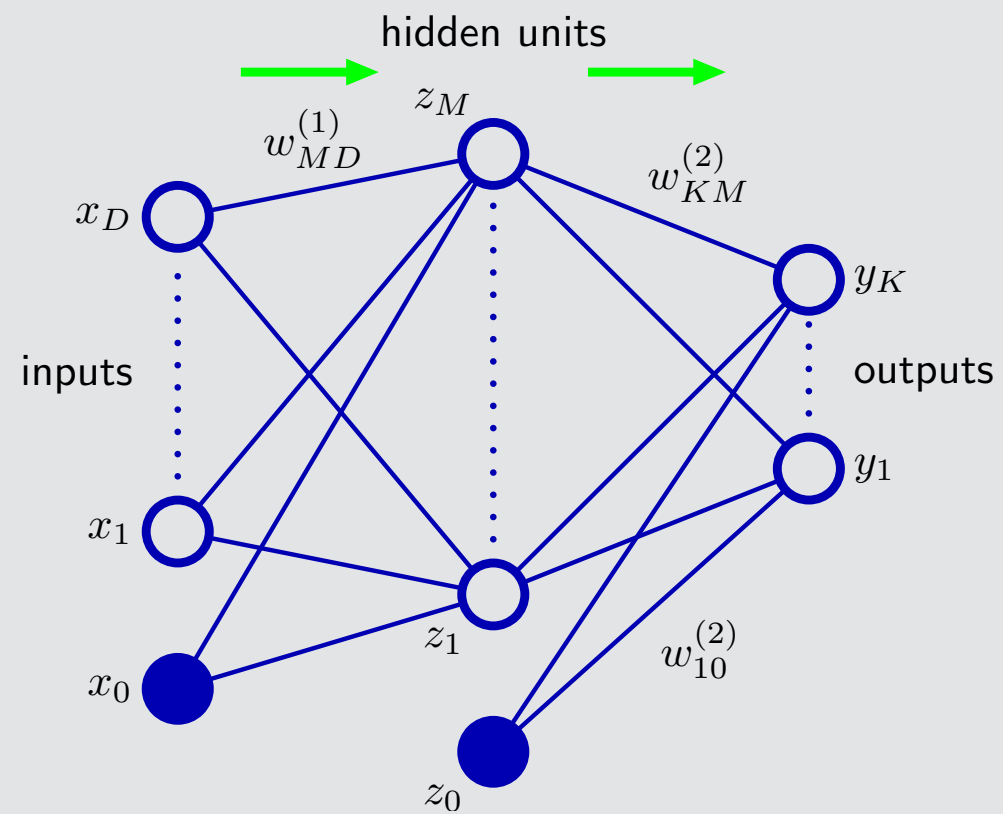
Probabilistic programming

BUGS, Church, Infer.NET, ...



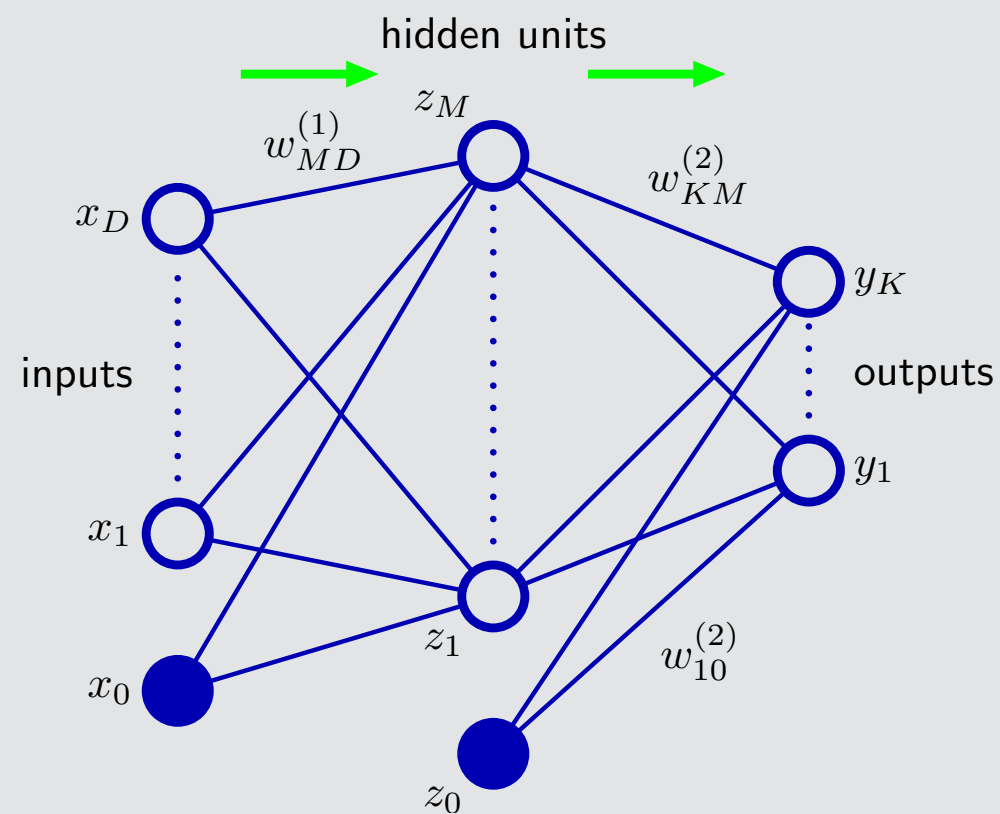
Probabilistic programming

BUGS, Church, Infer.NET, ...



Probabilistic programming

BUGS, Church, Infer.NET, ...



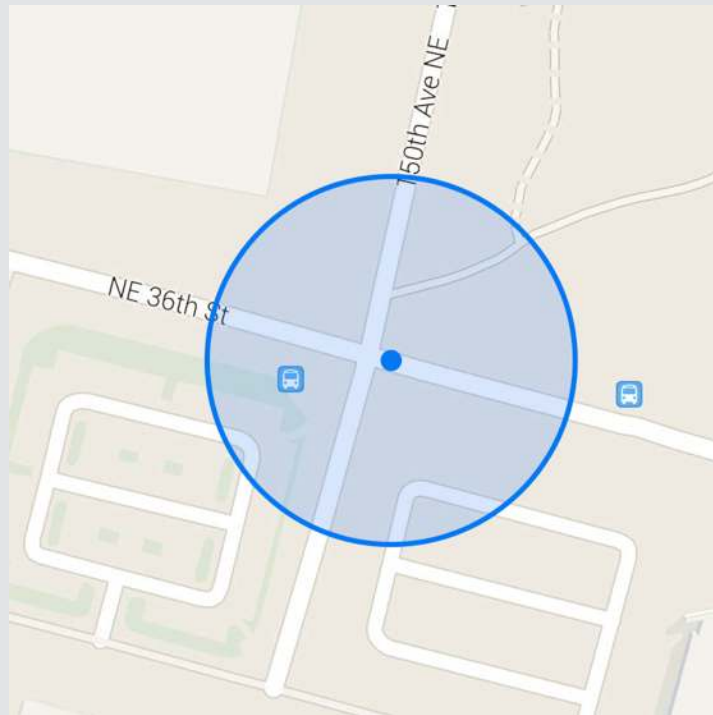
Uncertain $\langle T \rangle$ helps developers *without* statistics PhDs.

```
Uncertain<GeoCoordinate> LastLoc =  
    GPS.GetLocation();
```

A variable of type `Uncertain<T>` is a **random variable**,
represented by a **distribution**.

```
Uncertain<GeoCoordinate> LastLoc =  
    GPS.GetLocation();
```

A variable of type `Uncertain<T>` is a **random variable**,
represented by a **distribution**.



*“We define accuracy as the
radius of 68% confidence [of a]
normal distribution.”*

—Android

Sampling functions return random samples.

Sampling functions return random samples.

- ✓ Simple computations.

Sampling functions return random samples.

- ✓ Simple computations.
- ✓ Represent many distributions.

Sampling functions return random samples.

- ✓ Simple computations.
- ✓ Represent many distributions.
- ✗ Sampling is approximate.

(Later: how Uncertain $\langle T \rangle$ learned to love approximation, and you can too)

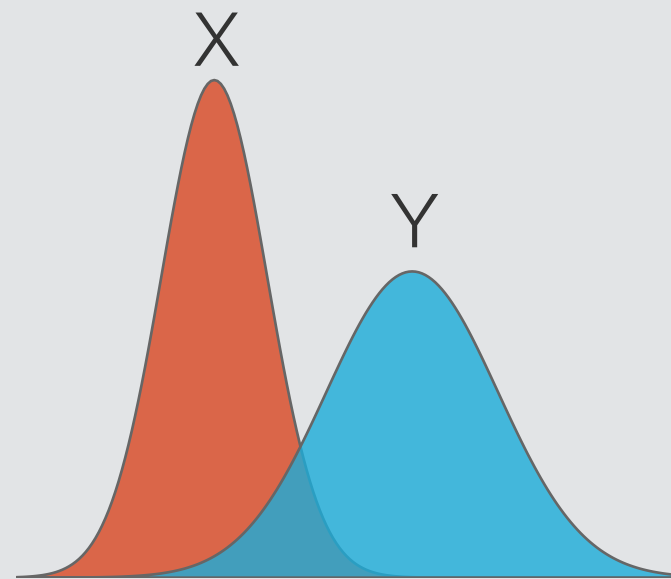
```
Uncertain<double> Speed = Dist / 5;
```

```
Uncertain<double> Speed = Dist / 5;
```

Or more generally, $Z = X + Y$, if X and Y are distributions.

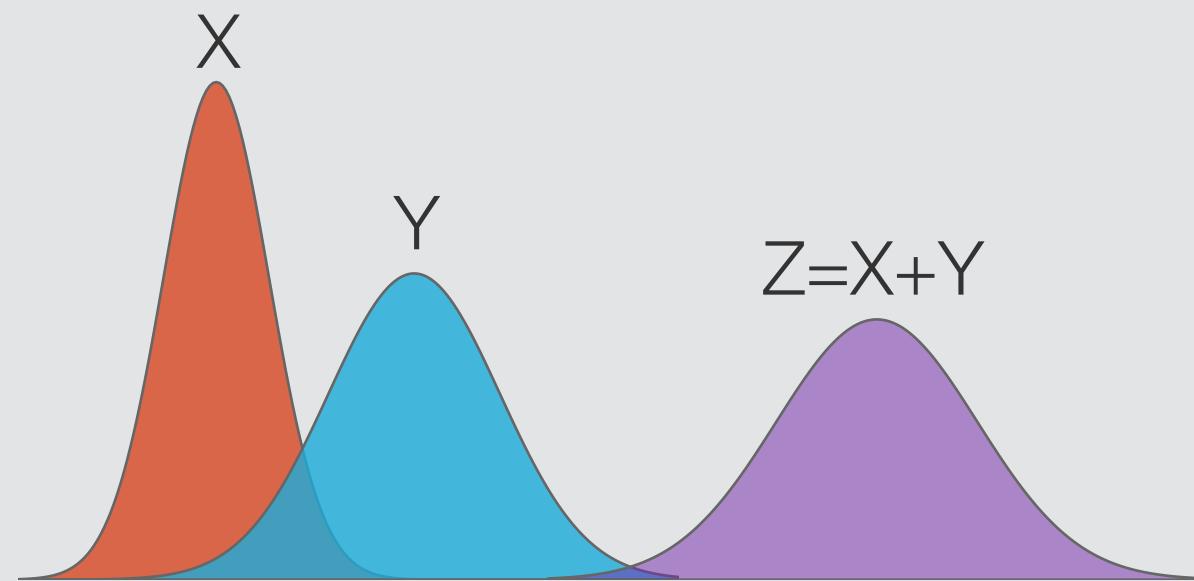
```
Uncertain<double> Speed = Dist / 5;
```

Or more generally, $Z = X + Y$, if X and Y are distributions.



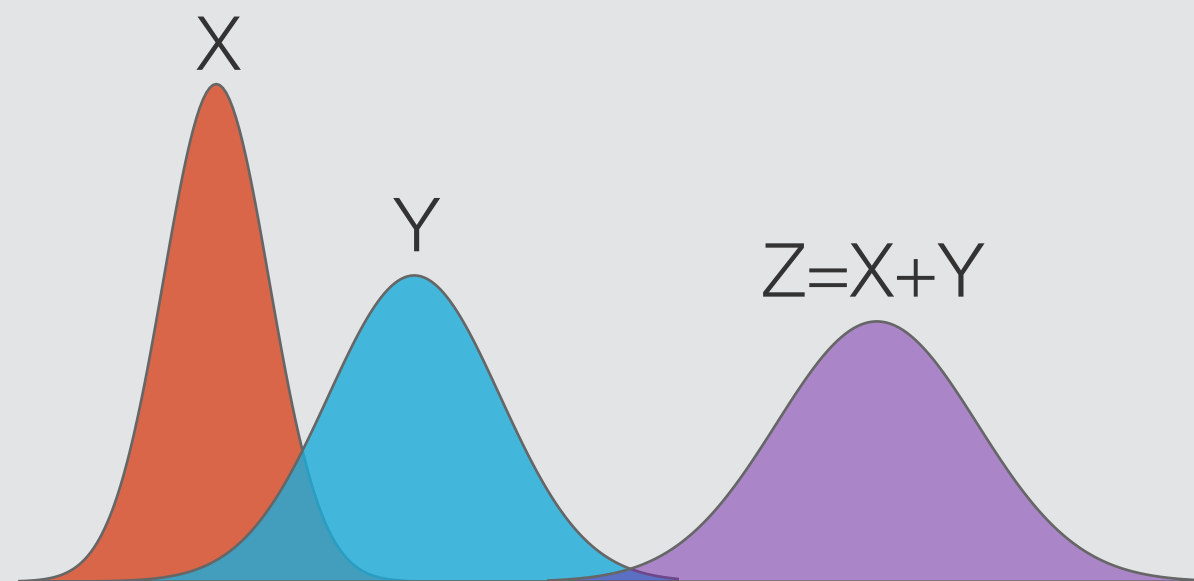
`Uncertain<double> Speed = Dist / 5;`

Or more generally, $Z = X + Y$, if X and Y are distributions.



`Uncertain<double> Speed = Dist / 5;`

Or more generally, $Z = X + Y$, if X and Y are distributions.



