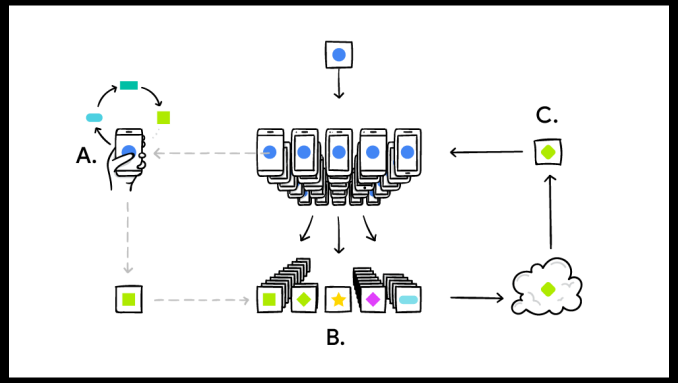
MORPH Guest Lecture

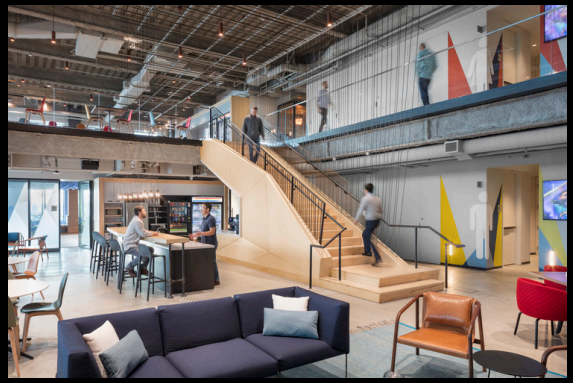
Implementing Machine Learning

Mentor: Eric Lin

Hi! I’m Eric


Done
SALE
Dragon Gate
Landmark · 43 mi

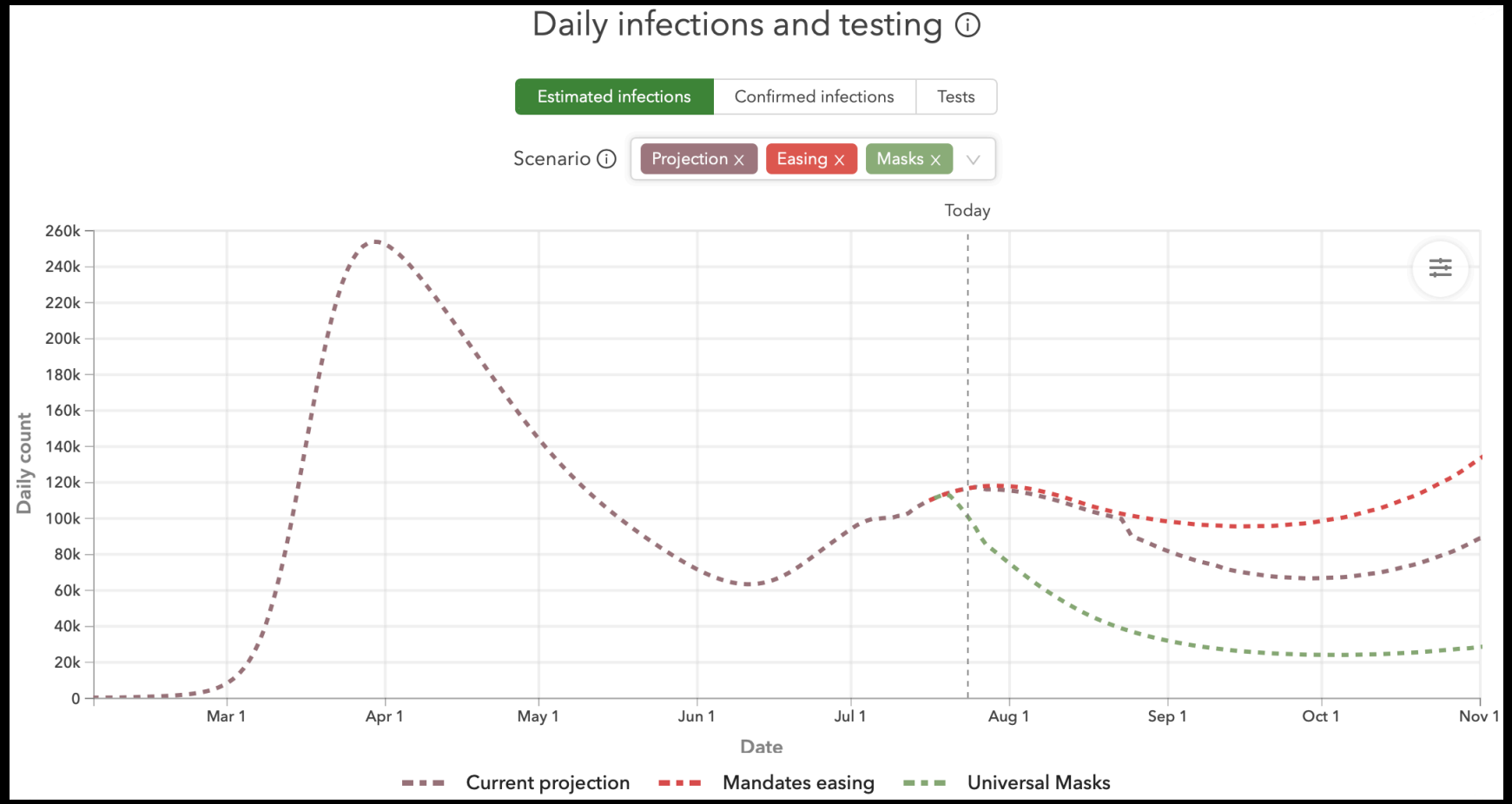

A.
B.
C.



