VINI CARIDÁ

Human-Centered Data & AI

Vinicius Caridá, Ph.D.

Data Science Manager, Itaú Unibanco

MBA Professor, FIAP

GDE – Machine Learning

@vinicius caridá @vfcarida @vinicius caridá @vfcarida @vinicius caridá @vfcarida

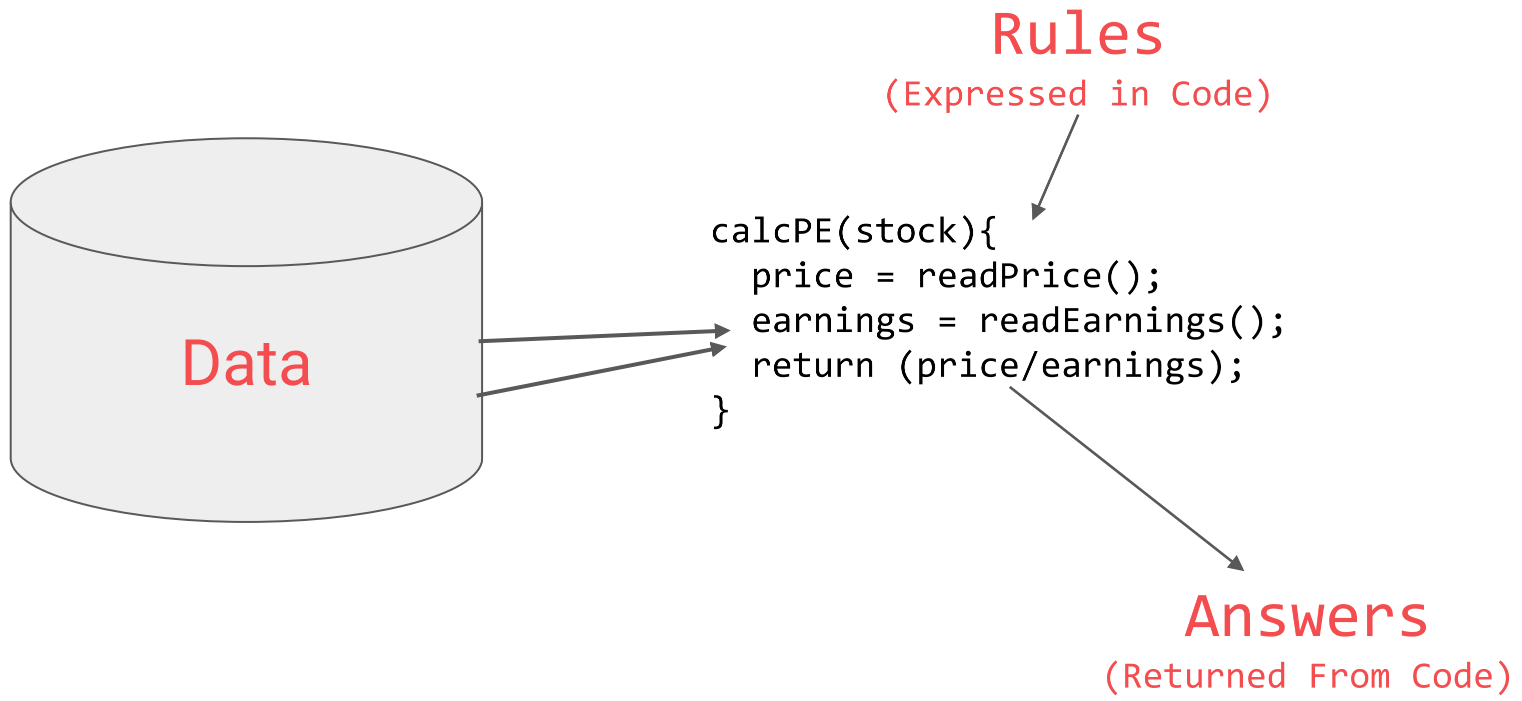
VINI CARIDÁ
Human-Centered Data & AI

TensorFlow

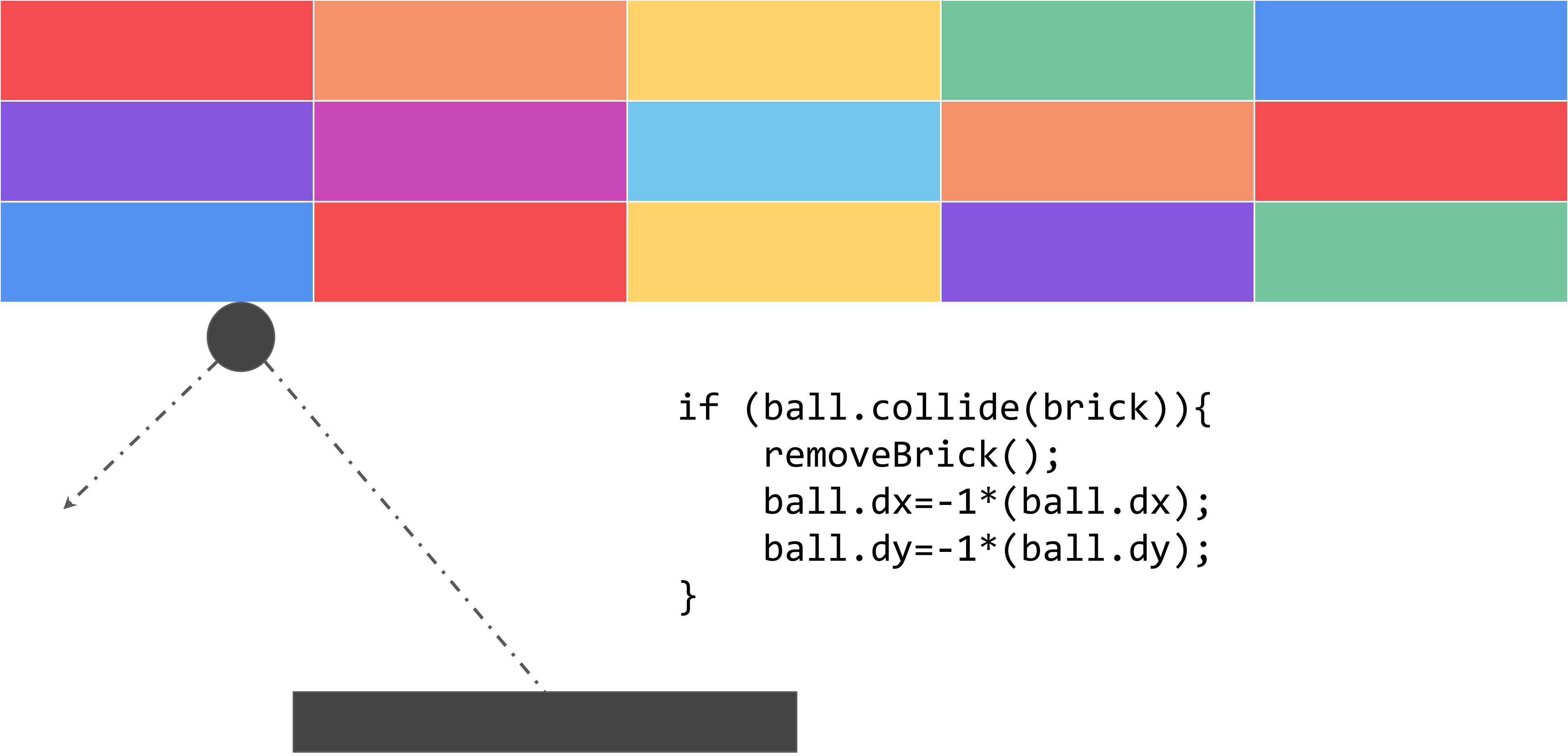
Zero to Hero with TensorFlow

Vinicius Caridá

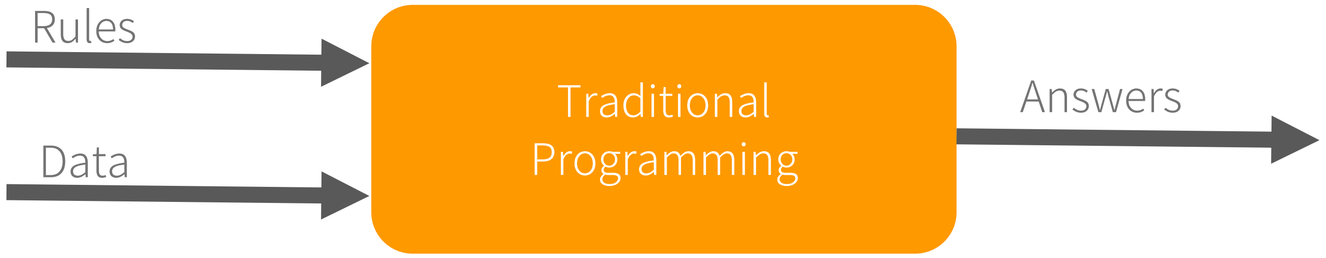
@vfcarida

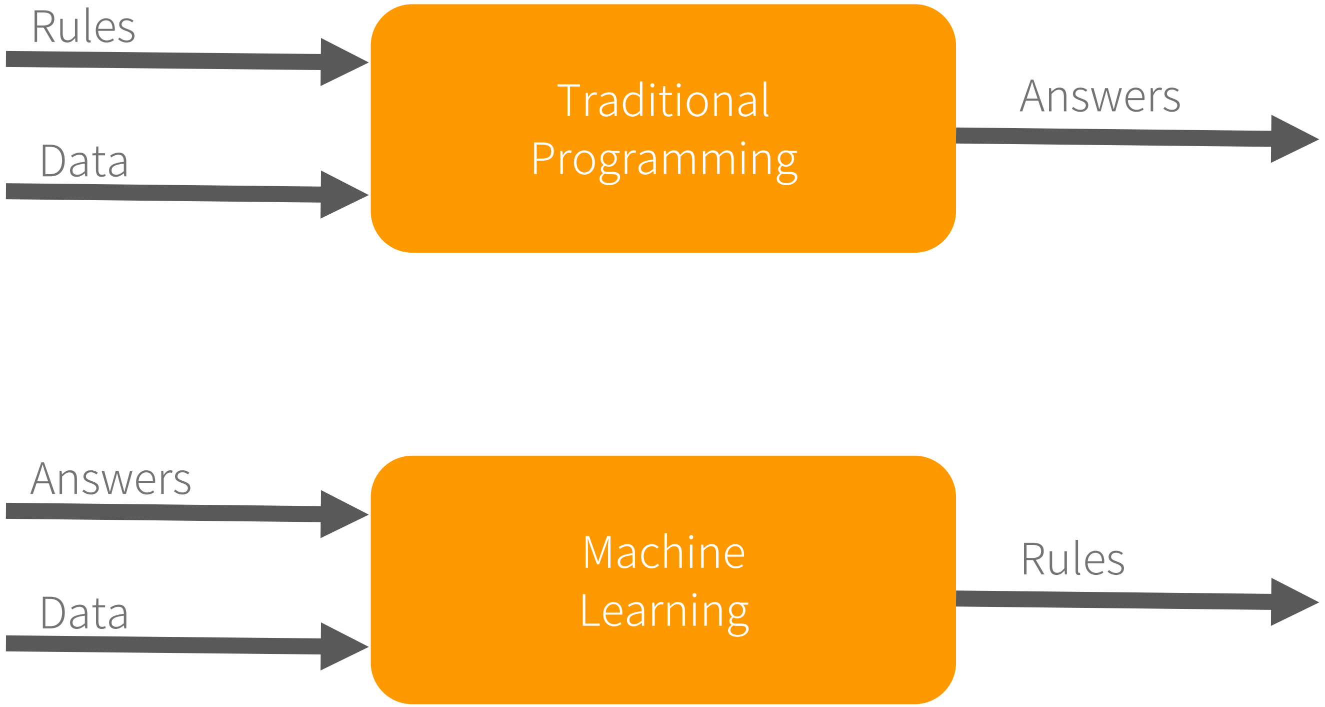


```
calcPE(stock){
  price = readPrice();
  earnings = readEarnings();
  return (price/earnings);
}
```

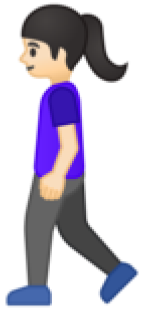


```
if (ball.collide(brick)){
    removeBrick();
    ball.dx=-1*(ball.dx);
    ball.dy=-1*(ball.dy);
}
```