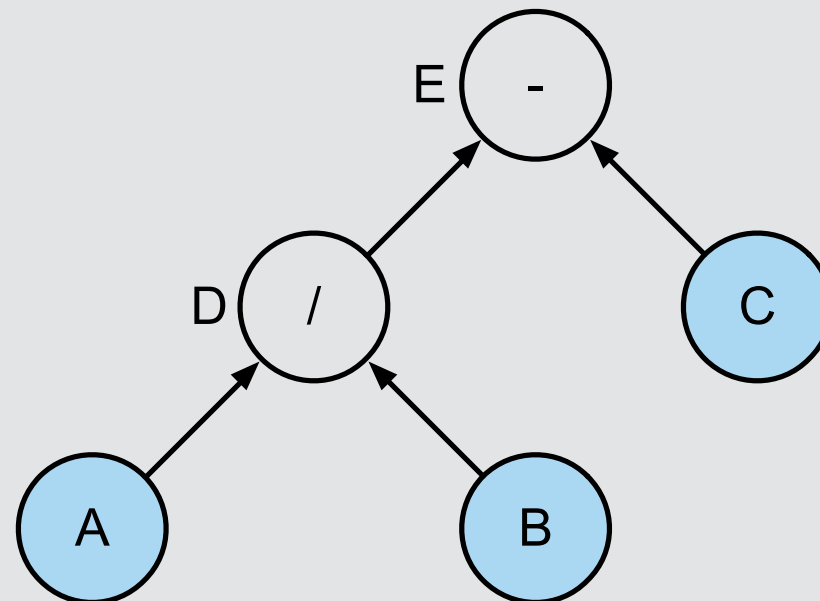
If x is a sample of X
and y is a sample of Y
then $x+y$ is a sample of $X+Y$ *

* if X and Y are independent

$$D = A / B$$

$$E = D - C$$

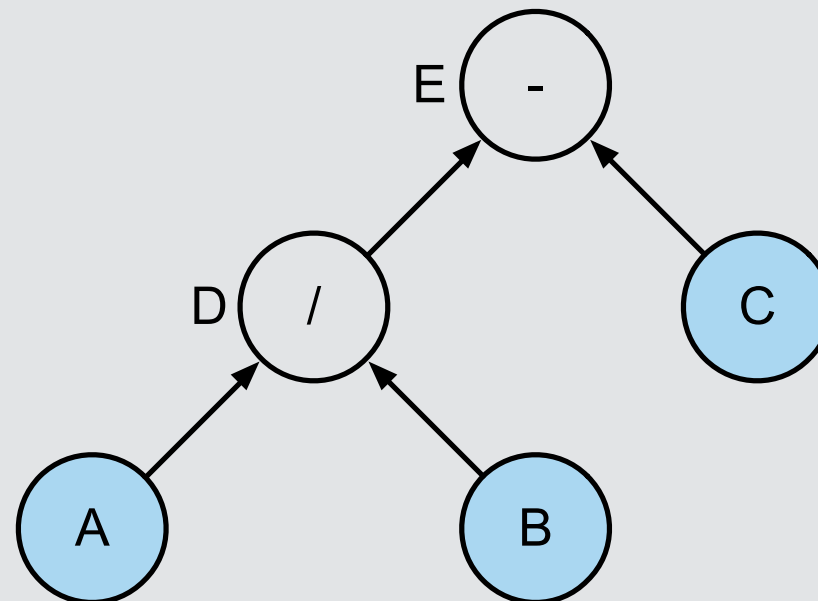
Bayesian network representation:



$$D = A / B$$

$$E = D - C$$

Bayesian network representation:



Sampling function for E recursively samples children.

If x is a sample of X
and y is a sample of Y
then $x+y$ is a sample of $X+Y$ *

* Only if X and Y are independent.

If x is a sample of X
and y is a sample of Y
then $x+y$ is a sample of $X+Y$ *

* Only if X and Y are independent.

$$A = X + Y \quad (X, Y \text{ independent})$$
$$B = A + X$$

If x is a sample of X
and y is a sample of Y
then $x+y$ is a sample of $X+Y$ *

* Only if X and Y are independent.

$$A = X + Y \quad (X, Y \text{ independent})$$
$$B = A + X$$

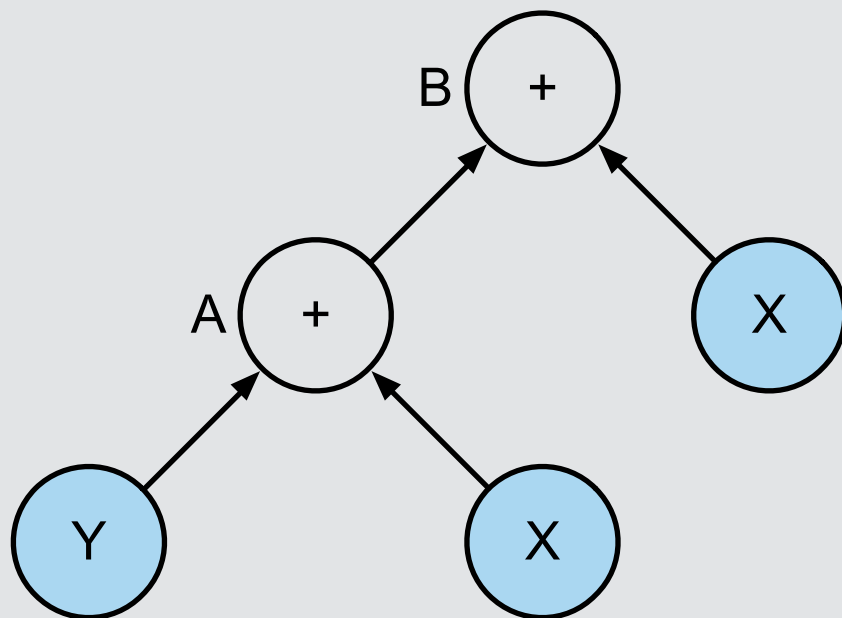
A and B depend on X – not independent!

If x is a sample of X
and y is a sample of Y
then $x+y$ is a sample of $X+Y$ *

* Only if X and Y are independent.

$$A = X + Y \quad (X, Y \text{ independent})$$
$$B = A + X$$

A and B depend on X – not independent!

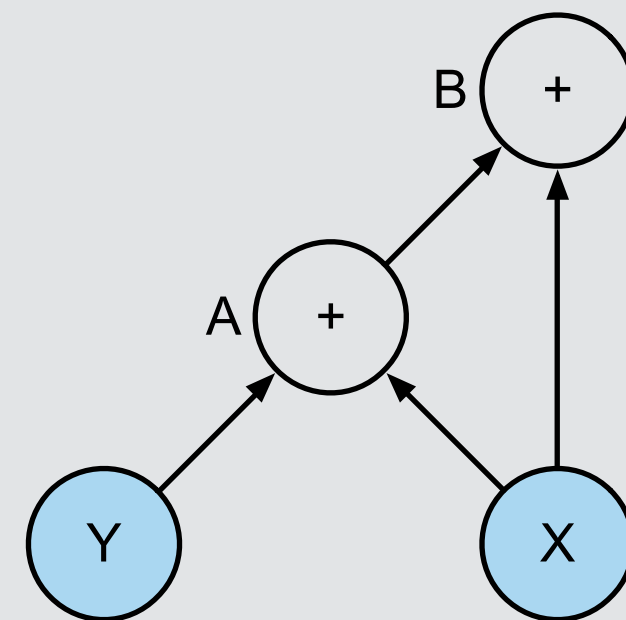
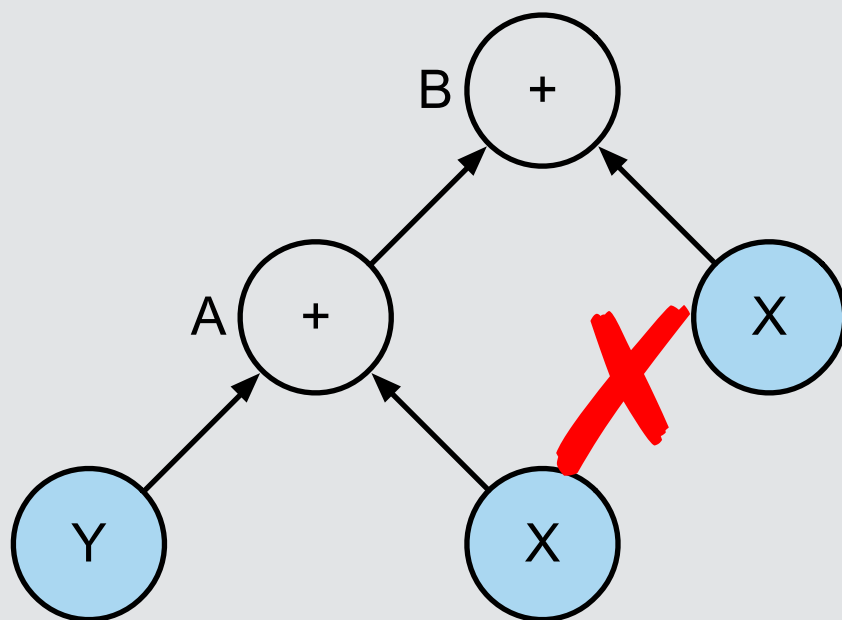


If x is a sample of X
and y is a sample of Y
then $x+y$ is a sample of $X+Y$ *

* Only if X and Y are independent.

$$A = X + Y \quad (X, Y \text{ independent})$$
$$B = A + X$$

A and B depend on X – not independent!

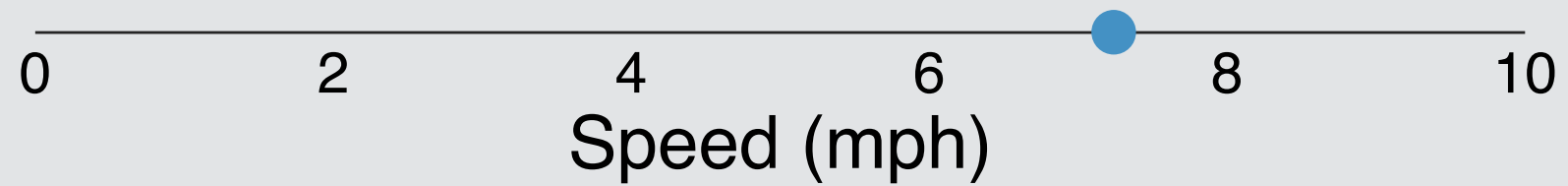



```
if (Speed > 4) print("Great job!");
```

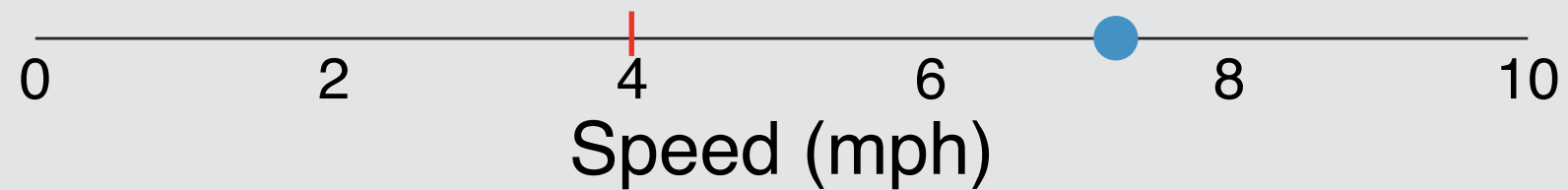
```
if (Speed > 4) print("Great job!");
```



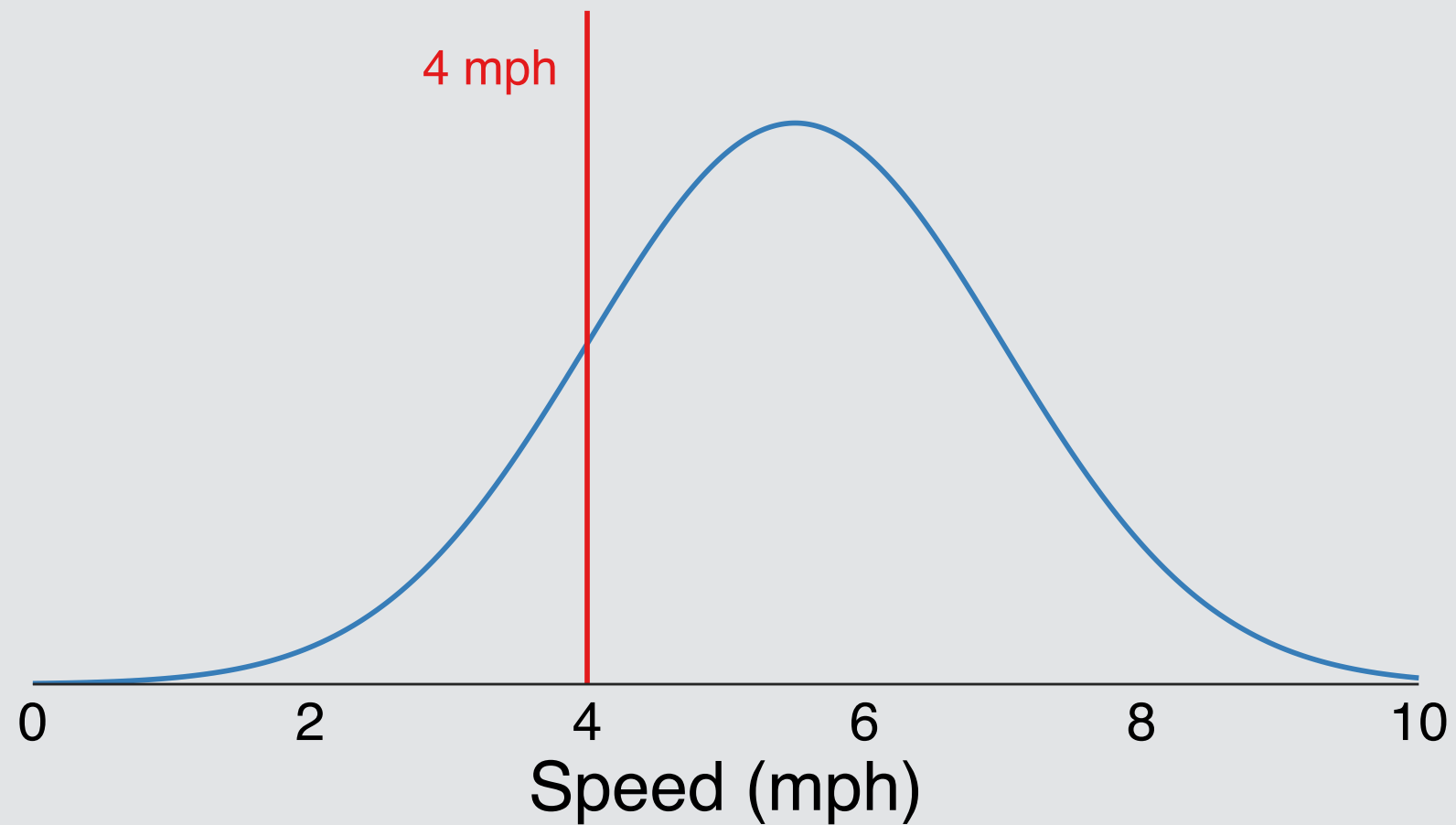
```
if (Speed > 4) print("Great job!");
```