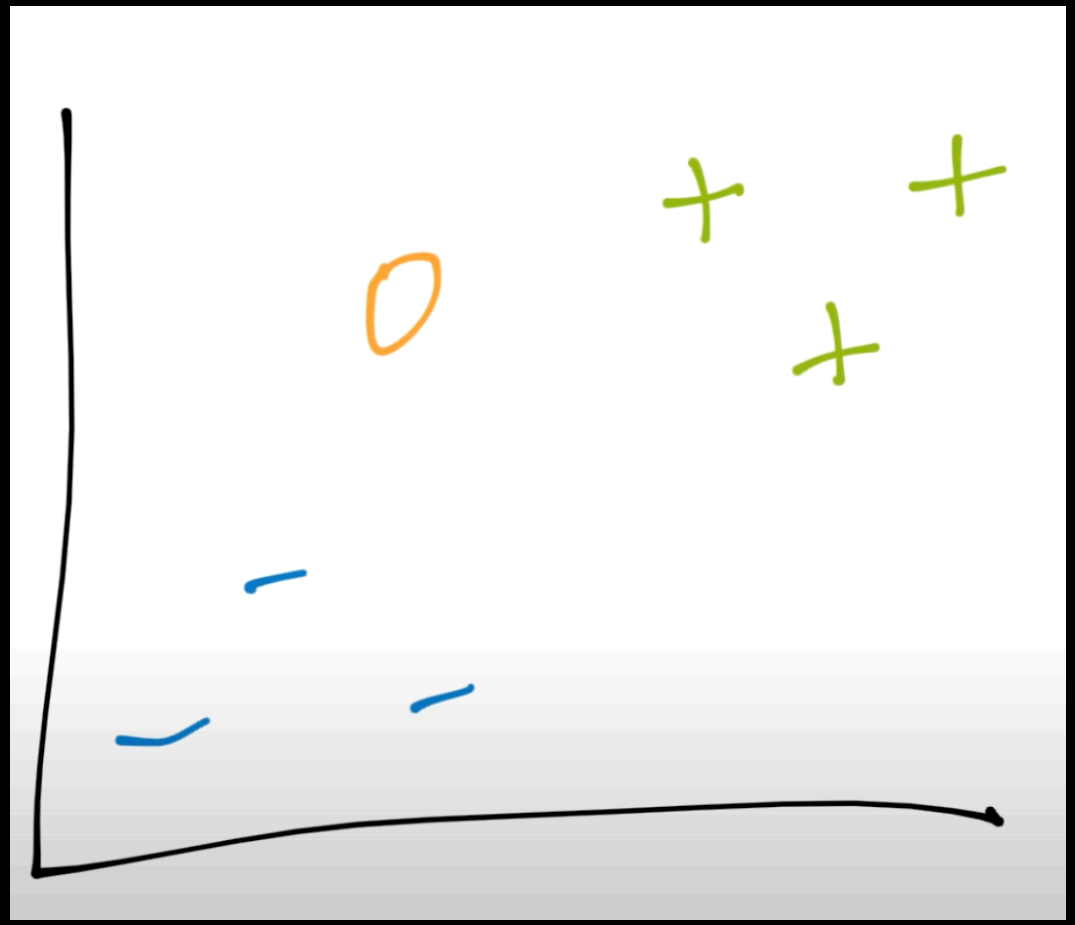
3 Archetypes of Machine Learning

1. Supervised
   - Given a dataset of points with correct output
2. Un-supervised
   - Input data without “correct” output
3. Reinforcement
   - Encouragement when actions align with positive behavior