Rules
Data
Traditional Programming
Answers

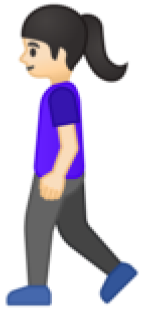

Rules
Data
Traditional Programming
Answers
Answers
Data
Machine Learning
Rules

Activity Recognition



```
if(speed<4){
    status=WALKING;
}
```

Activity Recognition



```
if(speed<4){
     status=WALKING;
}
```

```
if(speed<4){
     status=WALKING;
} else {
     status=RUNNING;
}
```

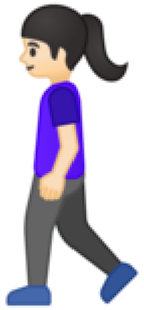
Activity Recognition

```
if(speed<4){
    status=WALKING;
}
```

```
if(speed<4){
    status=WALKING;
} else {
    status=RUNNING;
}
```

```
if(speed<4){
    status=WALKING;
} else if(speed<12){
    status=RUNNING;
} else {
    status=BIKING;
}
```

Activity Recognition

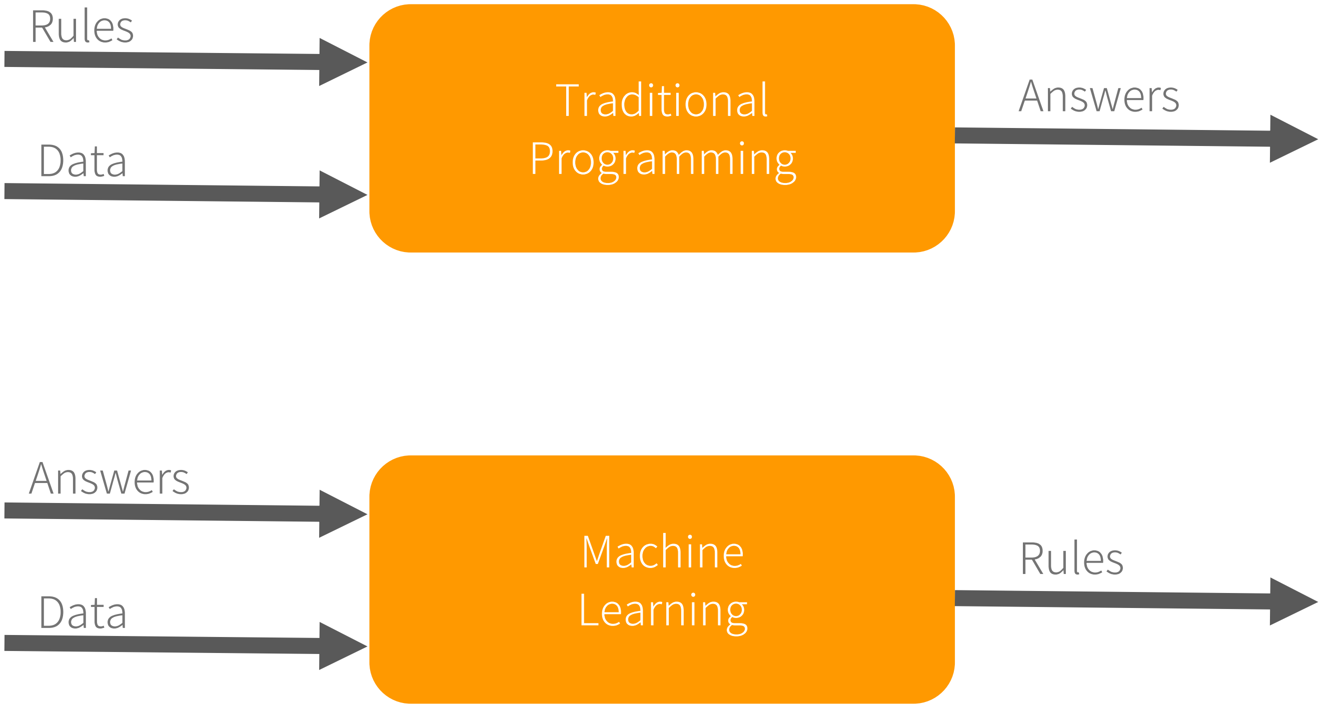


```
if(speed<4){
    status=WALKING;
}
```

```
if(speed<4){
    status=WALKING;
} else {
    status=RUNNING;
}
```

```
if(speed<4){
    status=WALKING;
} else if(speed<12){
    status=RUNNING;
} else {
    status=BIKING;
}
```

```
// ????
```


Rules
Data
Traditional Programming
Answers
Answers
Data
Machine Learning
Rules

Activity Recognition

```
0101001010100101010
1001010101001011101
0100101010010101001
0101001010100101010
```

Label = WALKING

```
1010100101001010101
0101010010010010001
0010011111010101111
1010100100111101011
```

Label = RUNNING

```
1001010011111010101
1101010111010101110
1010101111010101011
1111110001111010101
```

Label = BIKING

```
1111111111010011101
0011111010111110101
0101110101010101110
1010101010100111110
```

Label = GOLFING
(Sort of)

Activity Recognition

```
0101001010100101010
1001010101001011101
0100101010010101001
0101001010100101010
```

Label = WALKING

```
1010100101001010101
0101010010010010001
0010011111010101111
1010100100111101011
```

Label = RUNNING

```
1001010011111010101
1101010111010101110
1010101111010101011
1111110001111010101
```

Label = BIKING

```
1111111111010011101
0011111010111110101
0101110101010101110
1010101010100111110
```

Label = GOLFING
(Sort of)

Activity Recognition

010$\boxed{1001}$010100101010
1001010101001011101
010010101001010$\boxed{1001}$
0101001010100101010

Label = $\boxed{\text{WALKING}}$

1010100101001010101
0101010010010010001
001001111101010$\boxed{1111}$
1010100100111101011

Label = $\boxed{\text{RUNNING}}$

1001010011111010101
1101010111010101110
101010111101010$\boxed{1011}$
1111110001111010101

Label = $\boxed{\text{BIKING}}$

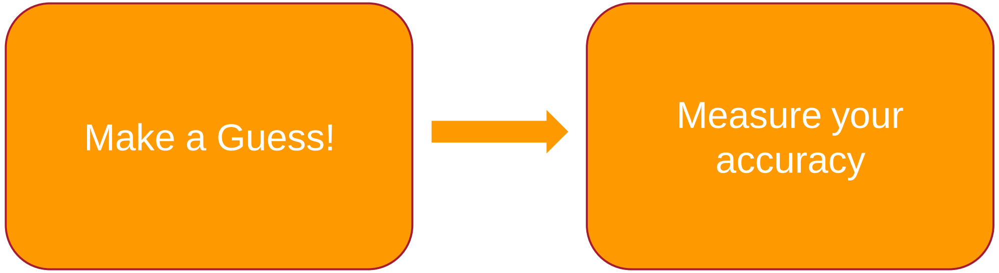
1111111111010011101
0011111010111110101
010111010101010$\boxed{1110}$
1010101010100111110

Label = $\boxed{\text{GOLFING}}$
(Sort of)

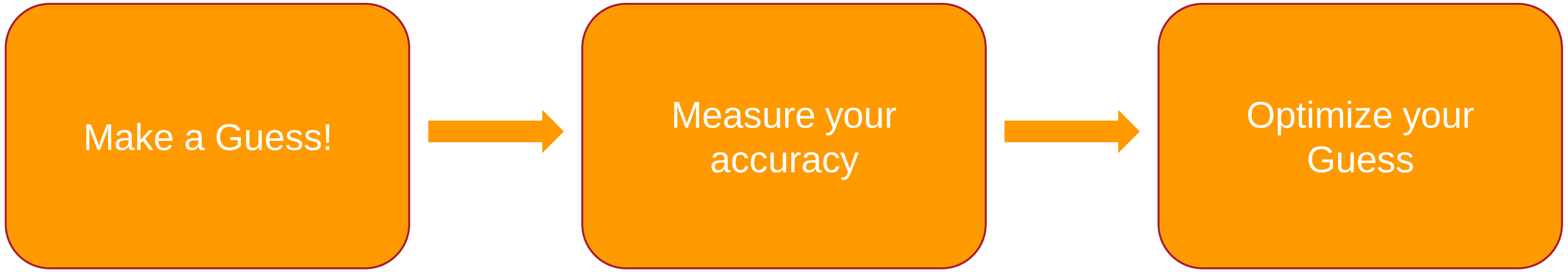
The Machine Learning Paradigm

Make a Guess!