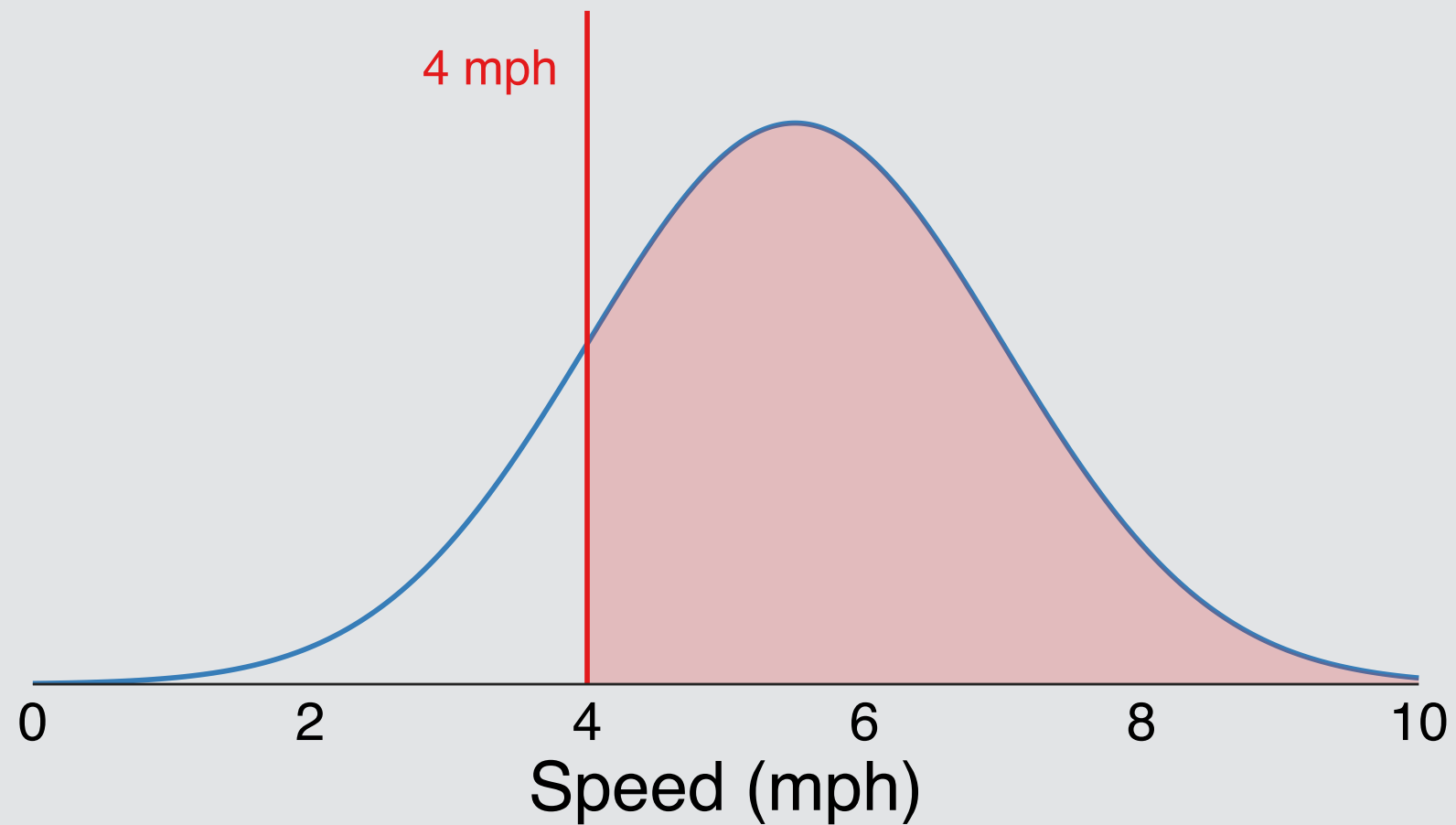
```
if (Speed > 4) print("Great job!");
```



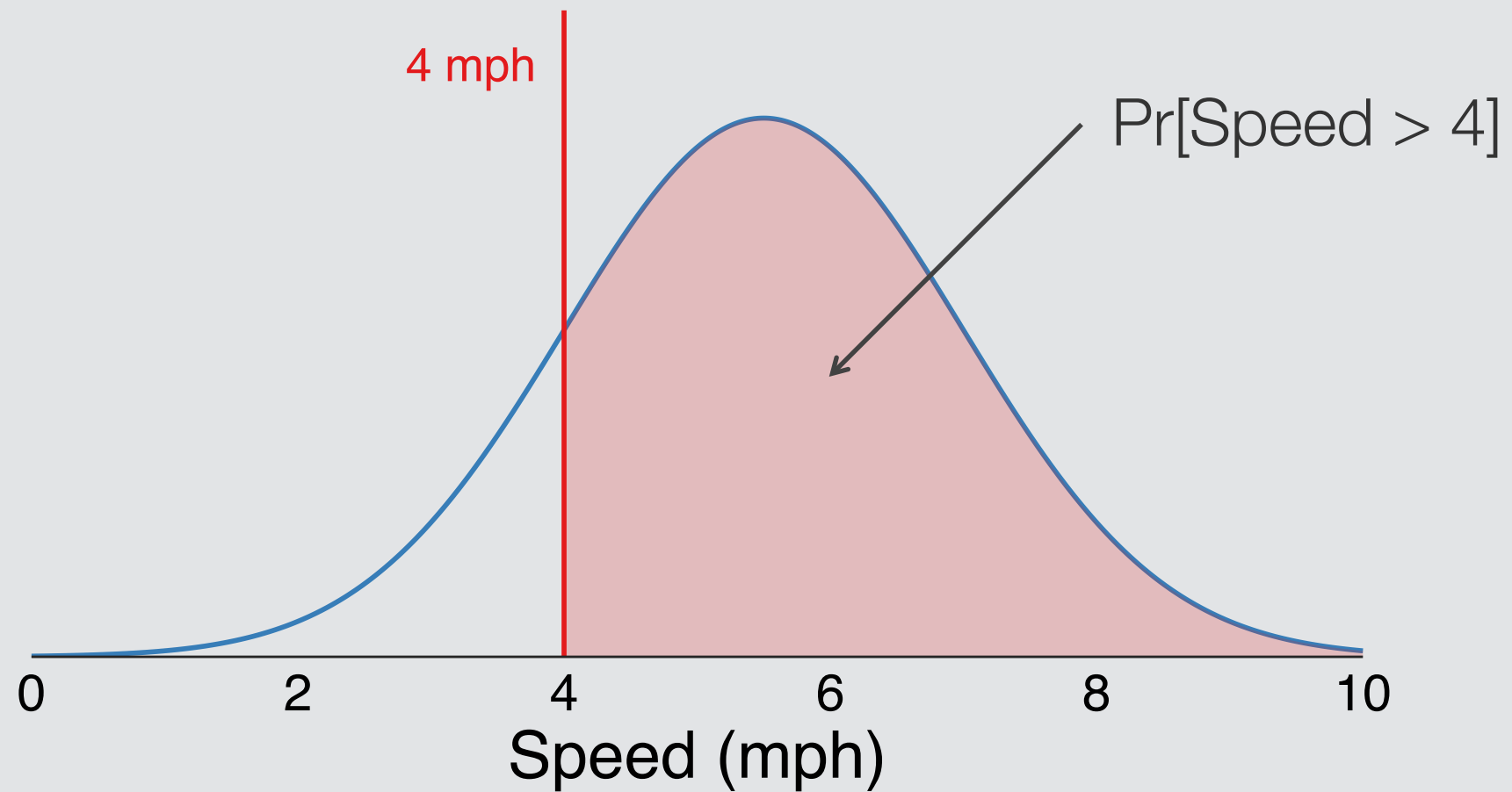
```
if (Speed > 4) print("Great job!");
```



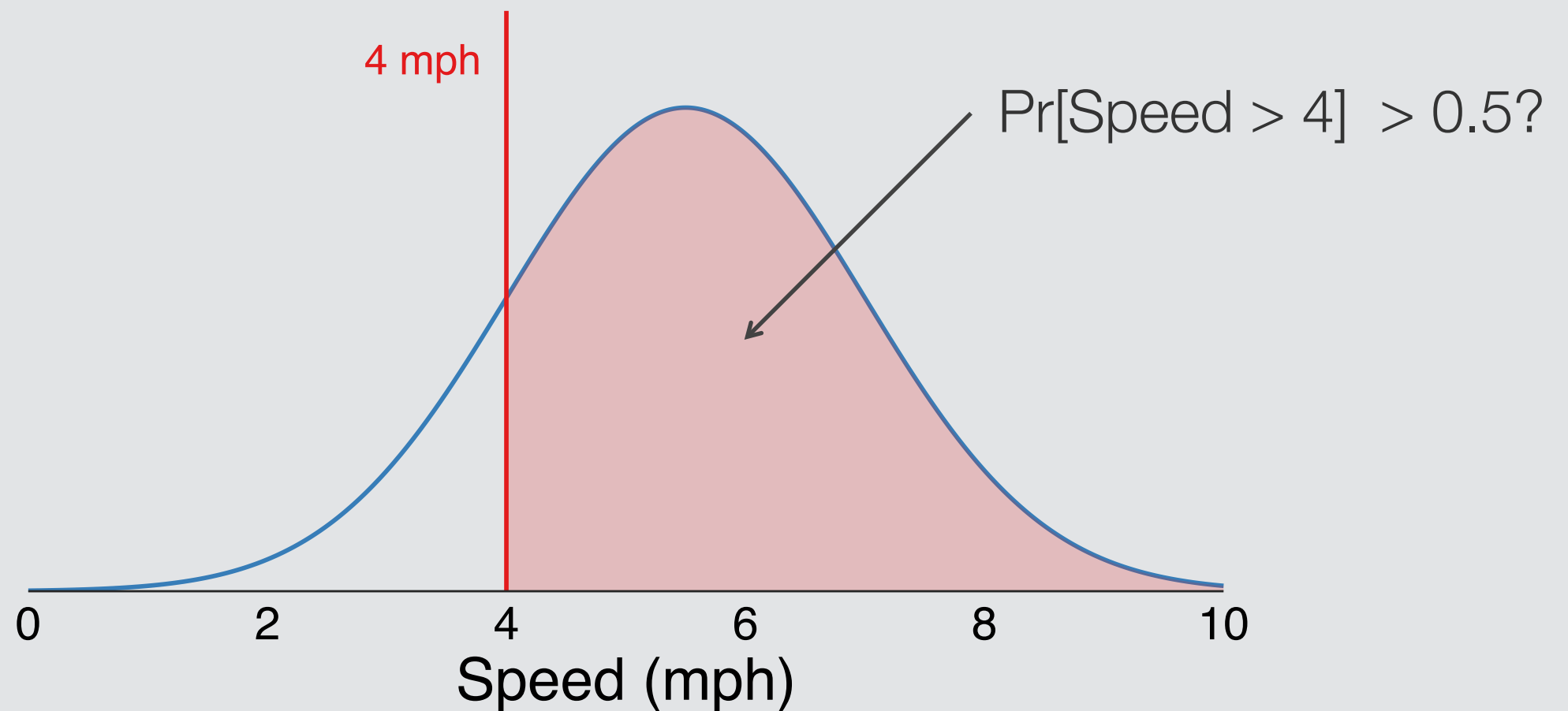
```
if (Speed > 4) print("Great job!");
```



```
if (Speed > 4) print("Great job!");
```

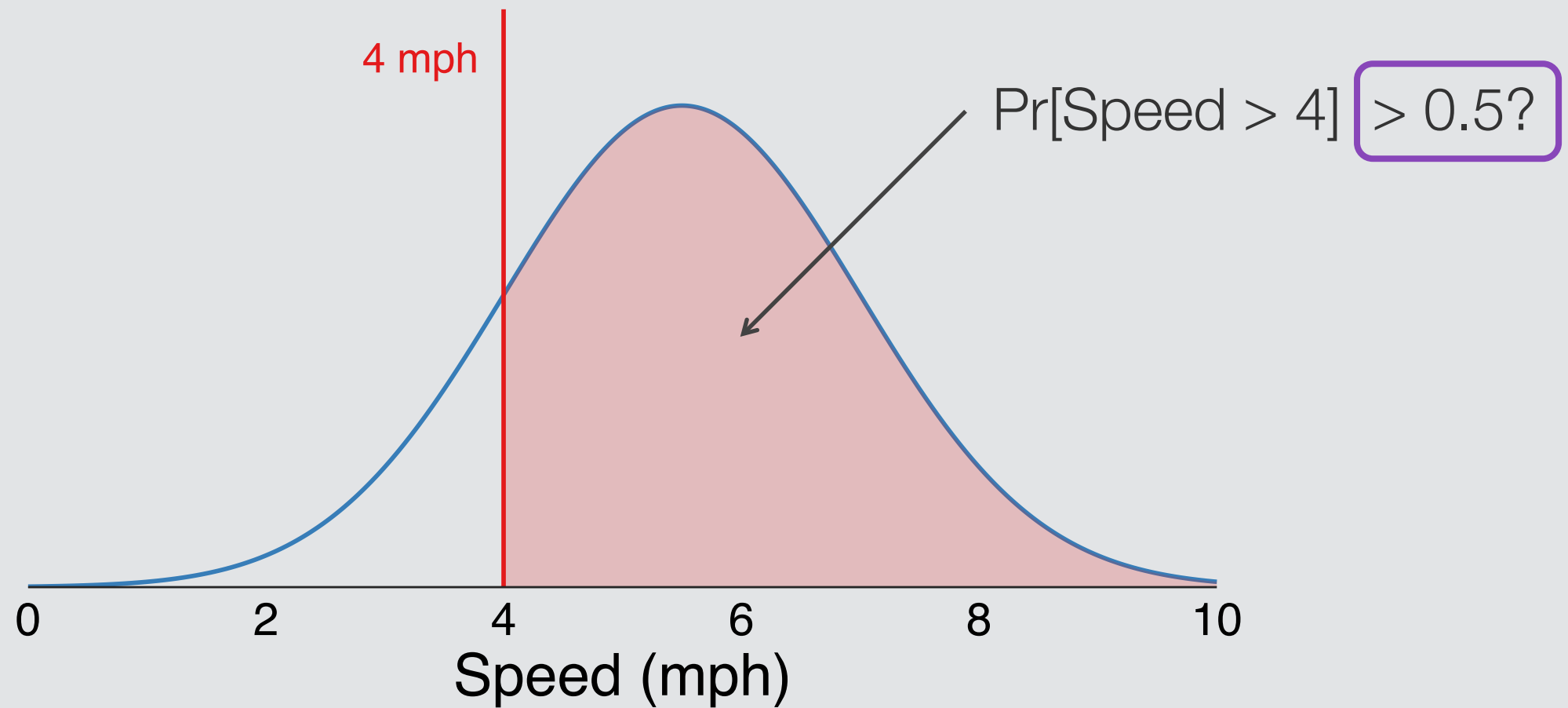


```
if (Speed > 4) print("Great job!");
```



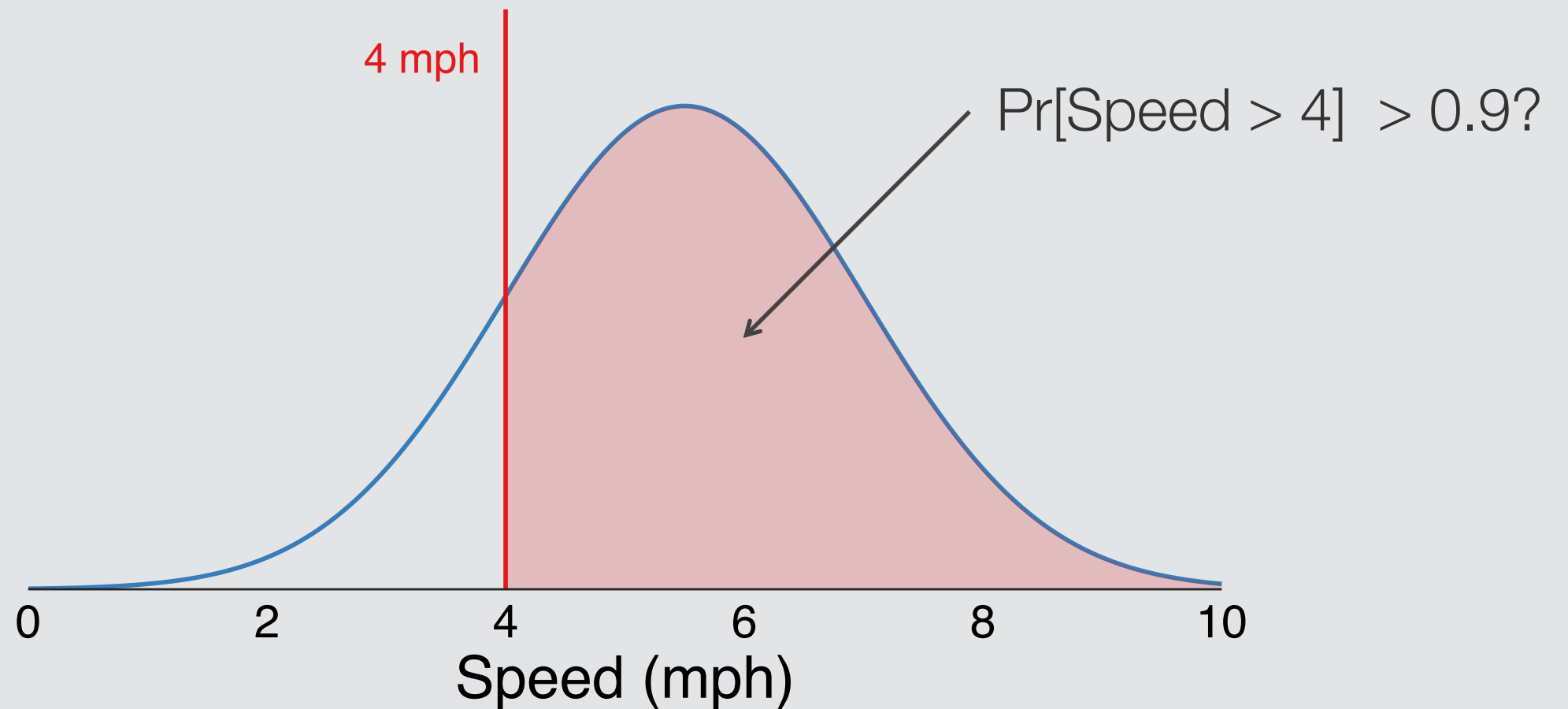
More likely than not that Speed > 4?


```
if (Speed > 4) print("Great job!");
```