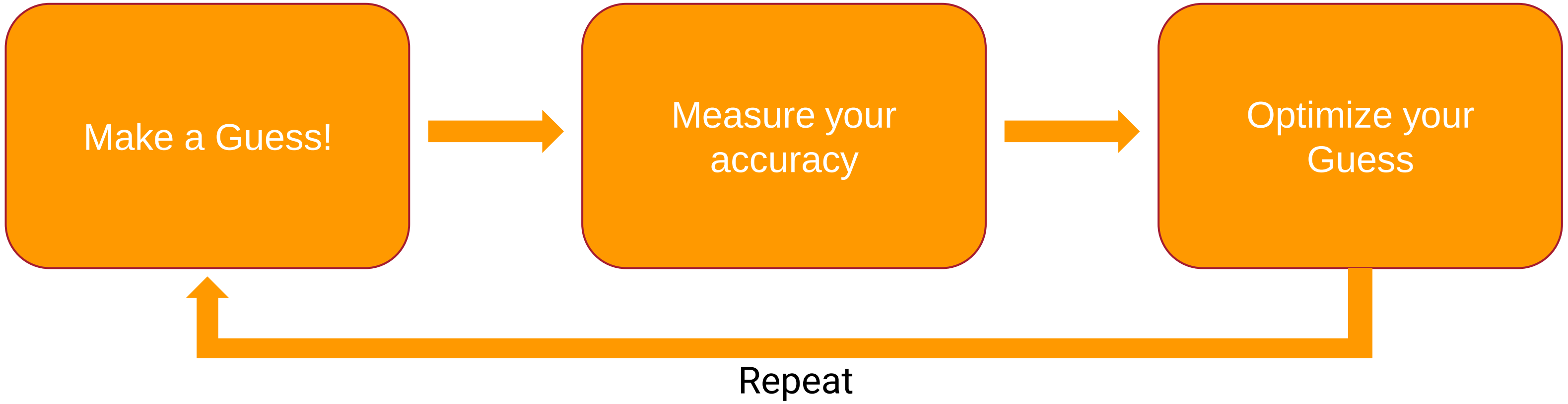
The Machine Learning Paradigm



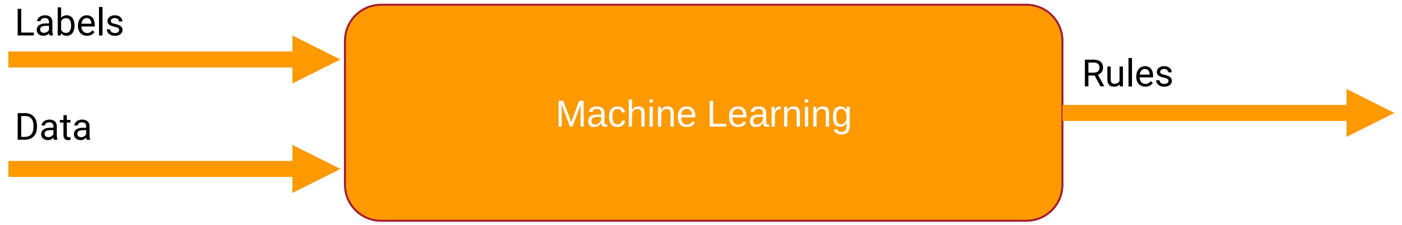
The Machine Learning Paradigm



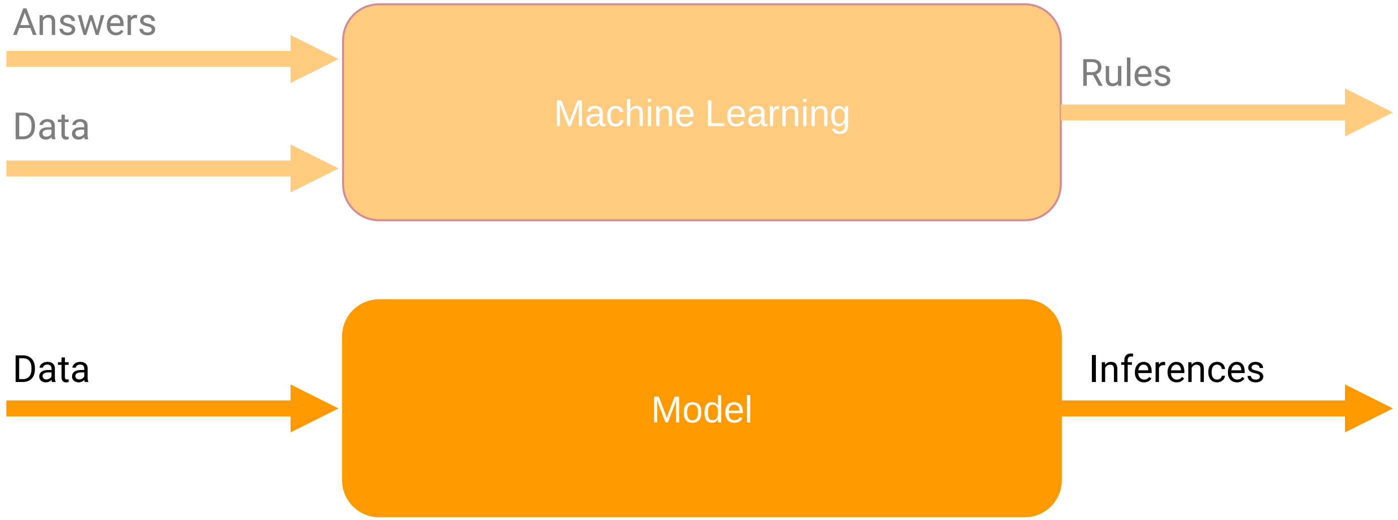
The Machine Learning Paradigm



The Machine Learning Paradigm



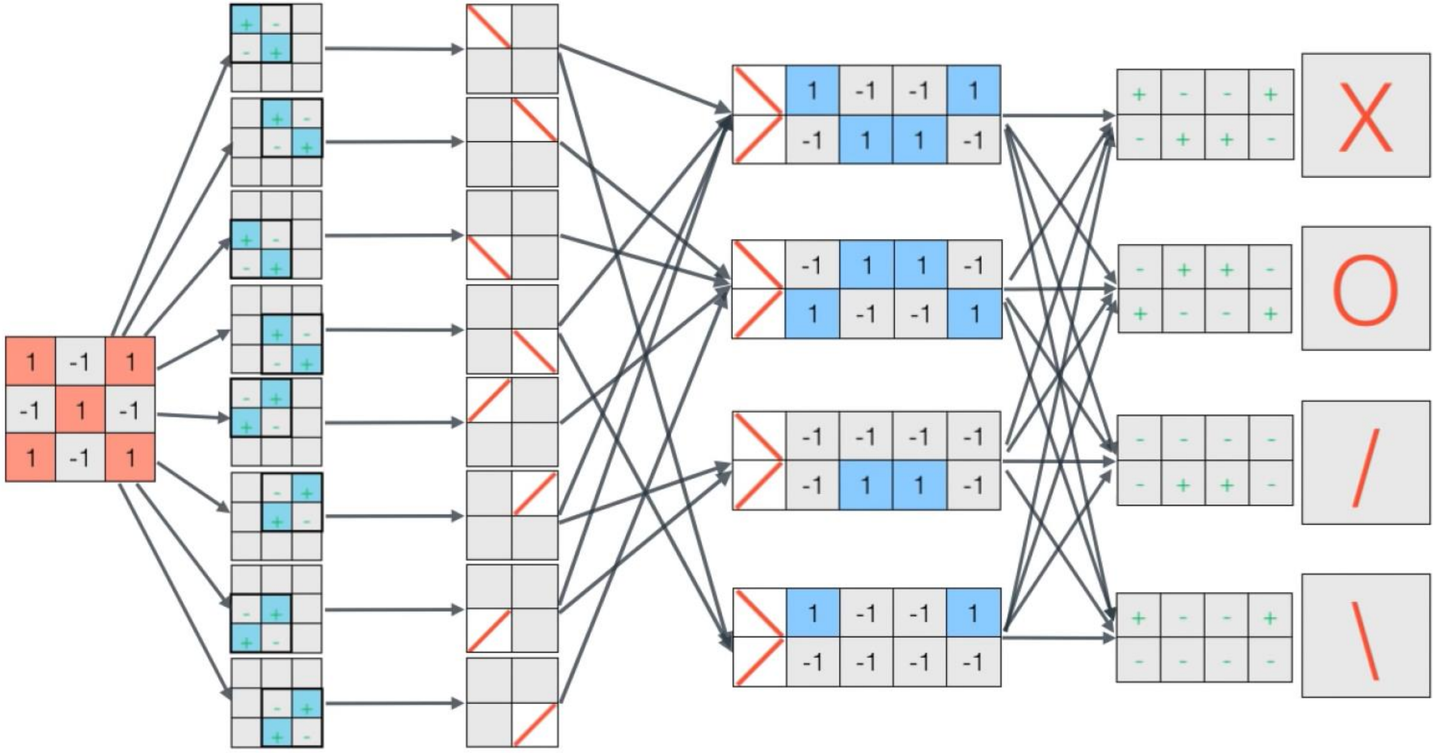
The Machine Learning Paradigm



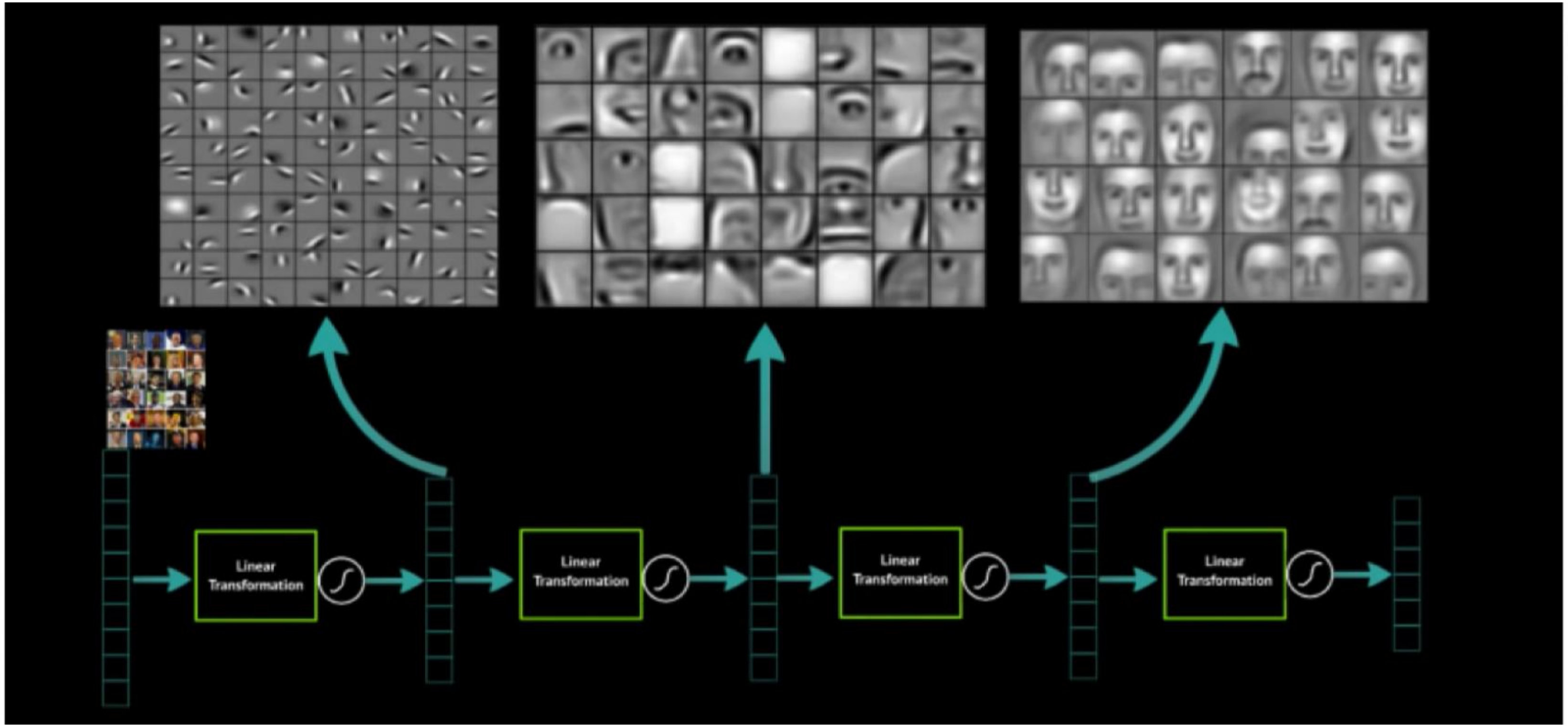
TensorFlow

Computer Vision

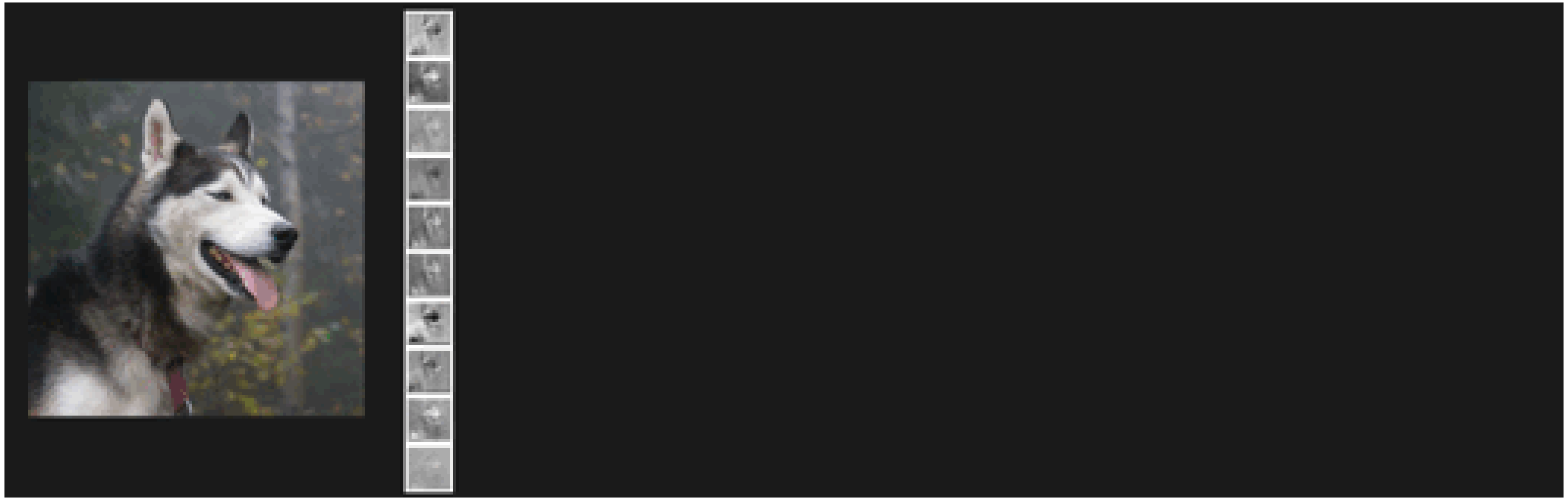
Using Machine Learning to understand images



Convolution Layer Pooling Layer Fully Connected Layer



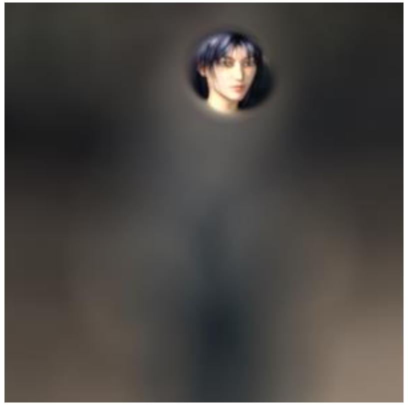
Linear Transformation
Linear Transformation
Linear Transformation
Linear Transformation