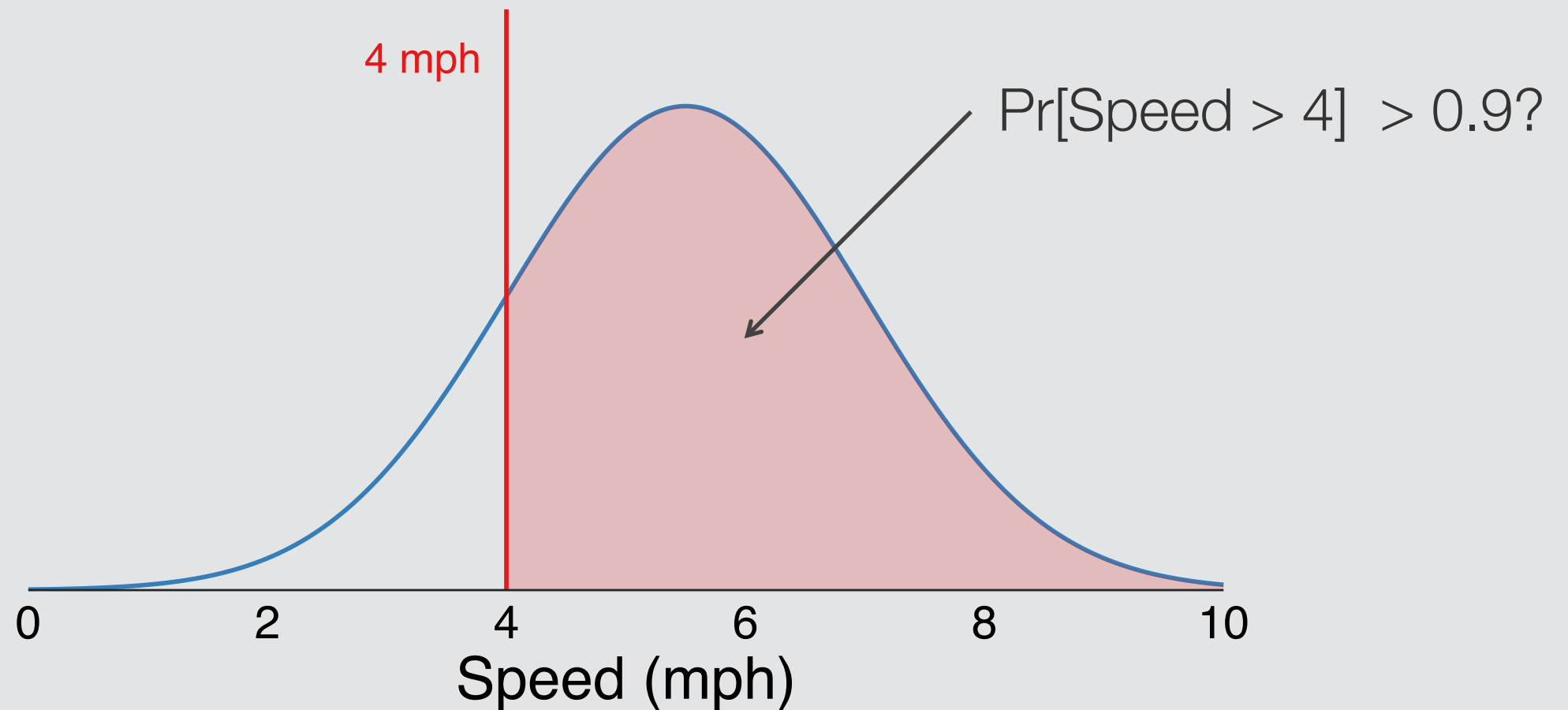


More likely than not that Speed > 4?

```
if (Speed > 4).Pr(0.9) print("Great job!");
```



```
if (Speed > 4).Pr(0.9) print("Great job!");
```



At least 90% likely that Speed > 4?

```
if (Speed > 4).Pr(0.9) print("Great job!");
```

$\text{Pr}[\text{Speed} > 4] > 0.9$

```
if (Speed > 4).Pr(0.9) print("Great job!");
```

$\text{Pr}[\underbrace{\text{Speed} > 4}] > 0.9$
approximate!

```
if (Speed > 4).Pr(0.9) print("Great job!");
```

$\underbrace{\text{Pr}[\text{Speed} > 4]}_{\text{approximate!}} > 0.9$


```
if (Speed > 4).Pr(0.9) print("Great job!");
```

null hypothesis $H_0: \Pr[\text{Speed} > 4] \leq 0.9$

$\underbrace{\Pr[\text{Speed} > 4]}_{\text{approximate!}} > 0.9$

approximate!

```
if (Speed > 4).Pr(0.9) print("Great job!");
```

null hypothesis $H_0: \Pr[\text{Speed} > 4] \leq 0.9$

alternate hypothesis $H_A: \underbrace{\Pr[\text{Speed} > 4]}_{\text{approximate!}} > 0.9$

```
if (Speed > 4).Pr(0.9) print("Great job!");
```

null hypothesis $H_0: \Pr[\text{Speed} > 4] \leq 0.9$

alternate hypothesis $H_A: \underbrace{\Pr[\text{Speed} > 4]}_{\text{approximate!}} > 0.9$

How many samples?

```
if (Speed > 4).Pr(0.9) print("Great job!");
```

null hypothesis $H_0: \Pr[\text{Speed} > 4] \leq 0.9$

alternate hypothesis $H_A: \underbrace{\Pr[\text{Speed} > 4]}_{\text{approximate!}} > 0.9$

How many samples? Too many = too slow
Too few = too noisy

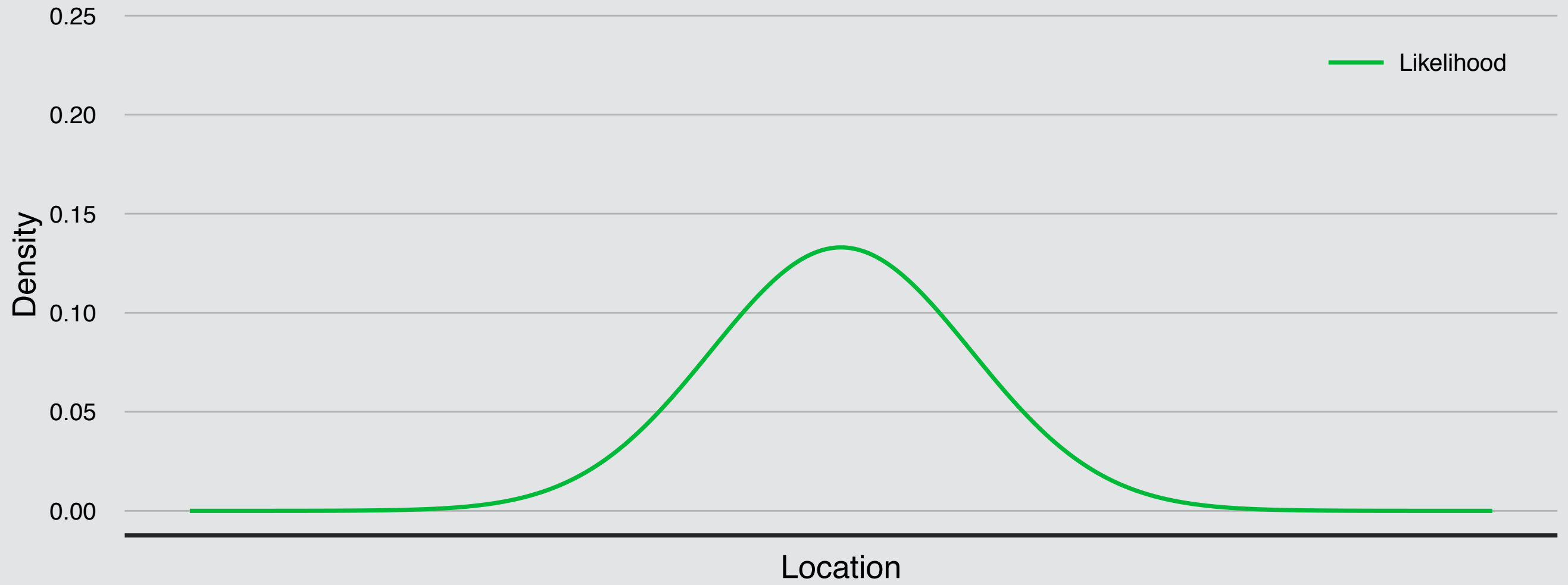
```
if (Speed > 4).Pr(0.9) print("Great job!");
```

null hypothesis $H_0: \Pr[\text{Speed} > 4] \leq 0.9$

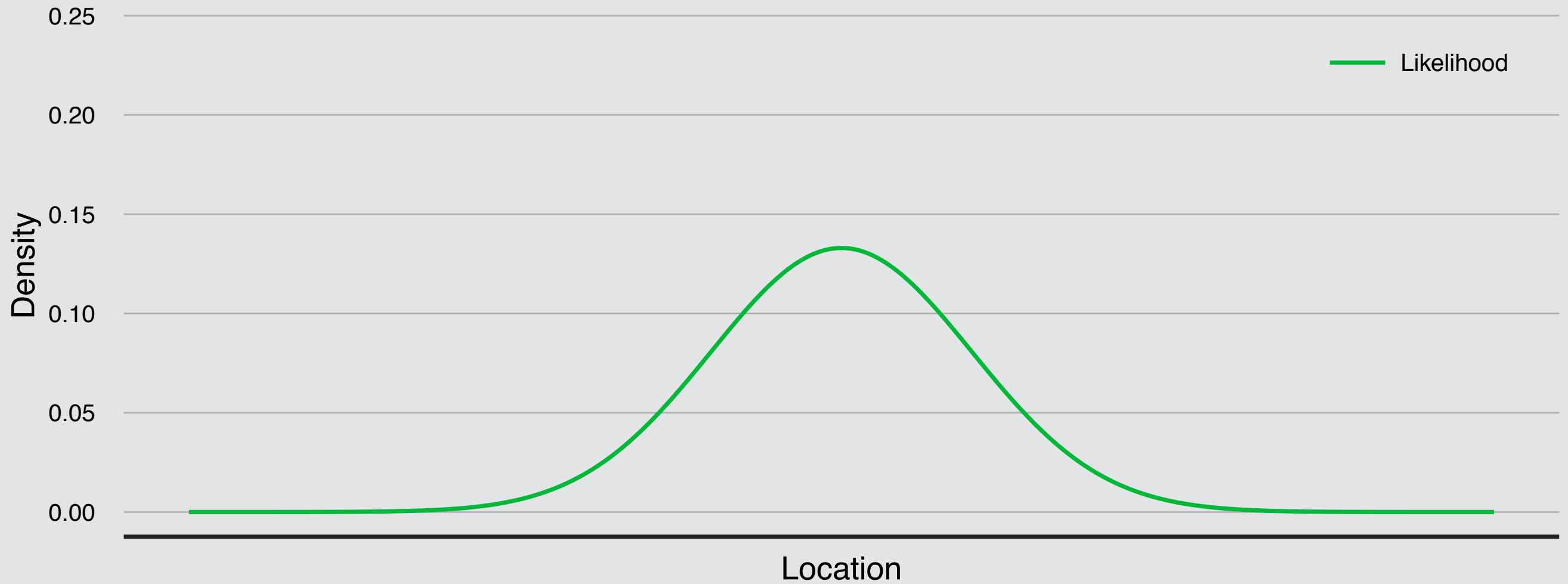
alternate hypothesis $H_A: \underbrace{\Pr[\text{Speed} > 4]}_{\text{approximate!}} > 0.9$

How many samples? Too many = too slow
Too few = too noisy

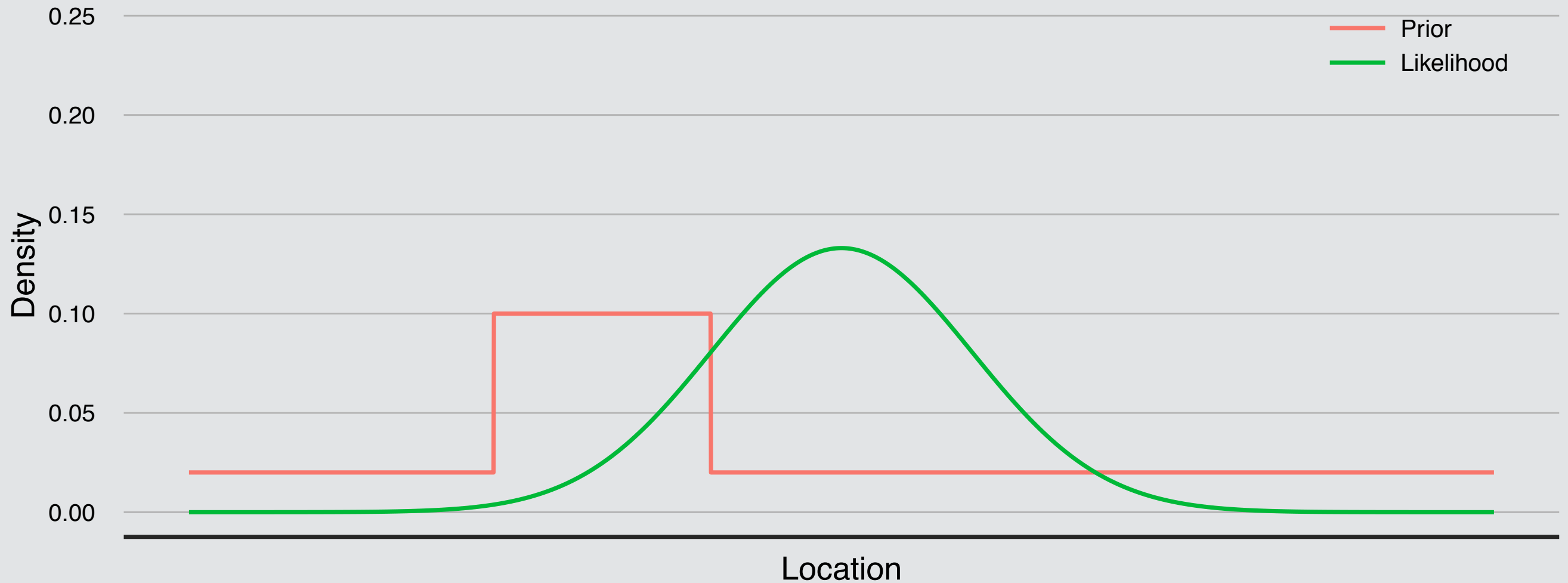
Sequential sampling: sample size depends on progress