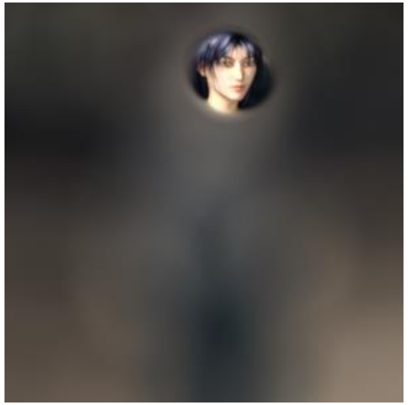
+

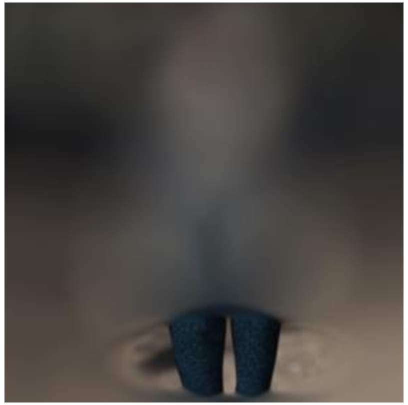
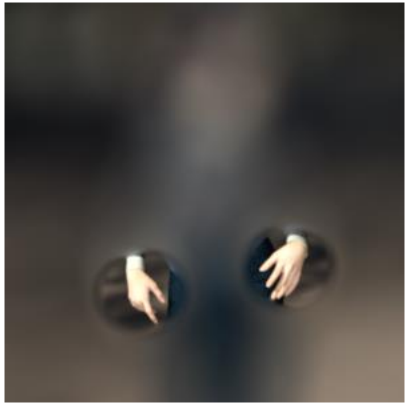
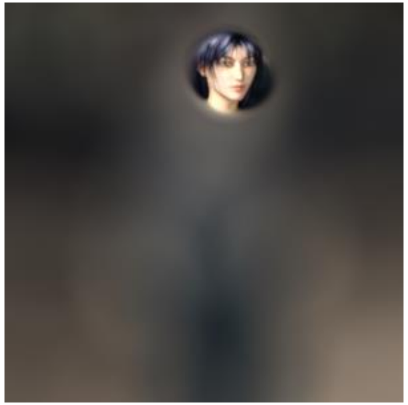
+

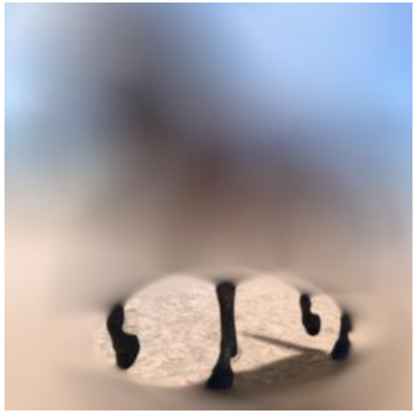
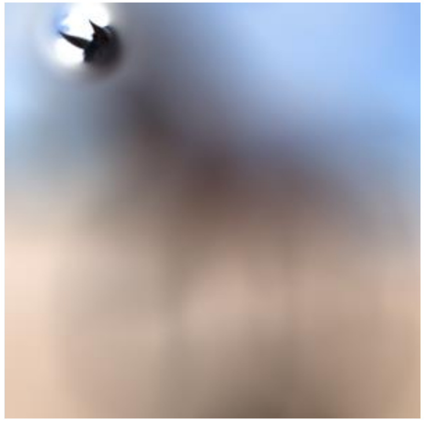
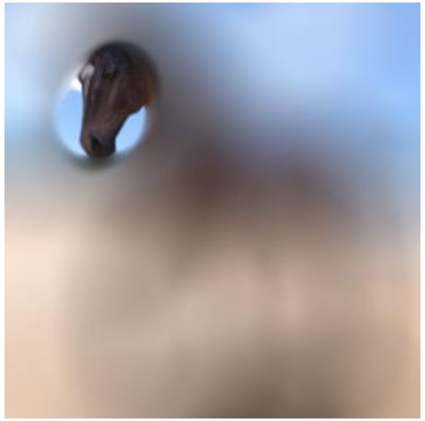
+