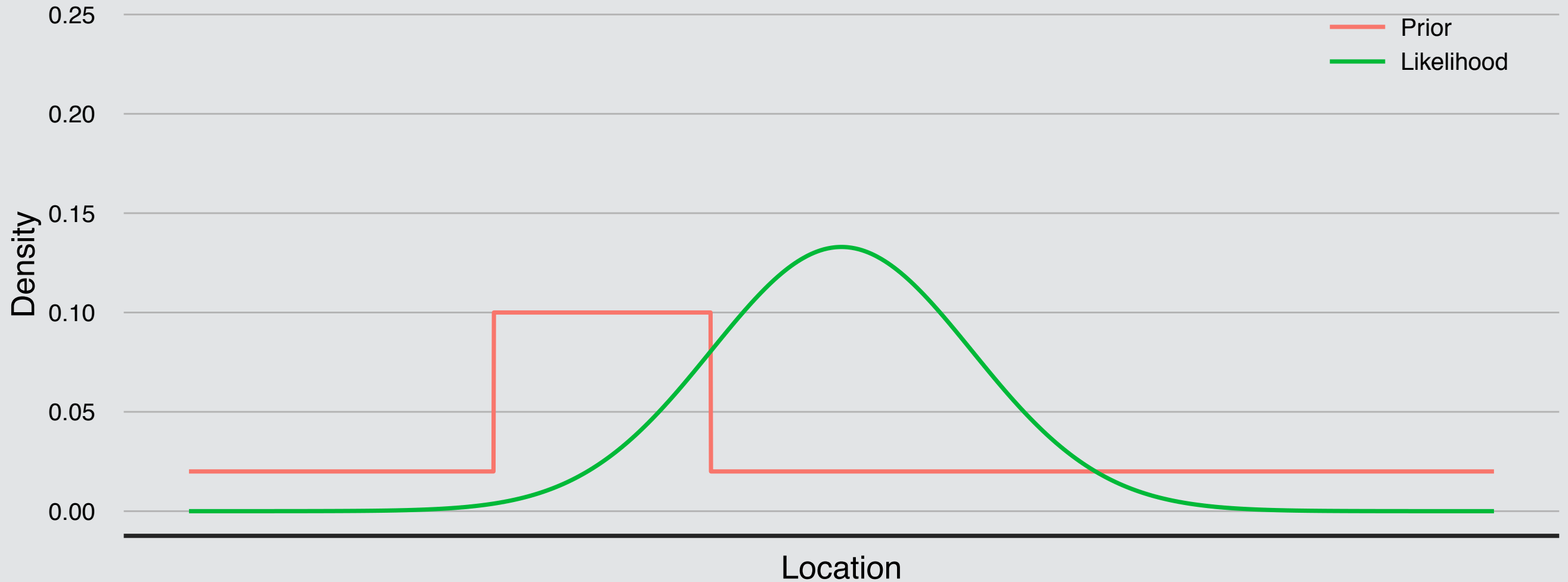
Incorporate domain knowledge: “I’m on a road”



Incorporate domain knowledge: “I’m on a road”



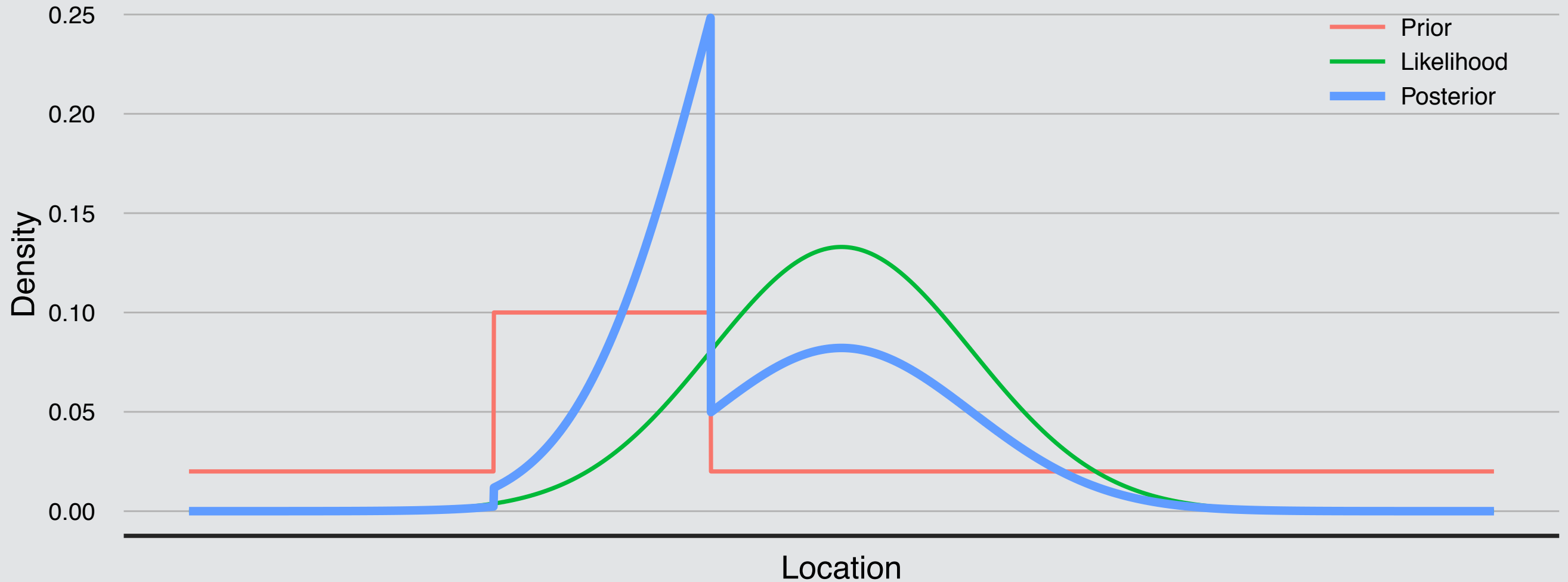
Incorporate domain knowledge: “I’m on a road”



posterior likelihood prior

$$\Pr[H|E] = \frac{\Pr[E|H] \Pr[H]}{\Pr[E]}$$

Incorporate domain knowledge: “I’m on a road”



posterior likelihood prior

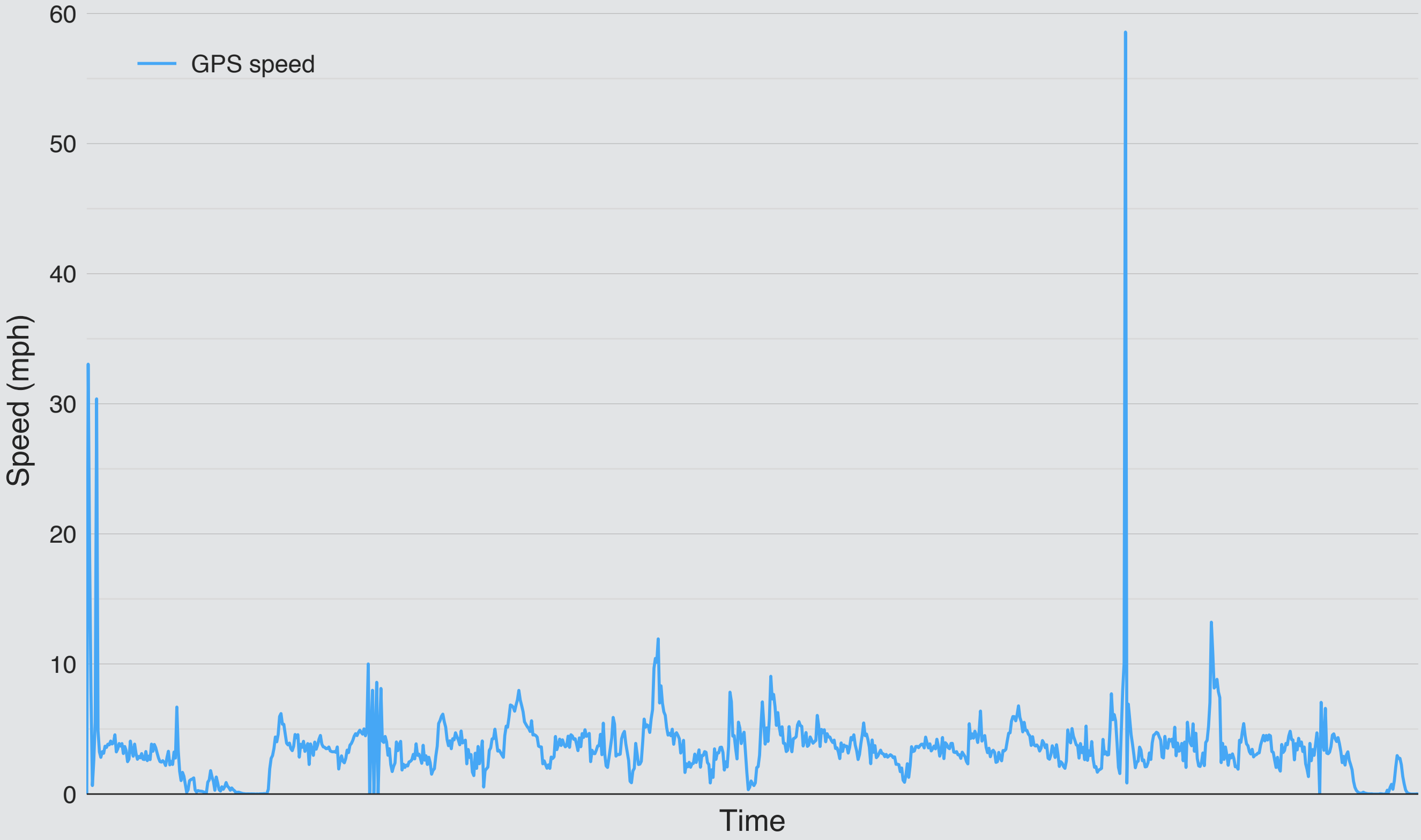
$$\Pr[H|E] = \frac{\Pr[E|H] \Pr[H]}{\Pr[E]}$$

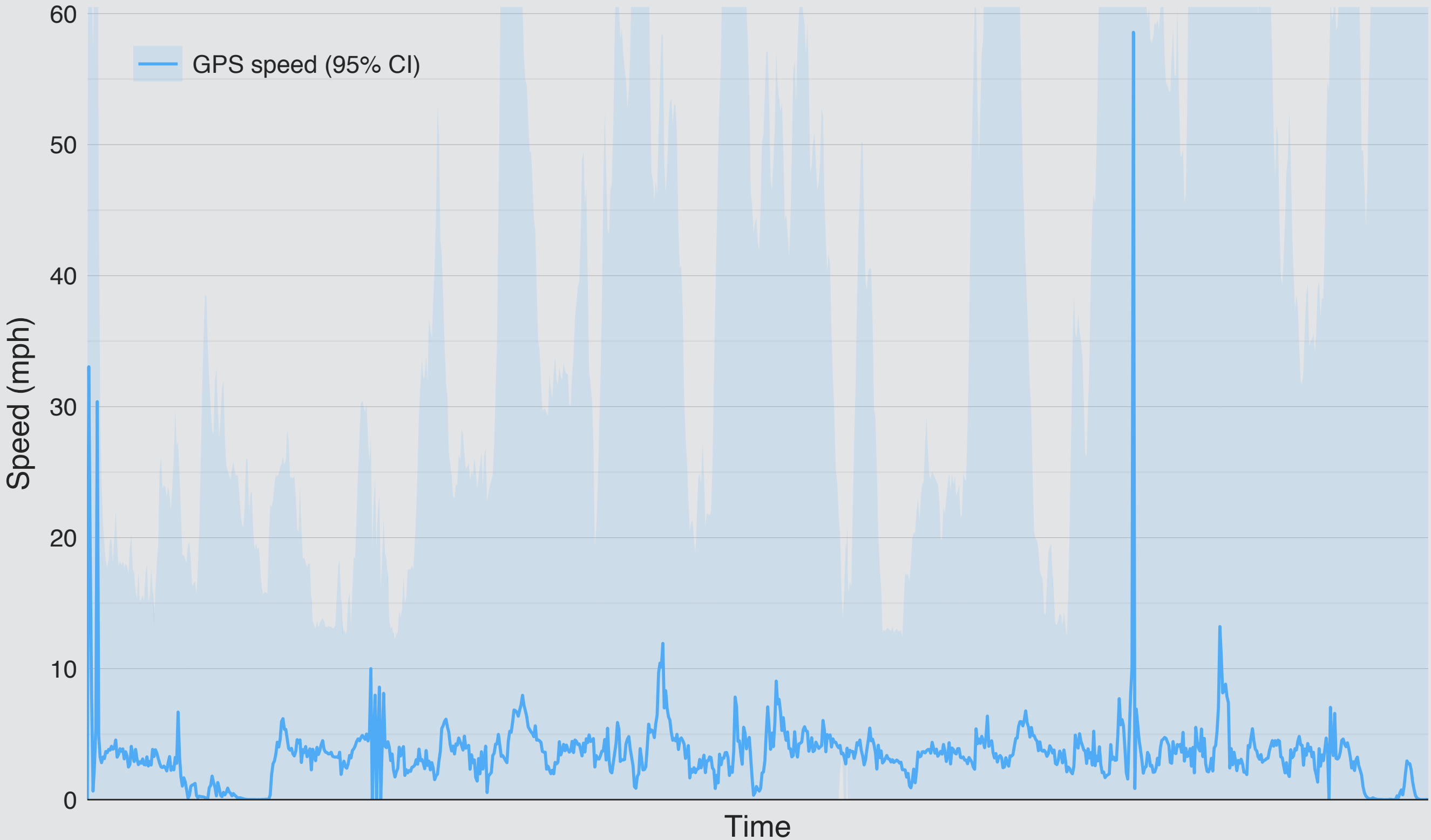
Case studies

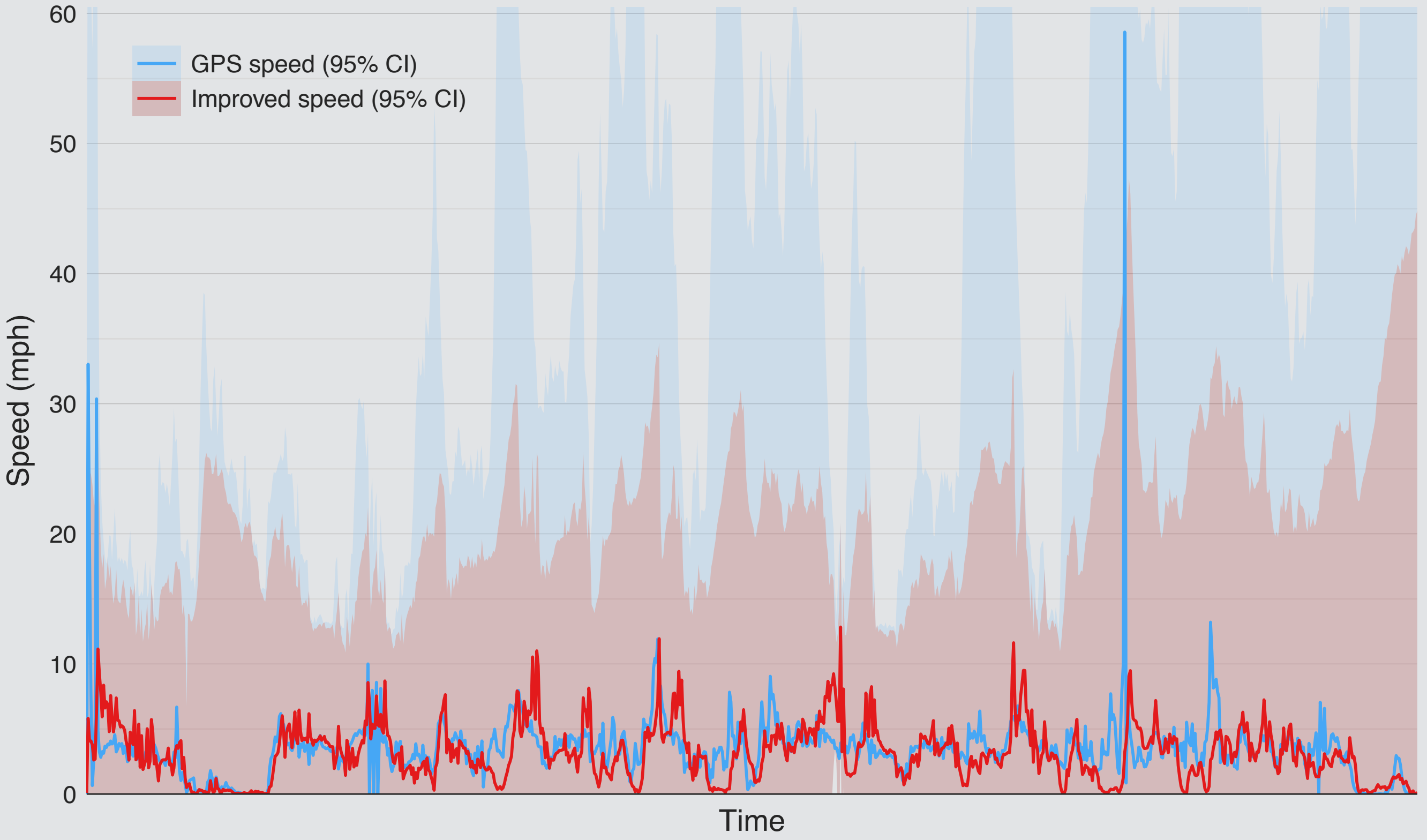
Smartphone GPS sensors

Noisy Game of Life (see the paper)

Neural networks/approximate computing







What is the gradient at pixel p ?

$\text{Sobel1}(p)$

3.4% average error

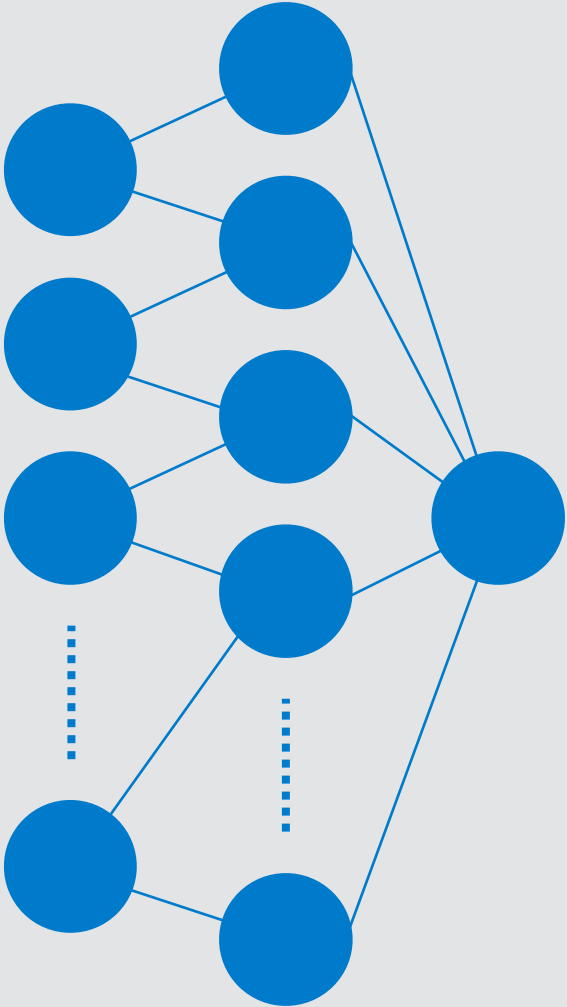
Is there an edge at pixel p ?

$\text{Sobel1}(p) > 0.1$

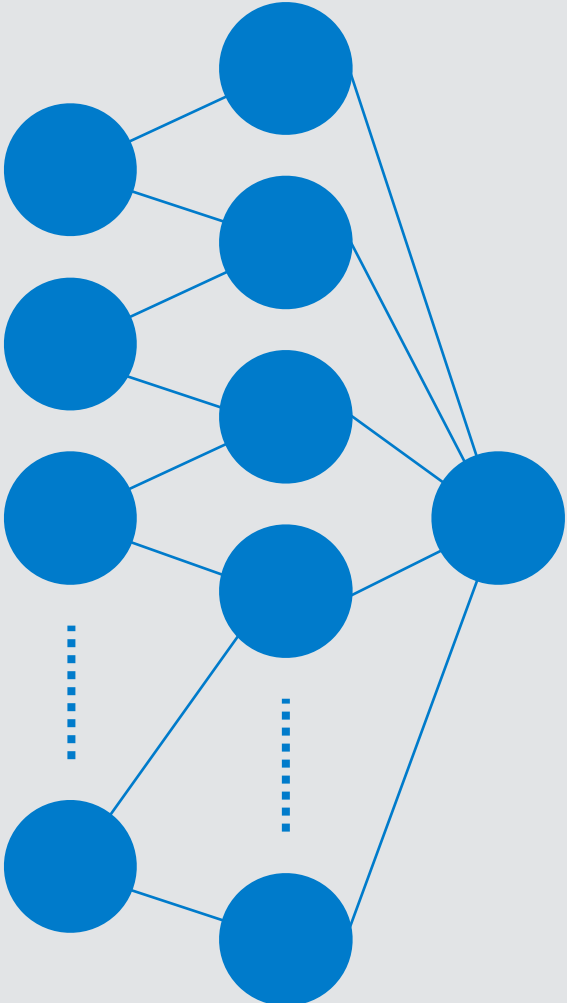
36% false positives!

single
input

single
input

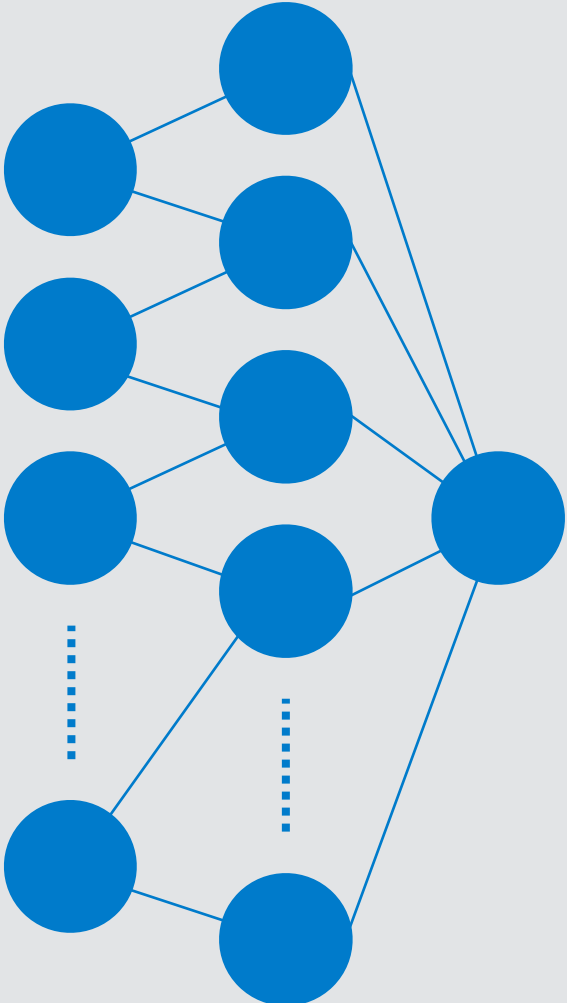


single
input



single
output

single
input

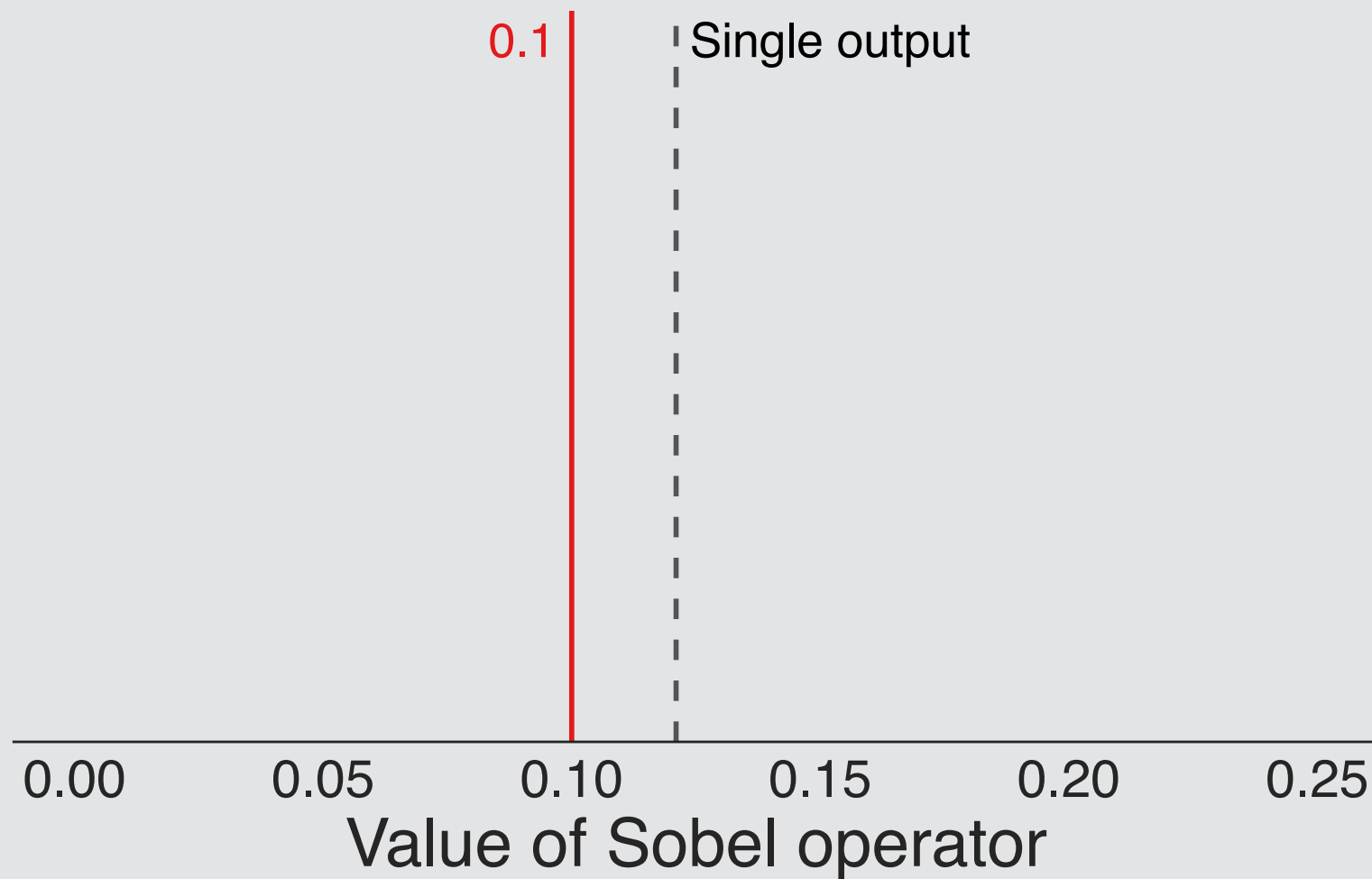


approximate
output

Is there an edge at pixel p ?

$\text{Sobel}(p) > 0.1$

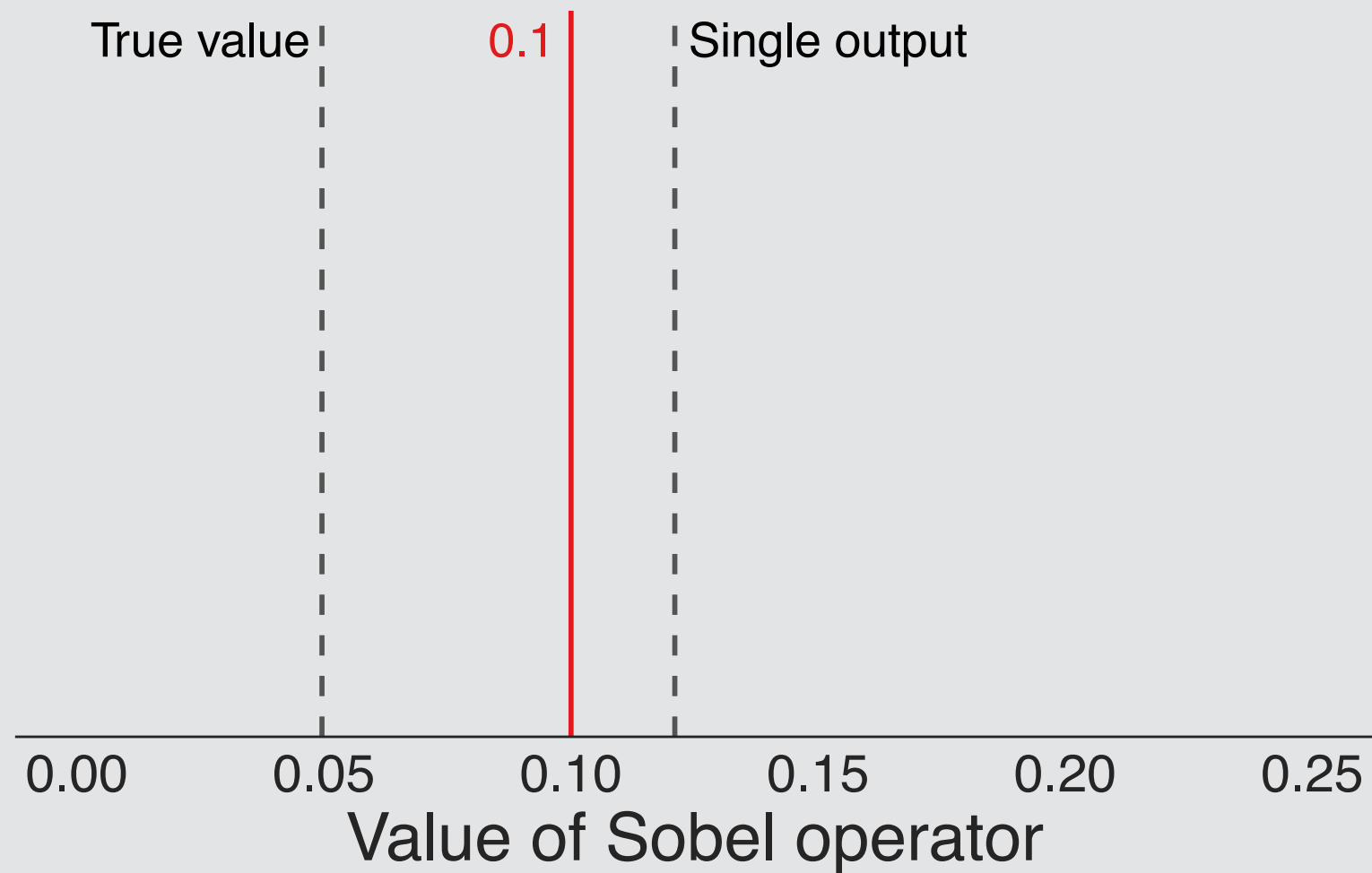
36% false positives!