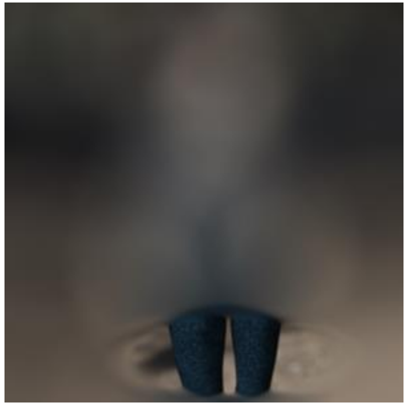


+

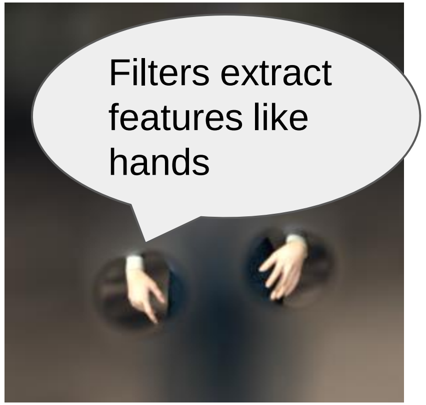
+

= HUMAN


+

+

= HUMAN

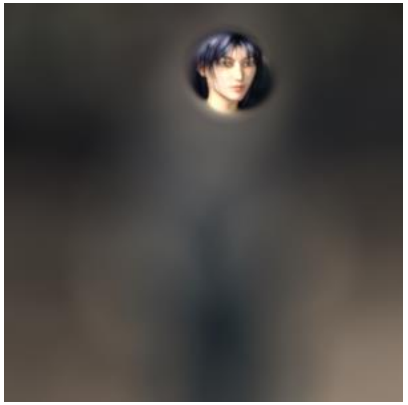

+

+

= HORSE



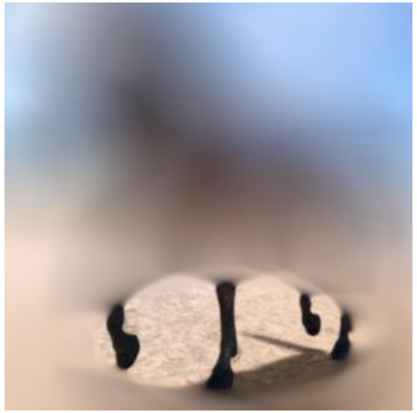
+



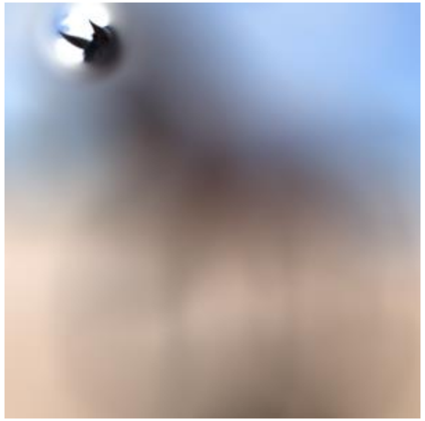
+



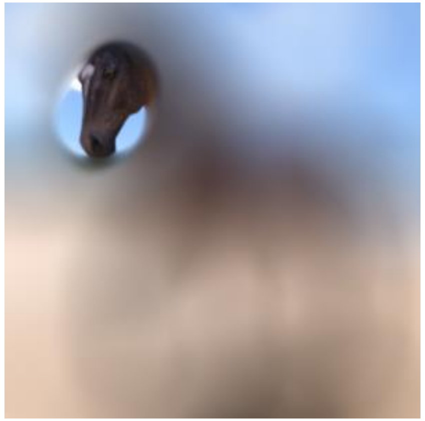
= HUMAN



+



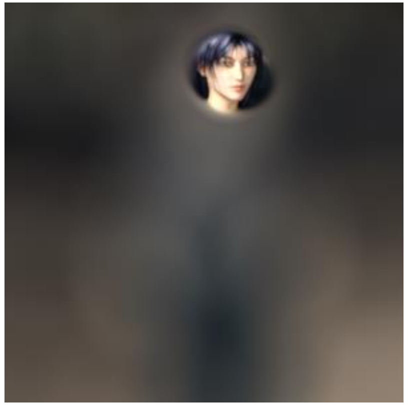
+



= HORSE

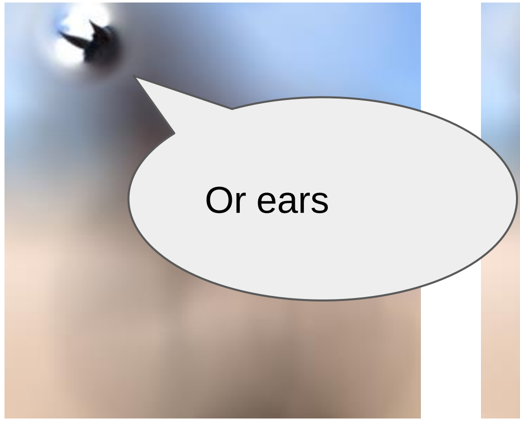
+

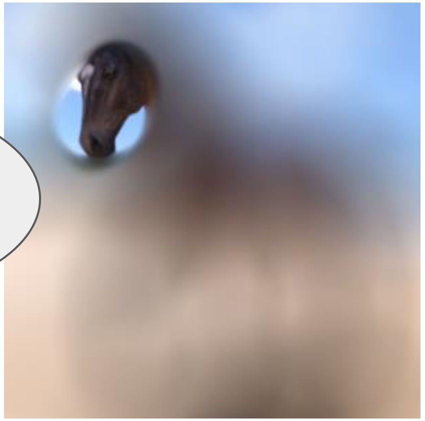
+



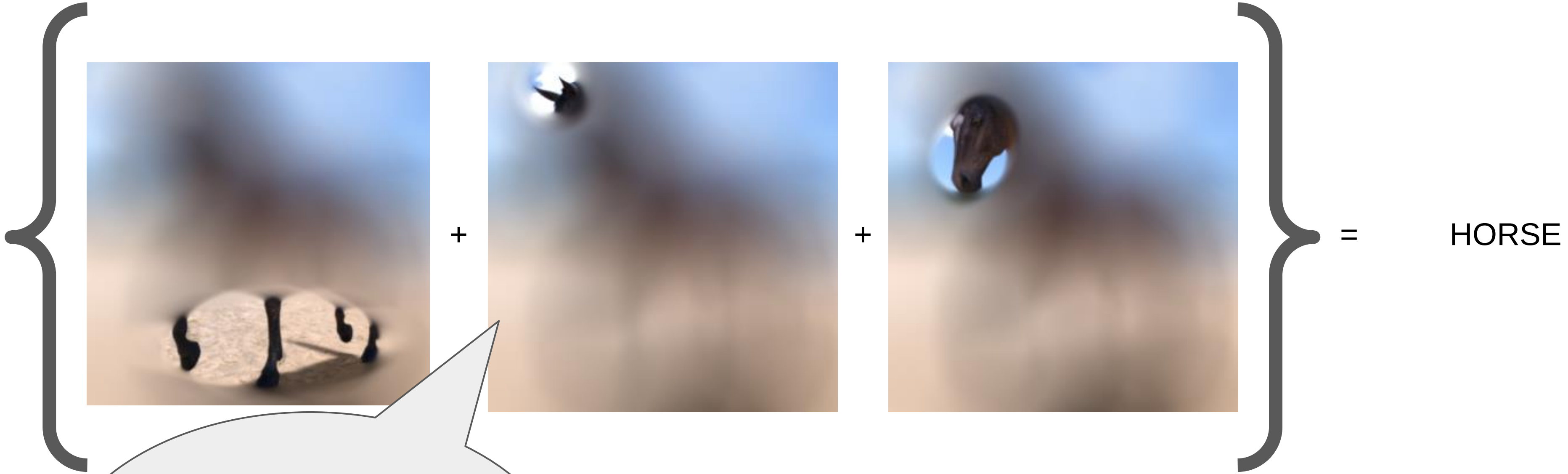
= HUMAN

+

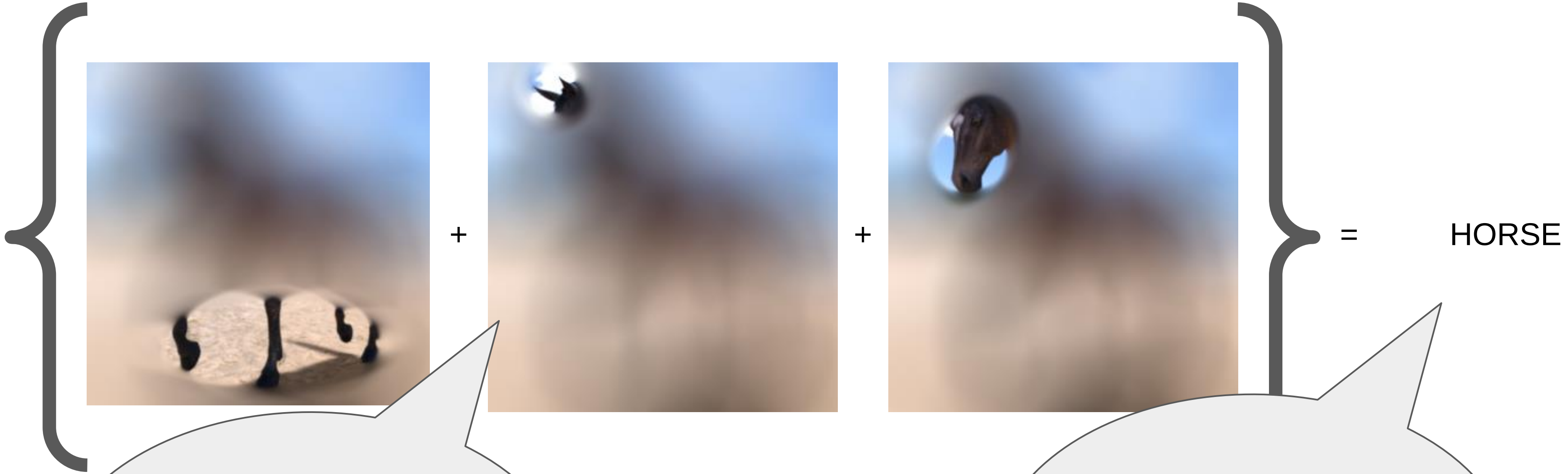




= HORSE

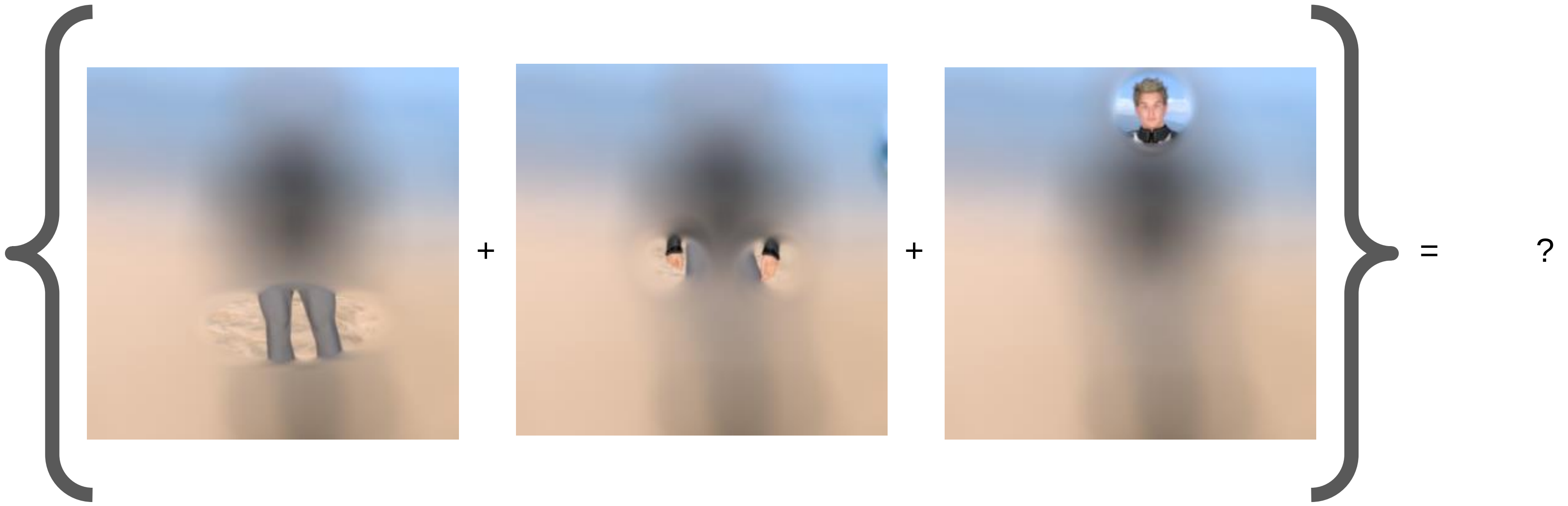


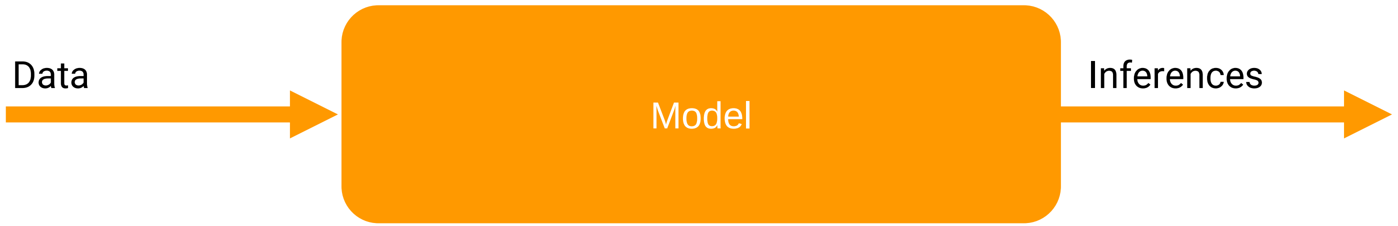
Filters can then be combined with labels to make a prediction of the image contents...

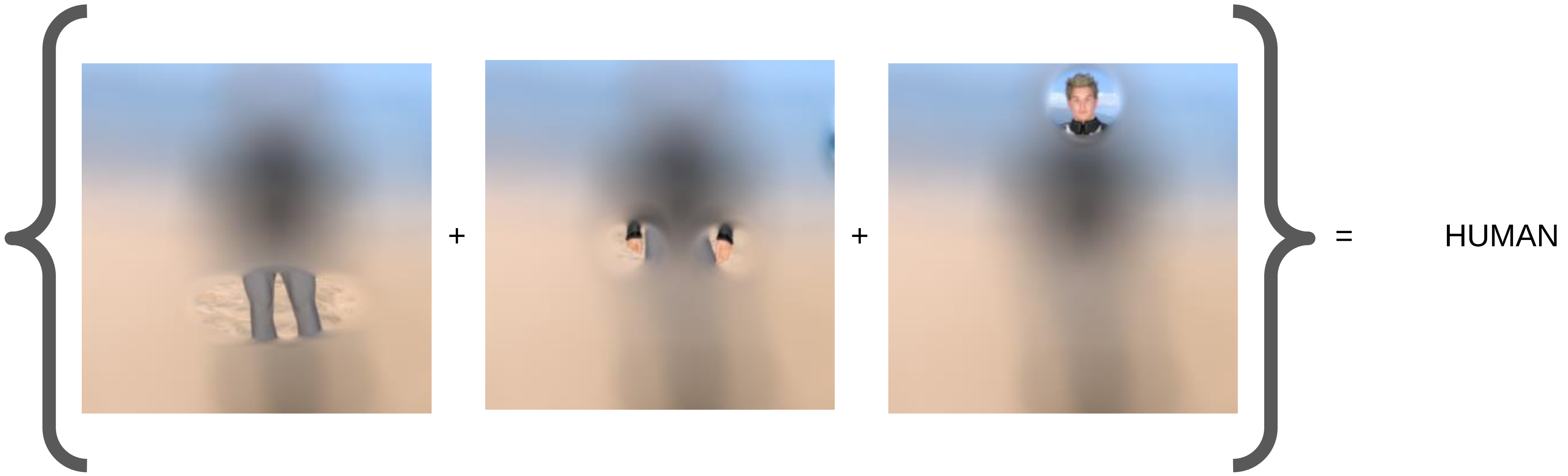


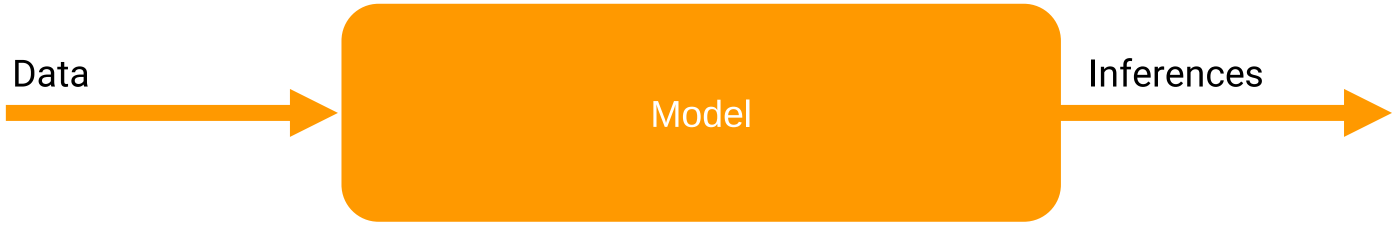
Filters can then be combined with labels to make a prediction of the image contents...

The filters that match the label are learned over time!

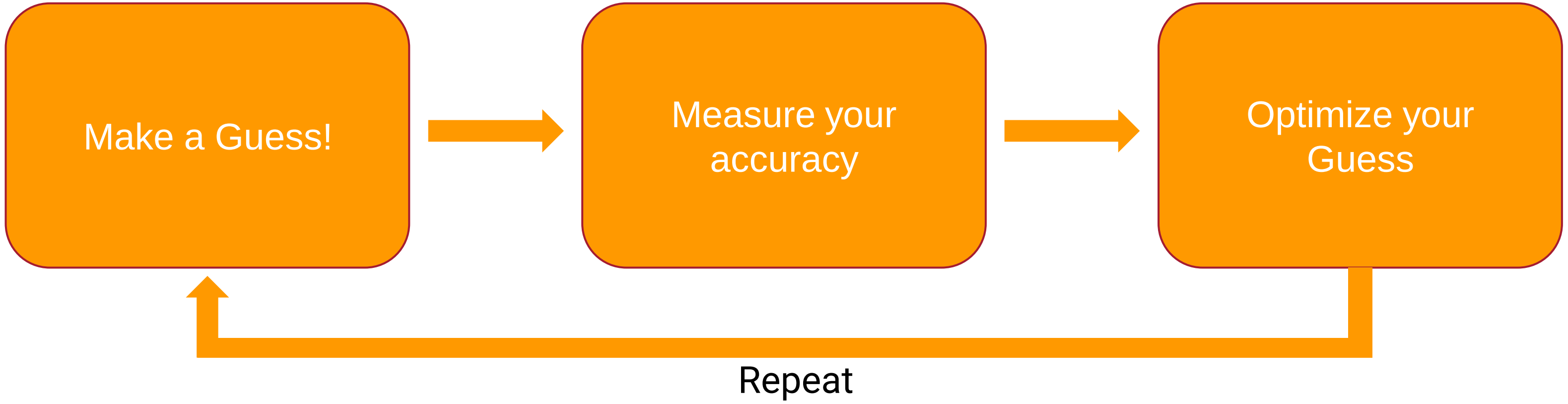

+
+
=
?


Data
Model
Inferences

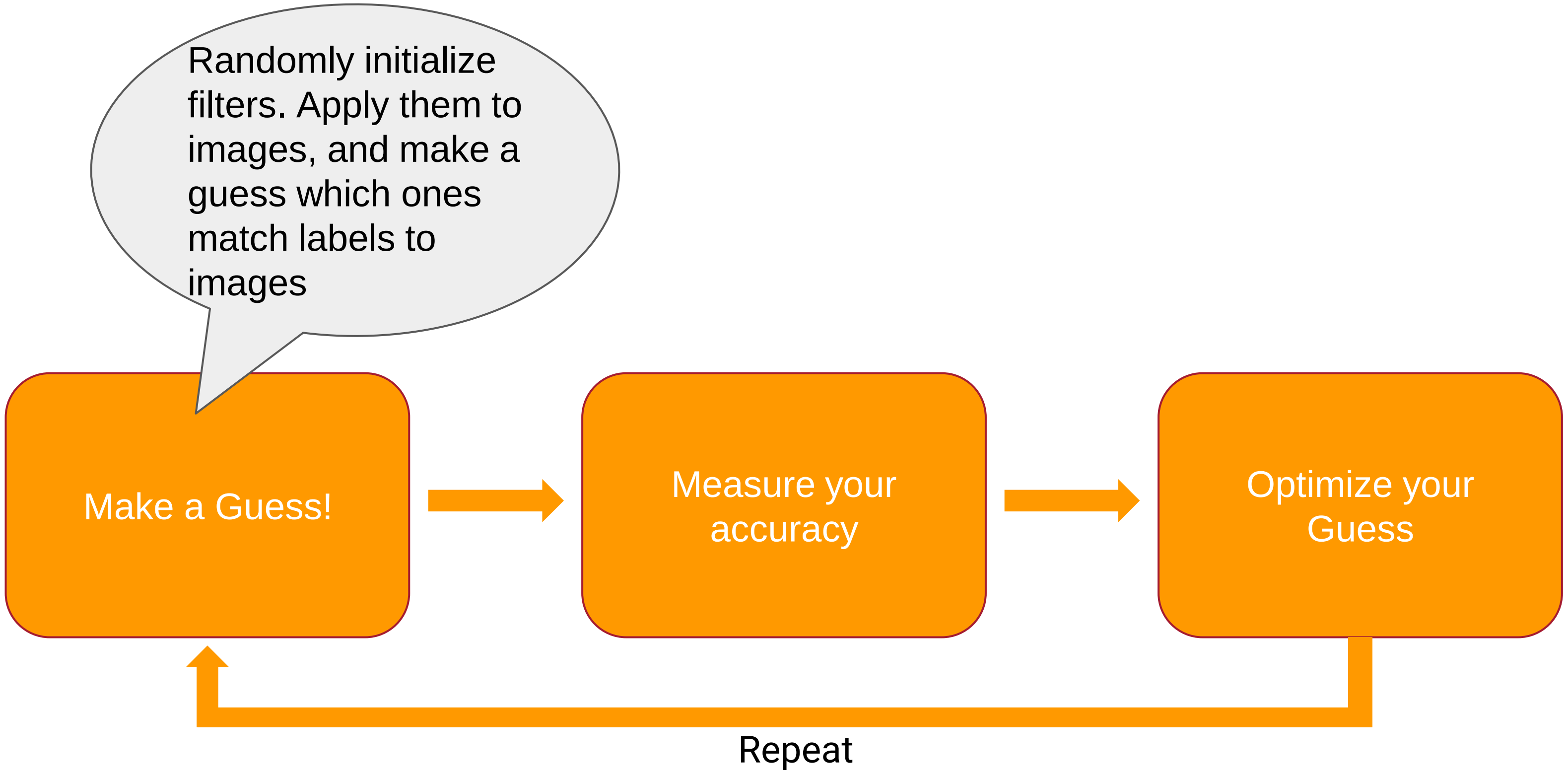

+
+
=
HUMAN


Data
Model
Inferences

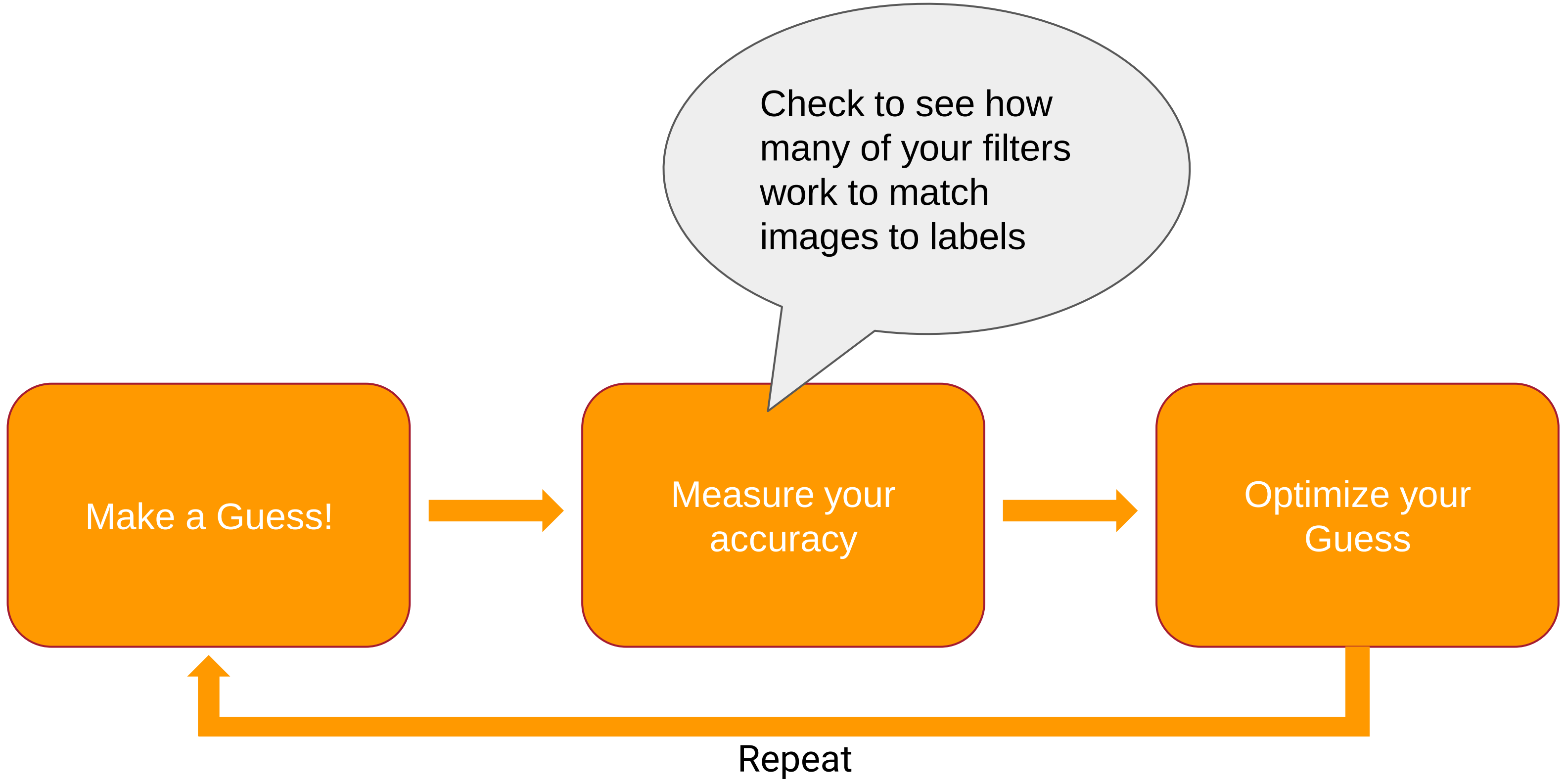
The Machine Learning Paradigm



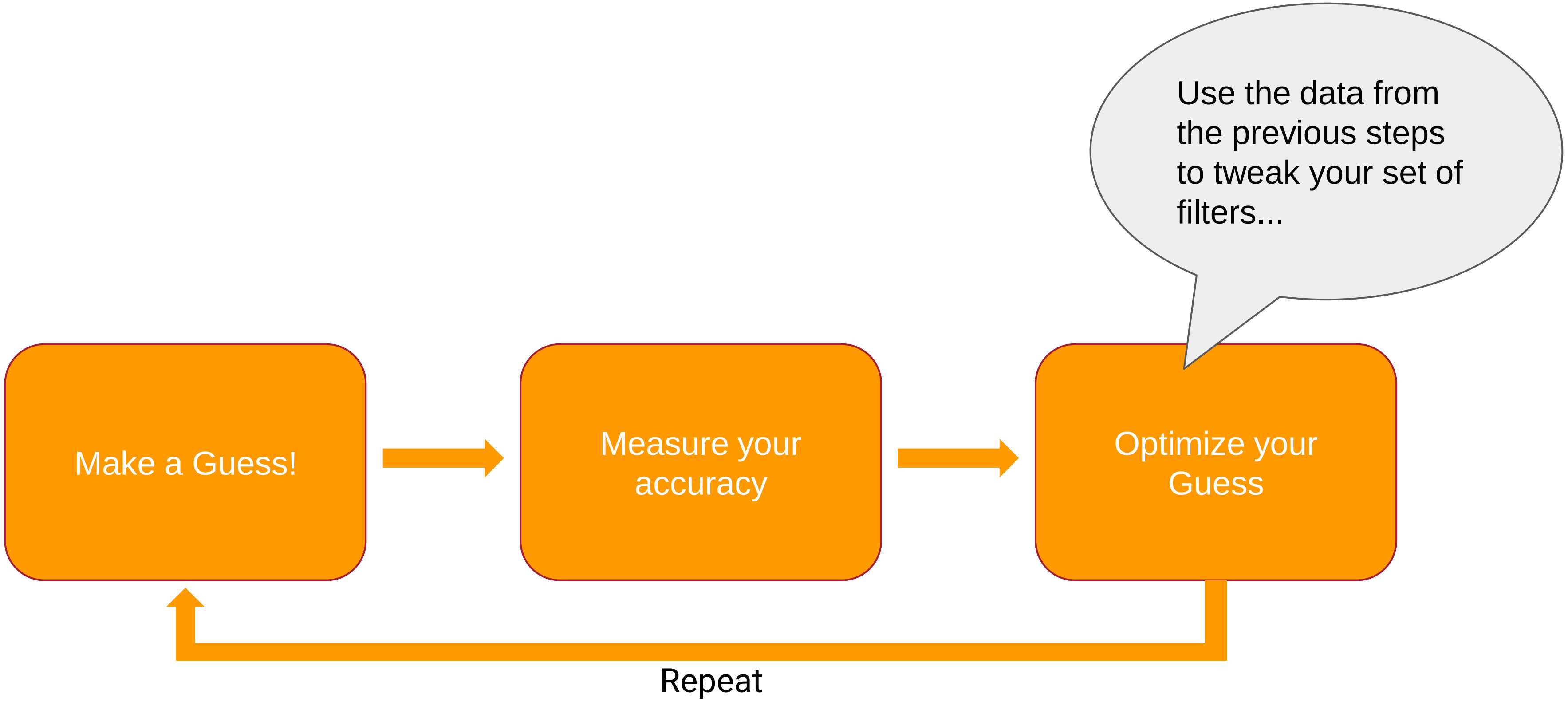
The Machine Learning Paradigm


Randomly initialize filters. Apply them to images, and make a guess which ones match labels to images
Make a Guess!
Measure your accuracy
Optimize your Guess
Repeat

The Machine Learning Paradigm

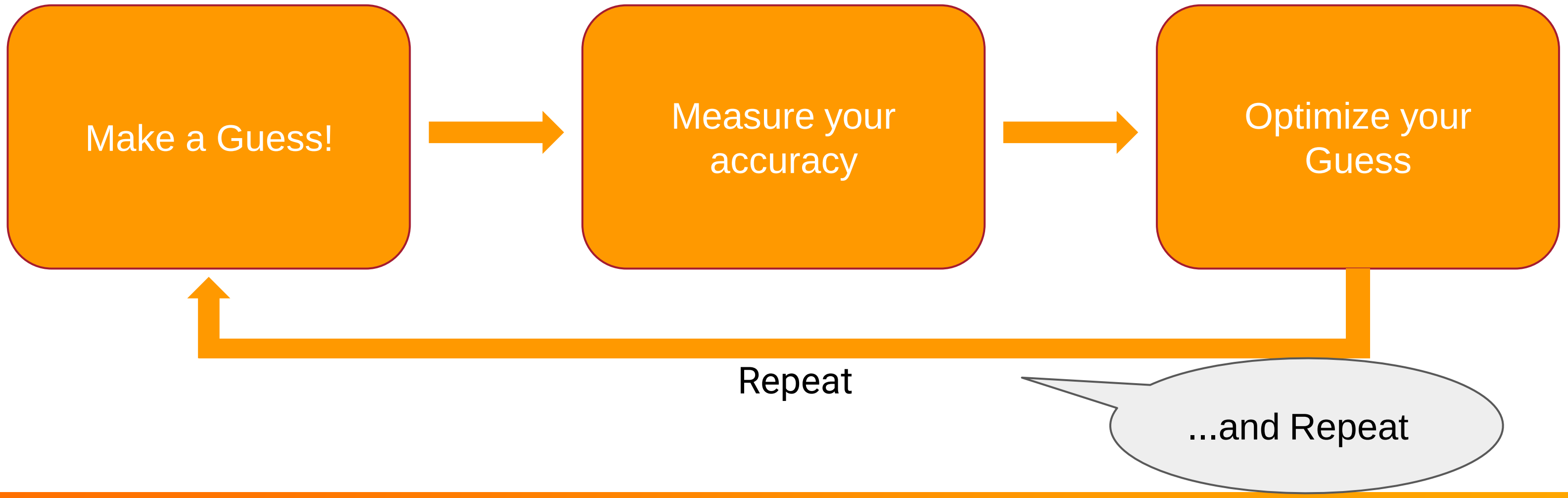

Check to see how many of your filters work to match images to labels
Make a Guess!
Measure your accuracy
Optimize your Guess
Repeat