Is there an edge at pixel p ?

$\text{Sobel1}(p) > 0.1$

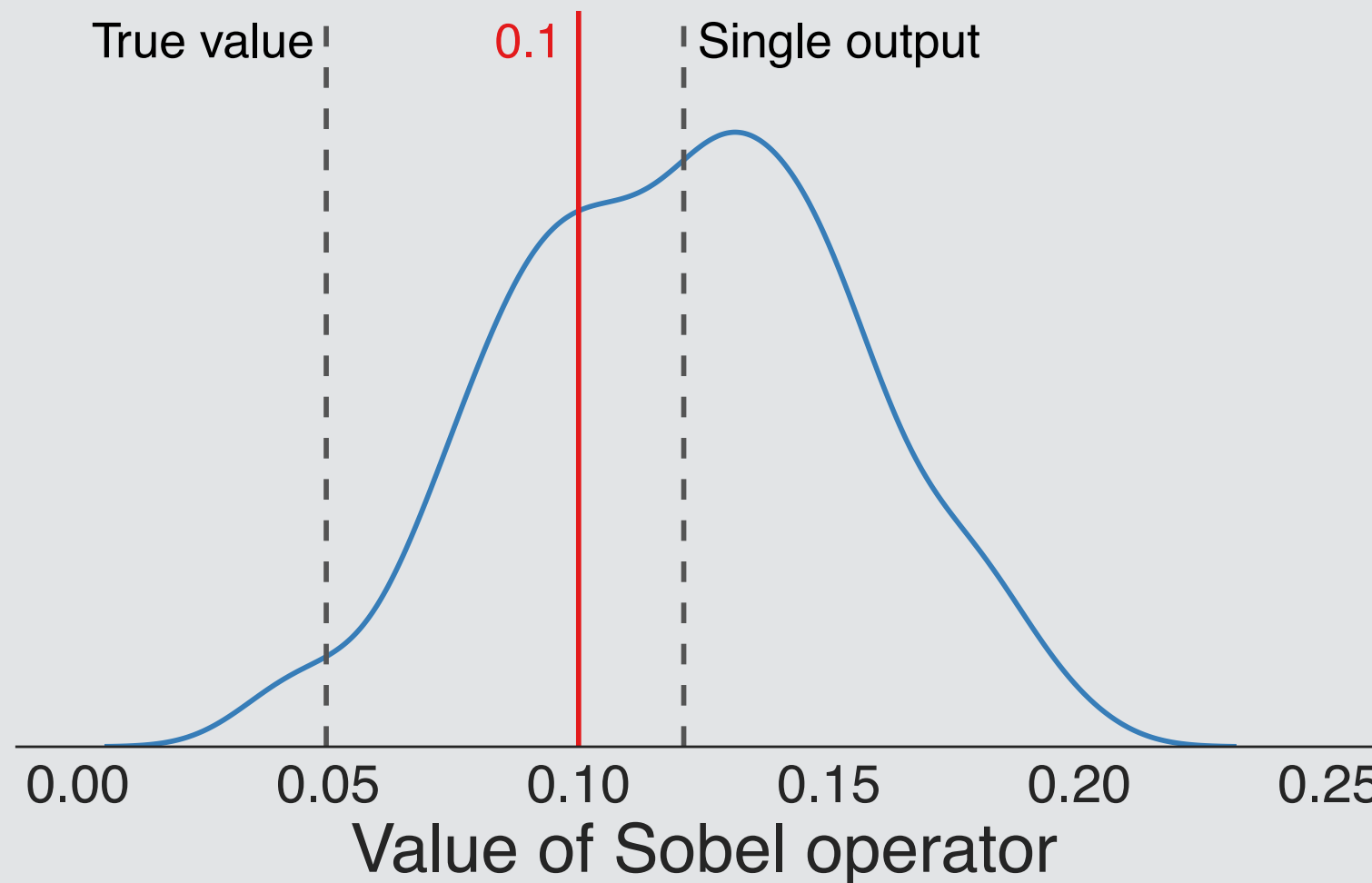
36% false positives!



Is there an edge at pixel p ?

$\text{Sobel1}(p) > 0.1$

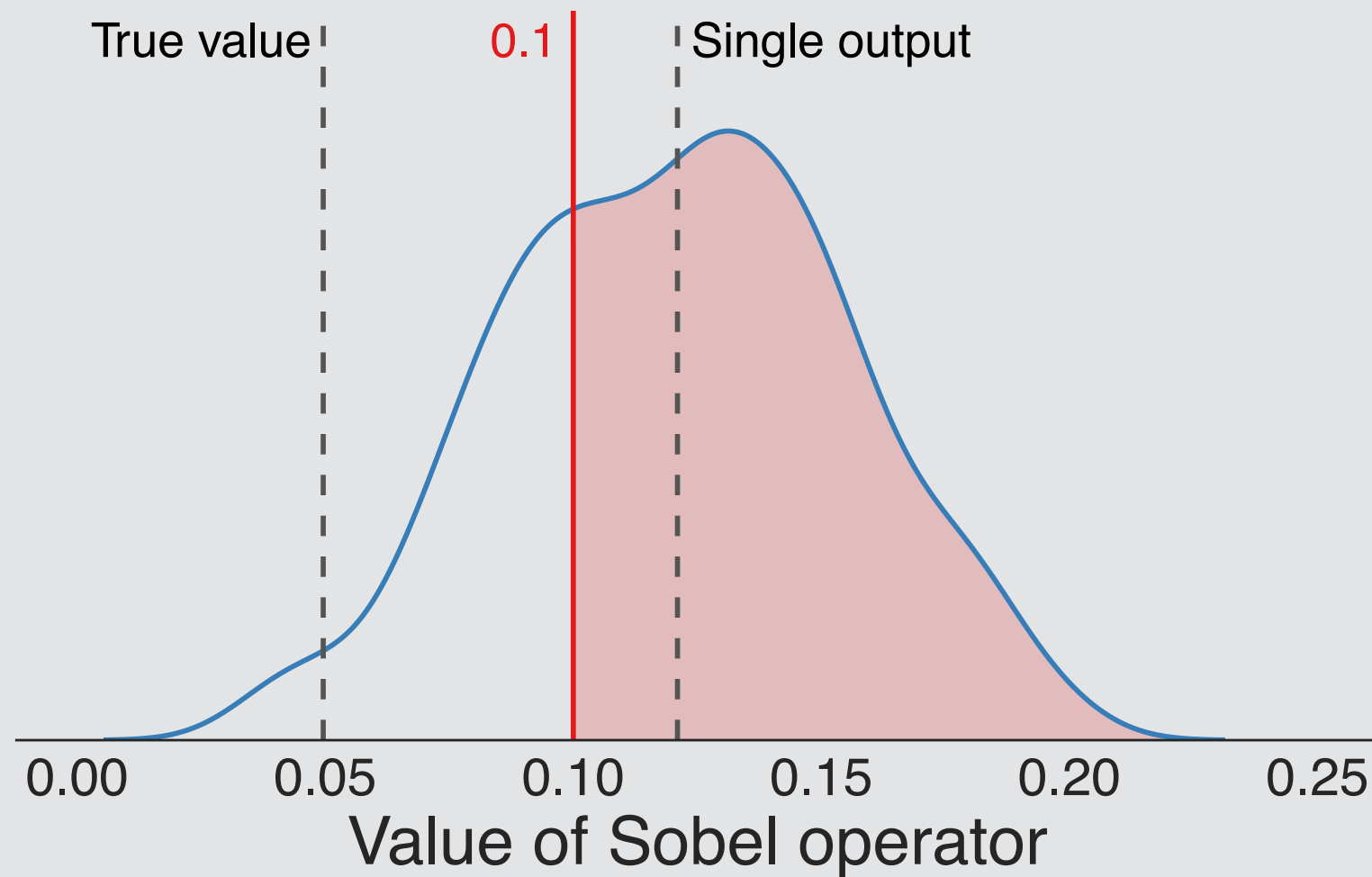
36% false positives!



Is there an edge at pixel p ?

$\text{Sobel1}(p) > 0.1$

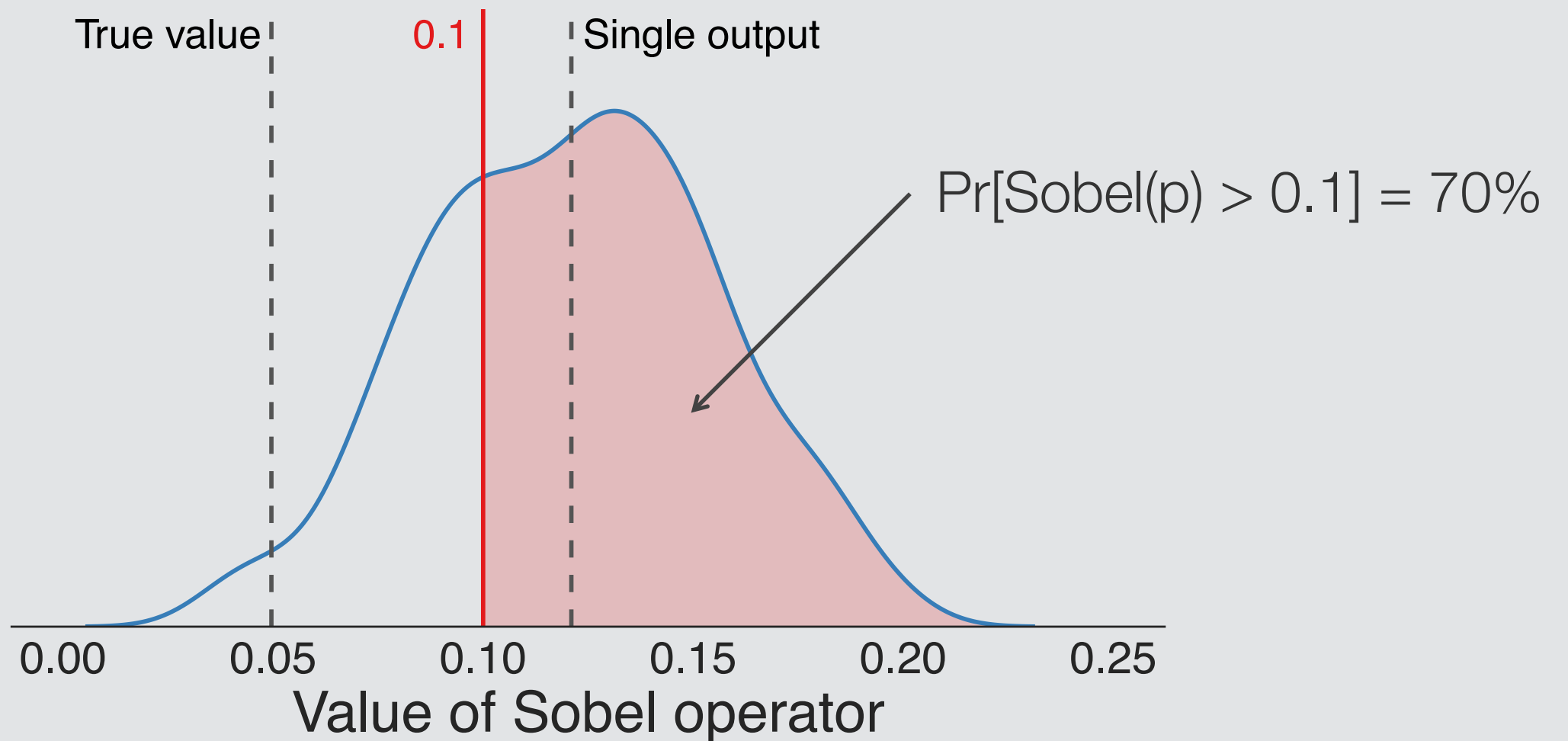
36% false positives!