The Machine Learning Paradigm



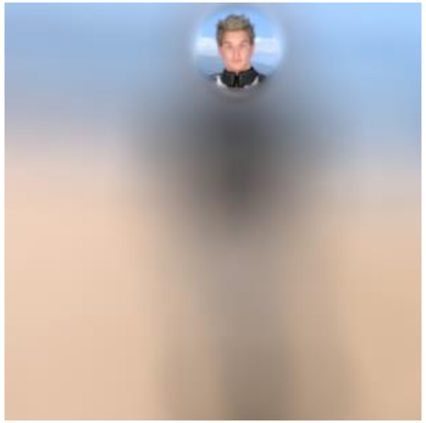
Use the data from the previous steps to tweak your set of filters...
Make a Guess!
Measure your accuracy
Optimize your Guess
Repeat

The Machine Learning Paradigm



+

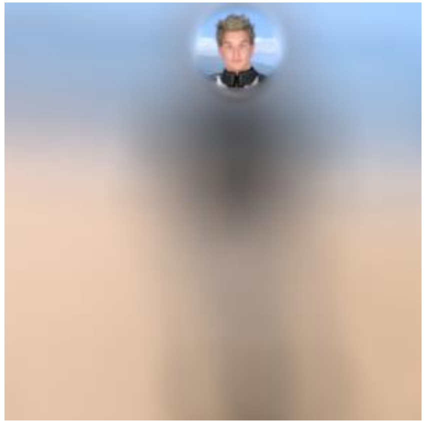
+



= ?

+

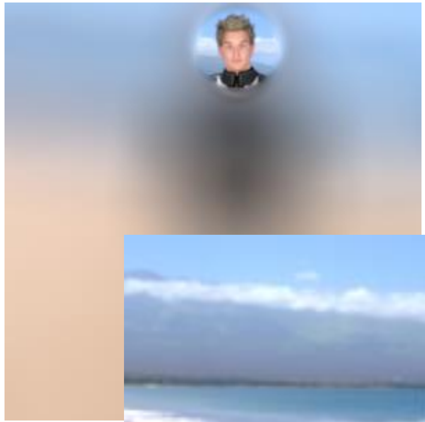
+



= HUMAN

+

+



= HUMAN

```python
model = tf.keras.models.Sequential([
    tf.keras.layers.Conv2D(16, (3, 3), activation='relu',
                           input_shape=(300, 300, 3)),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(512, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
```

```
model = tf.keras.models.Sequential([
    tf.keras.layers.Conv2D(16, (3, 3), activation='relu',
                           input_shape=(300, 300, 3)),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(512, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
```

Conv2D stands for 2D Convolution -- another word for a filter

```
model = tf.keras.models.Sequential([
    tf.keras.layers.Conv2D(16, (3, 3), activation='relu',
                           input_shape=(300, 300, 3)),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(512, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
```

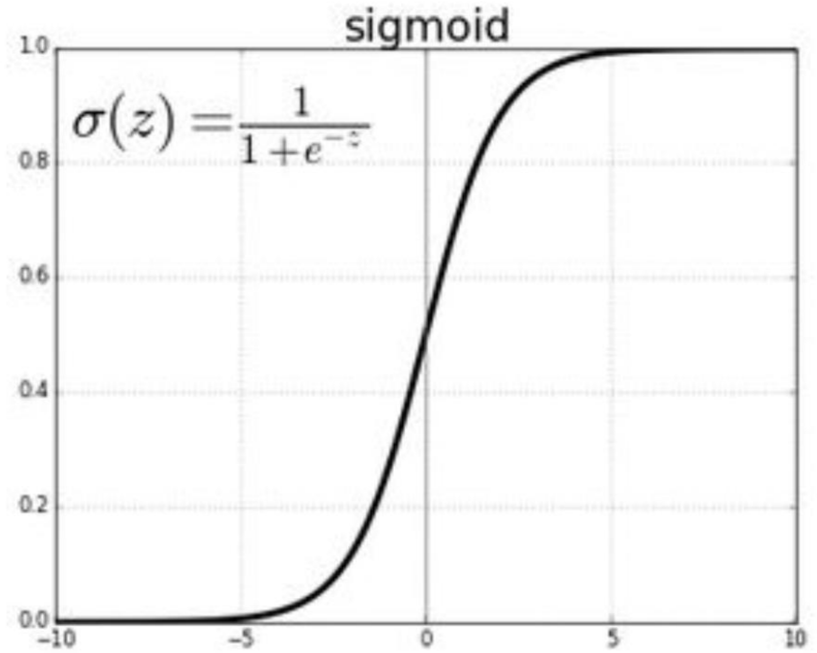
MaxPooling is a way of compressing the image while enhancing features

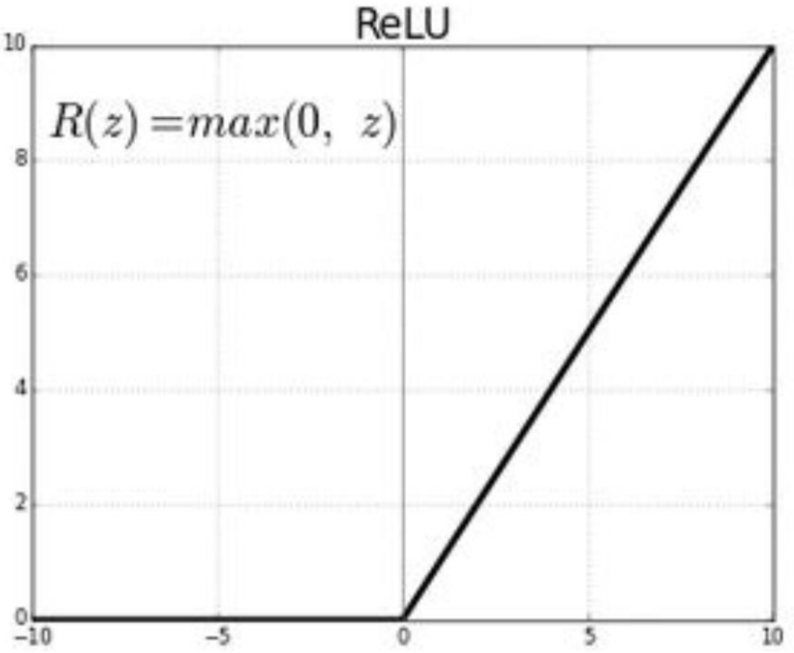
```
model = tf.keras.models.Sequential([
    tf.keras.layers.Conv2D(16, (3, 3), activation='relu',
                           input_shape=(300, 300, 3)),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(512, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
```

Dense is a neural network that matches the filters to the labels

```
model = tf.keras.models.Sequential([
    tf.keras.layers.Conv2D(16, (3, 3), activation='relu',
                           input_shape=(300, 300, 3)),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(512, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
```

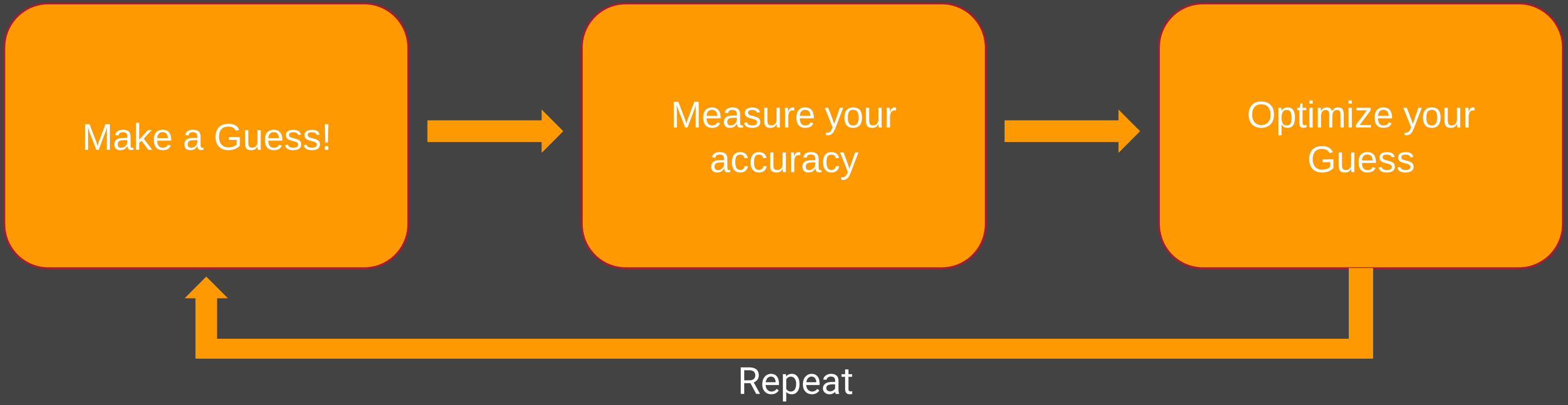
The final Dense represents the labels -
Horse = 0,
Human = 1


sigmoid
$\sigma(z) = \frac{1}{1+e^{-z}}$
1.0
0.8
0.6
0.4
0.2
0.0
−10
−5
0
5
10

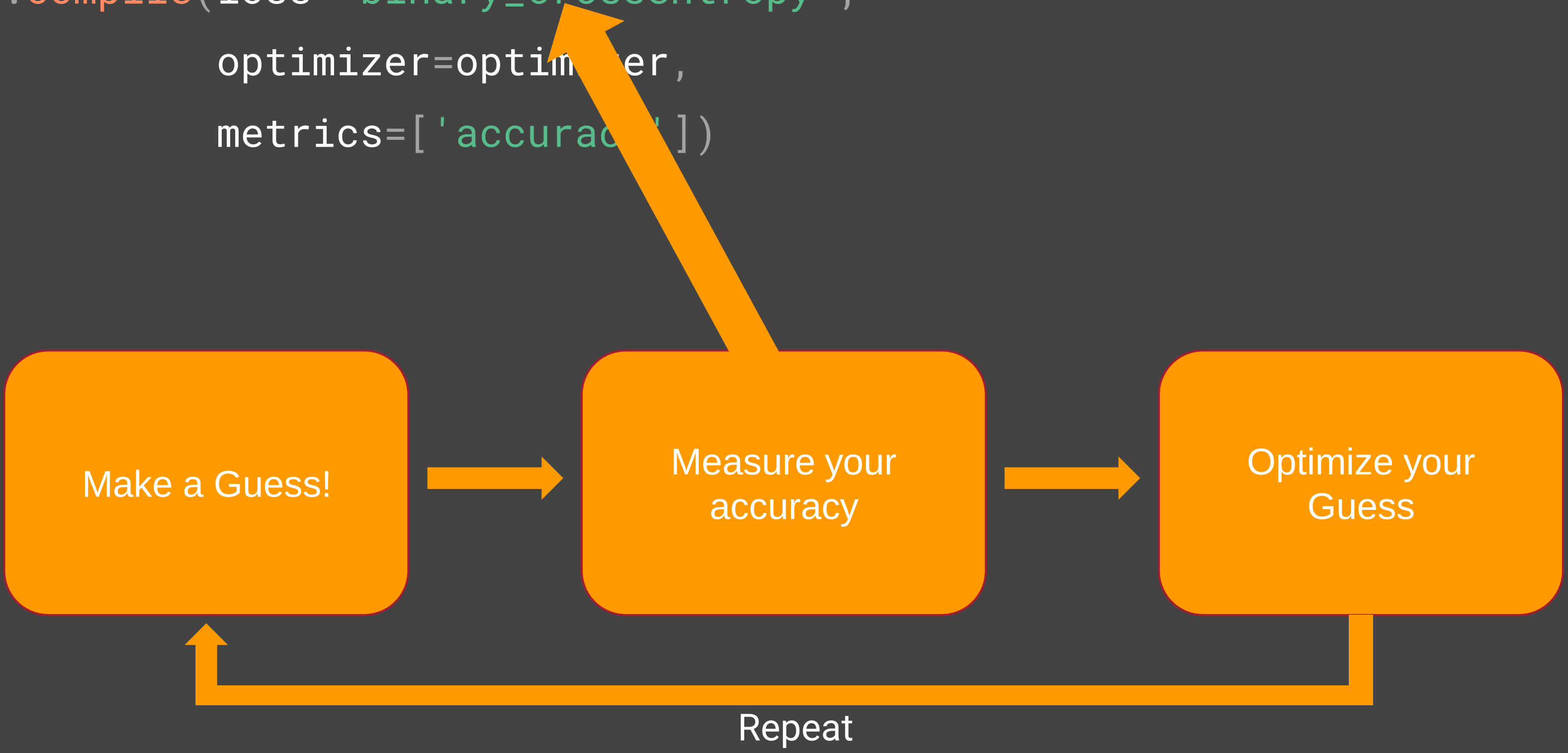

ReLU
$R(z) = max(0, z)$
10
8
6
4
2
0
−10
−5
0
5
10

```
optimizer = tf.keras.optimizers.RMSprop(lr=0.001)
model.compile(loss='binary_crossentropy',
              optimizer=optimizer,
              metrics=['accuracy'])
```