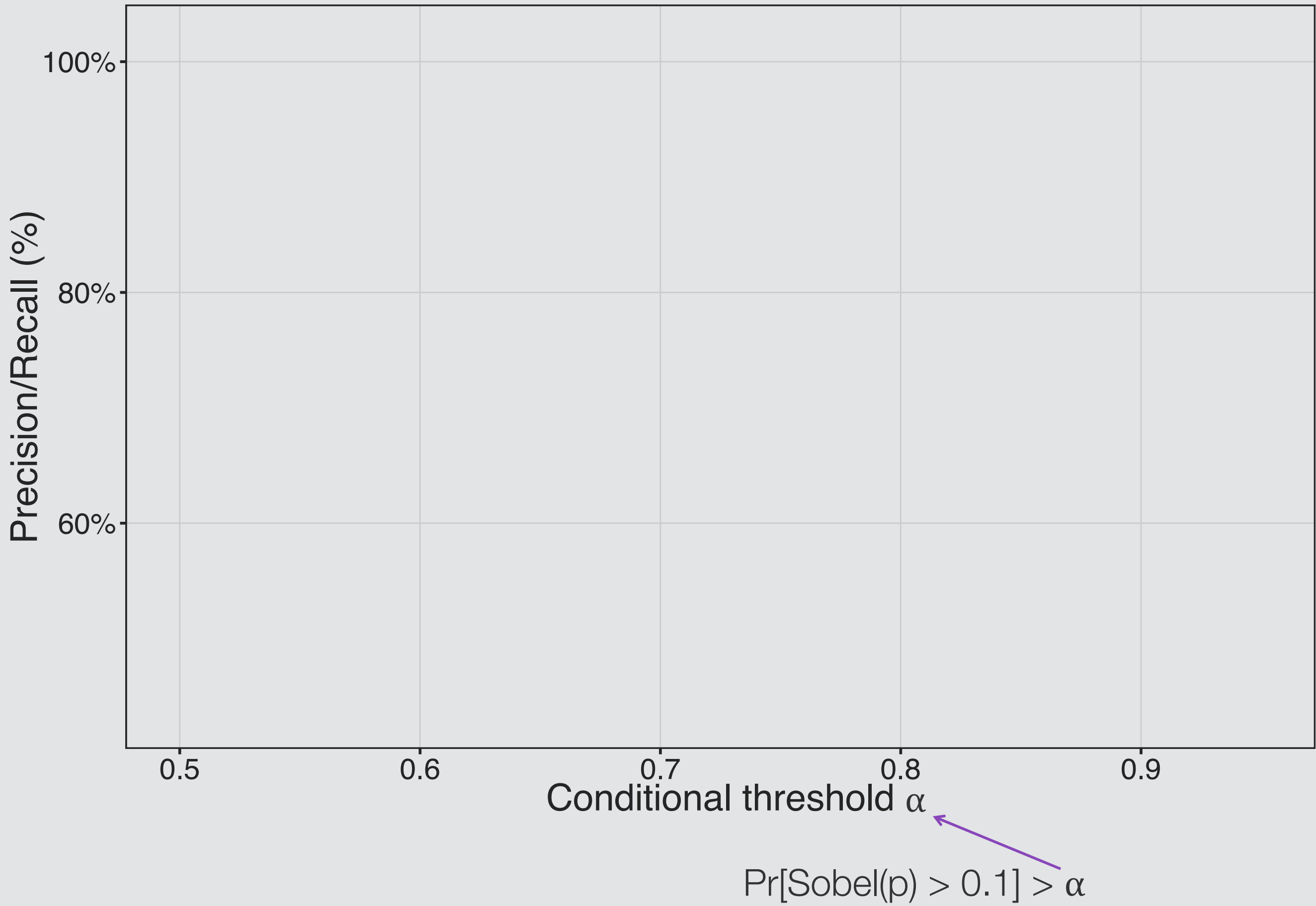


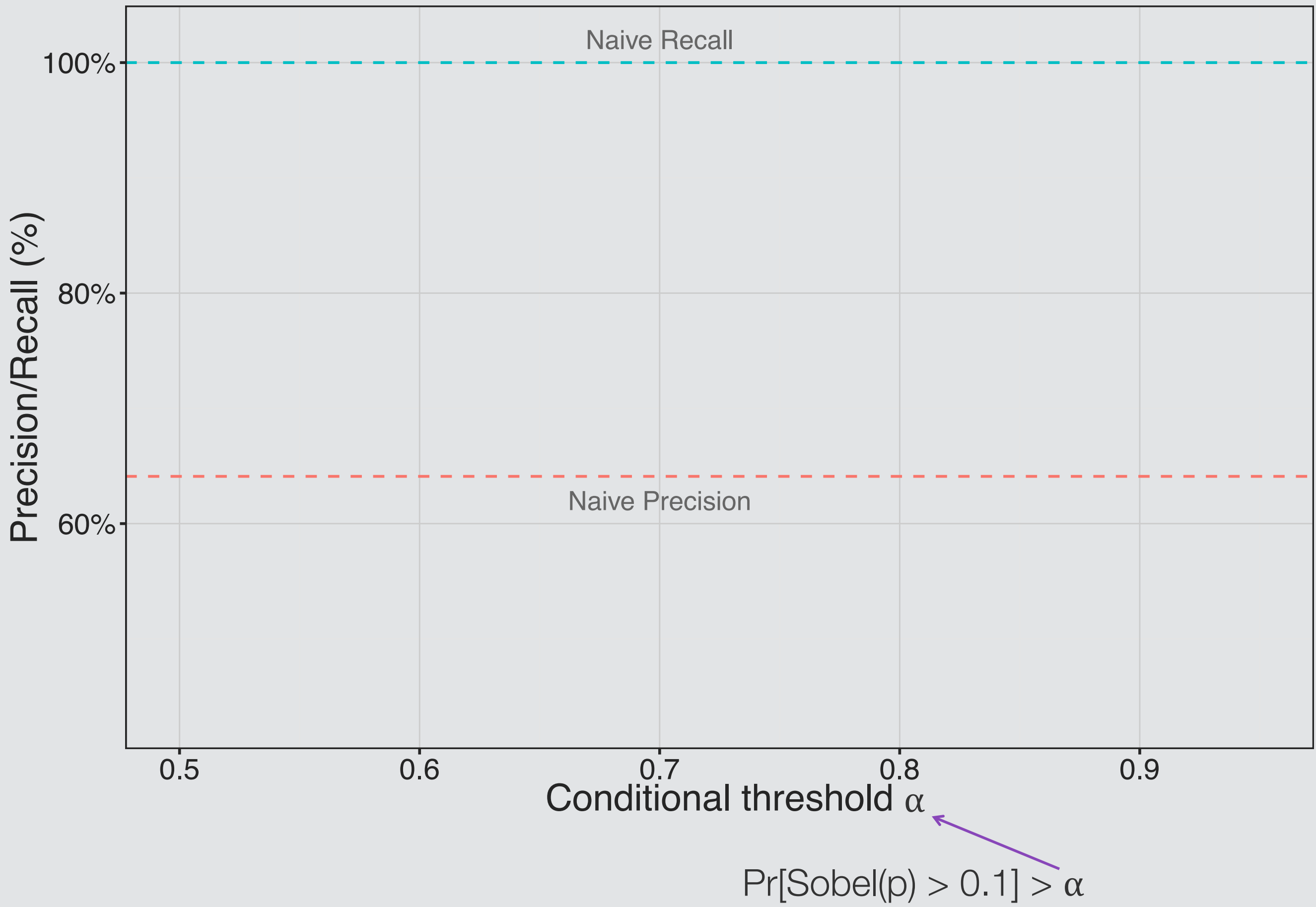
Is there an edge at pixel p ?

$\text{Sobel}(p) > 0.1$

36% false positives!





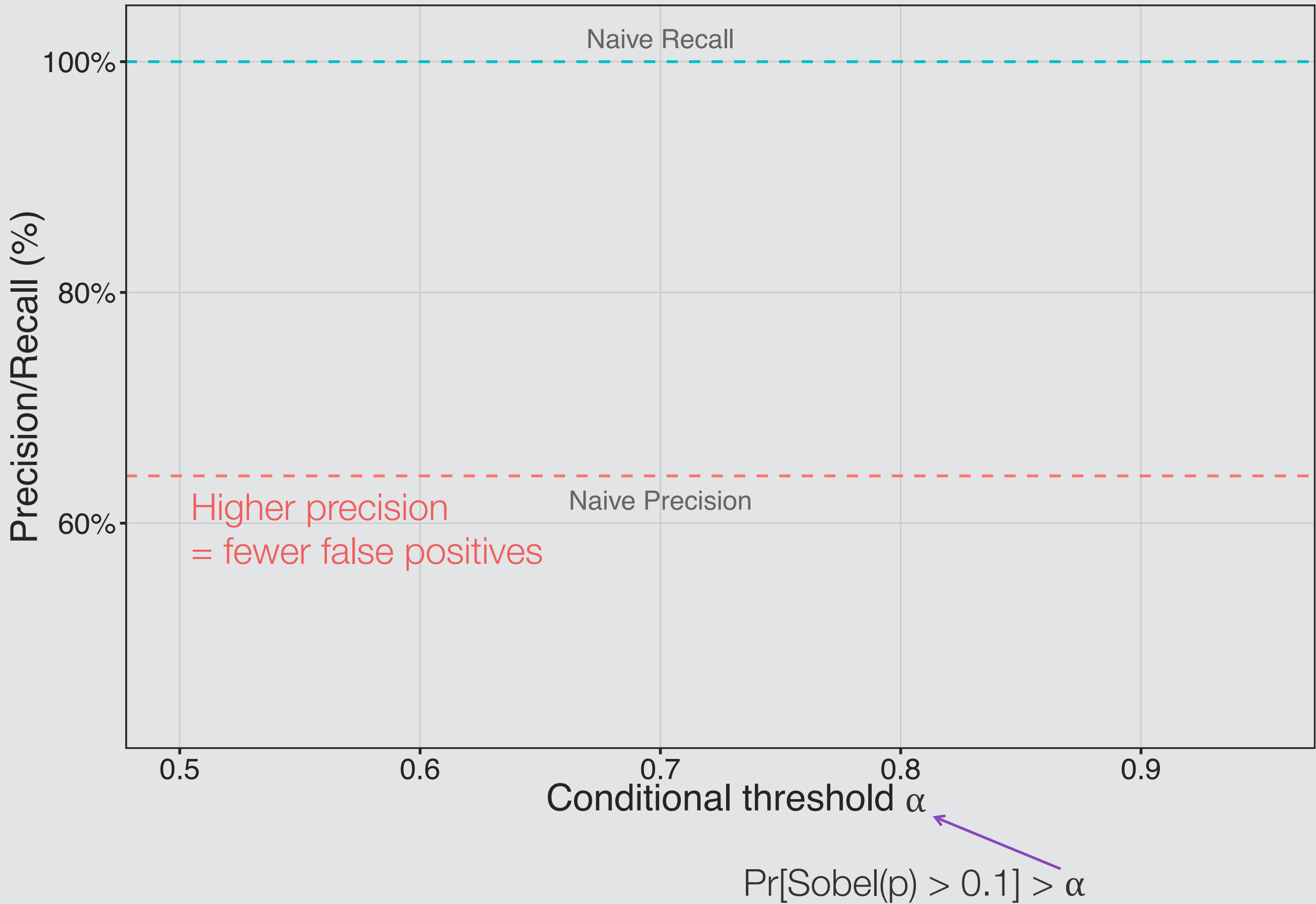


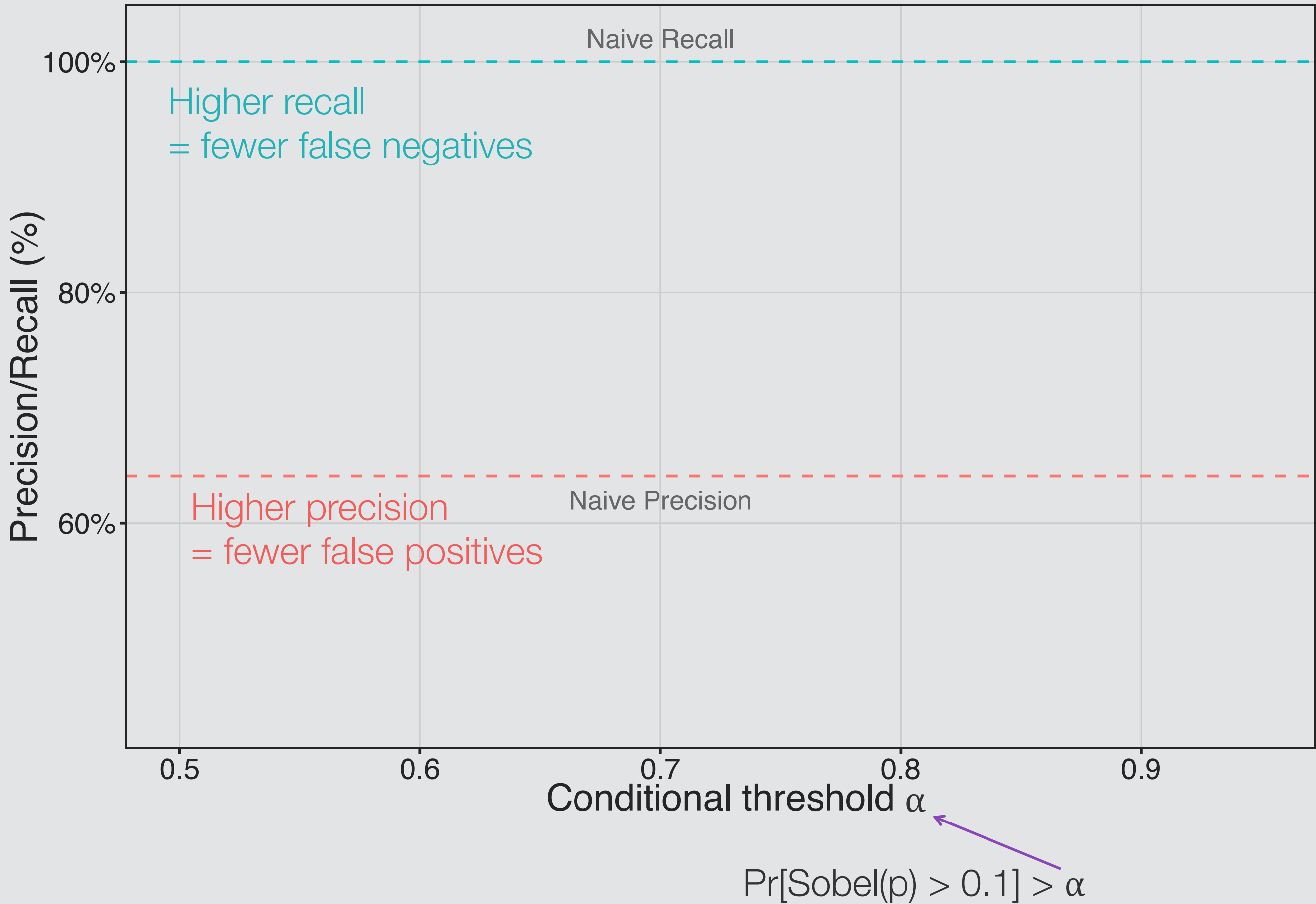
Naive Recall

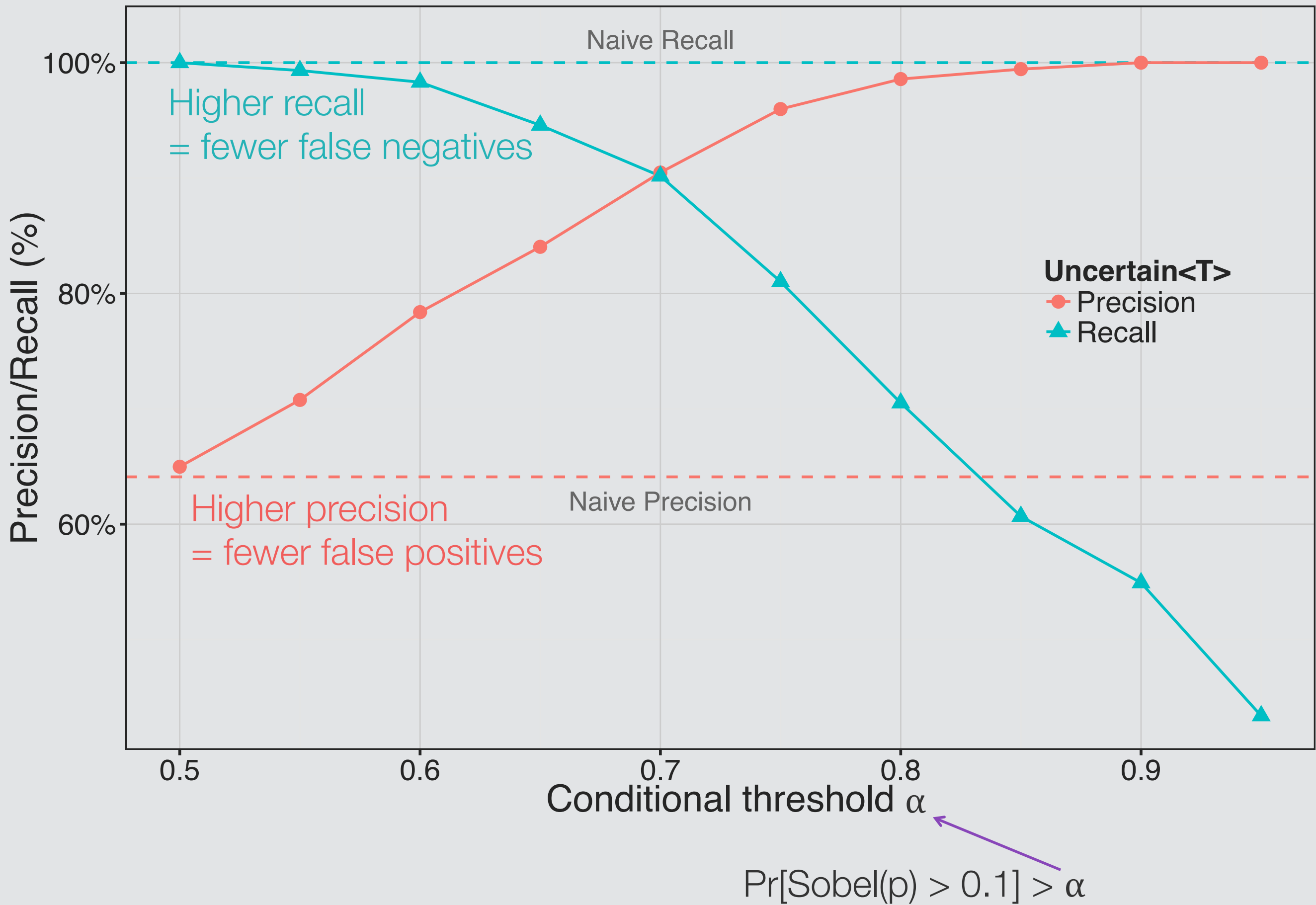
Naive Precision

Conditional threshold α

$\Pr[\text{Sobel}(p) > 0.1] > \alpha$







Uncertain<T> is an uncertain type abstraction.

It encourages **non-expert developers** to explicitly reason about uncertainty.

Uncertain<T> is an uncertain type abstraction.

It encourages **non-expert developers** to explicitly reason about uncertainty.

Thank you!