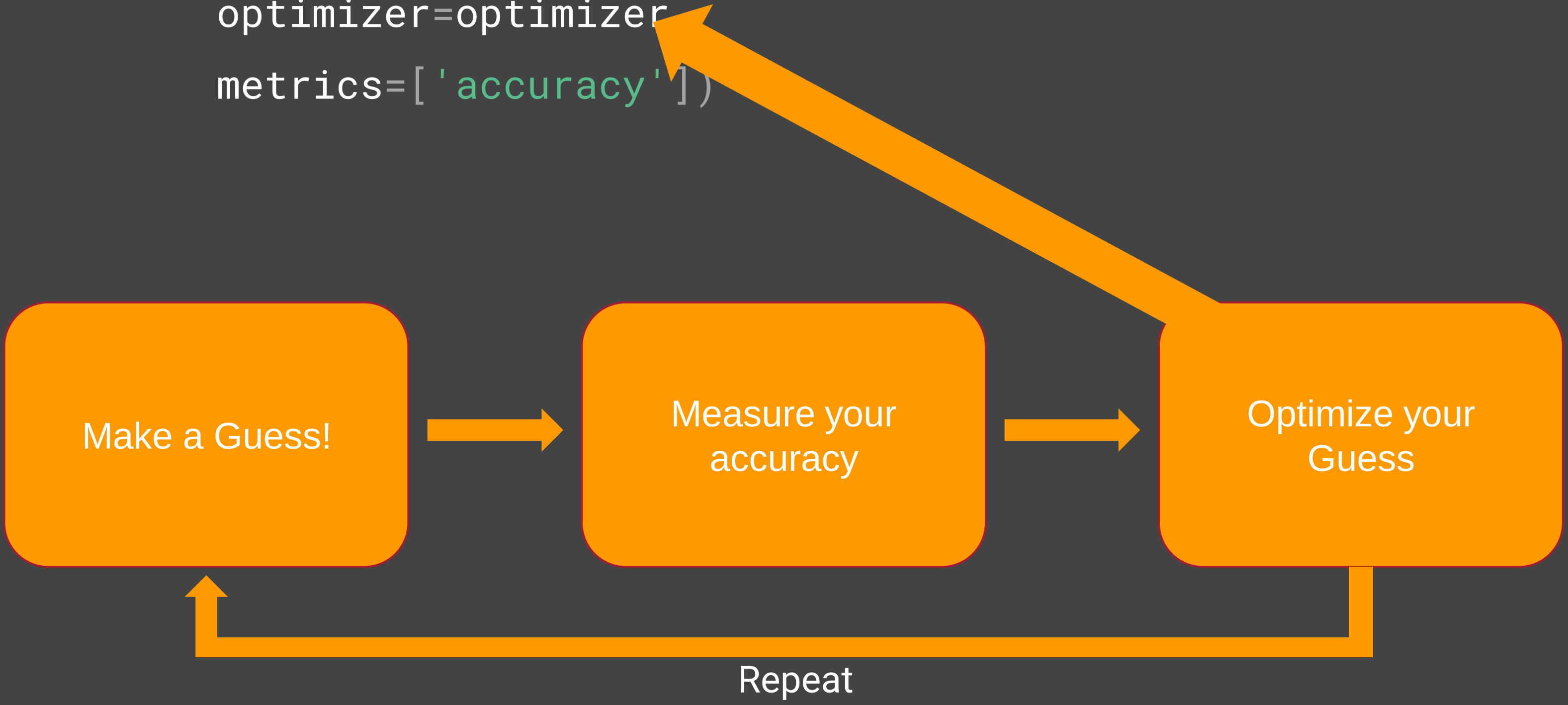
```
optimizer = tf.keras.optimizers.RMSprop(lr=0.001)
model.compile(loss='binary_crossentropy',
              optimizer=optimizer,
              metrics=['accuracy'])
```



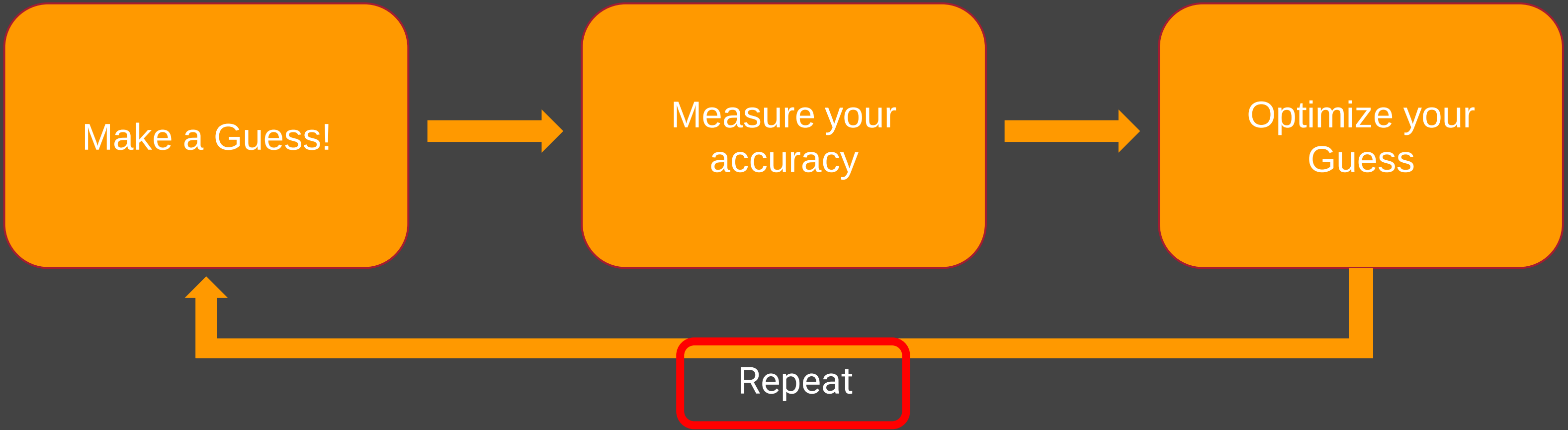
```
optimizer = tf.keras.optimizers.RMSprop(lr=0.001)
model.compile(loss='binary_crossentropy',
              optimizer=optimizer,
              metrics=['accuracy'])
```



```
optimizer = tf.keras.optimizers.RMSprop(lr=0.001)
model.compile(loss='binary_crossentropy',
              optimizer=optimizer,
              metrics=['accuracy'])
```



```
model.fit(train_generator, epochs=12, ...)
model.fit(train_generator, epochs=12, ...)
```

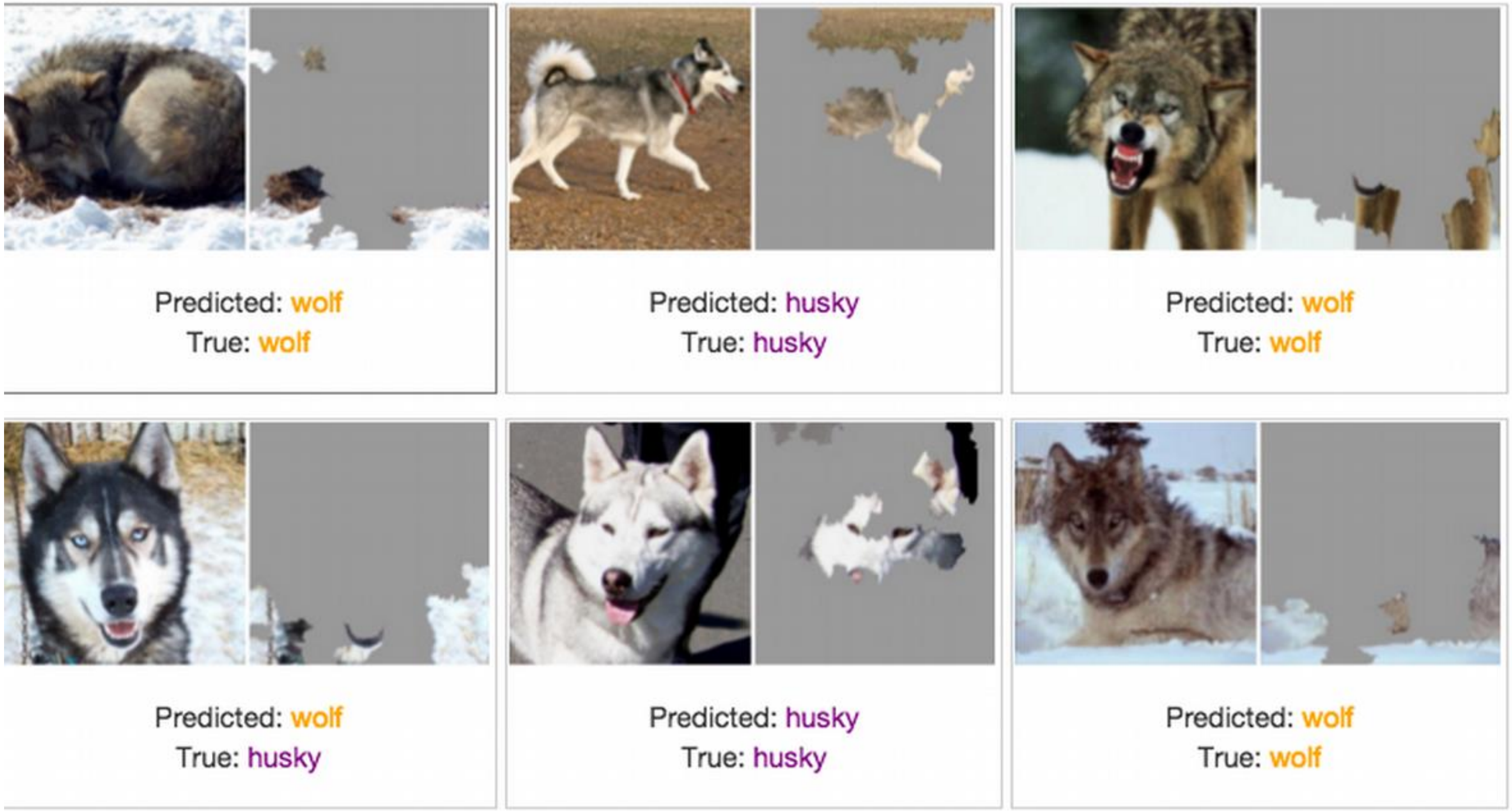


TensorFlow

Demo

Training a computer to recognize Horses or Humans

@teenybiscuit


Predicted: wolf
True: wolf
Predicted: husky
True: husky
Predicted: wolf
True: wolf
Predicted: wolf
True: husky
Predicted: husky
True: husky
Predicted: wolf
True: wolf

Thanks !

VINICARIDÁ
Human-Centered Data & AI

Vinicius Fernandes Caridá

vfcarida@gmail.com

@vinicius caridá @vfcarida @vinicius caridá @vfcarida @vinicius caridá @vfcarida