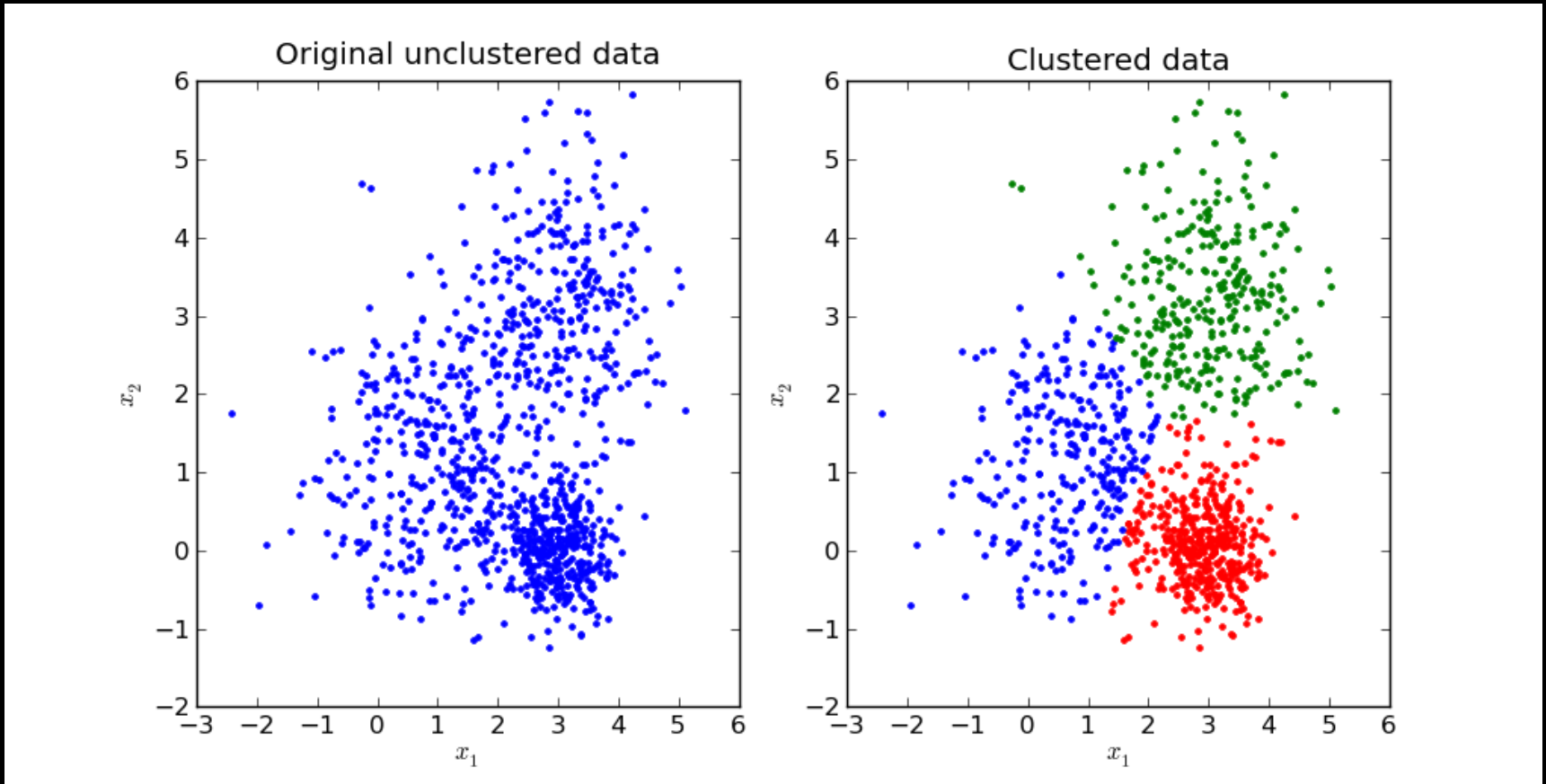
Supervised Learning: Regression



Supervised Learning: Classification



Unsupervised Learning: Clustering



Data Science Workflow

1. Formalizing problem statement and approach
2. Setting up / building data pipeline
3. Collecting data
4. Cleaning data
5. Preprocessing data
6. Exploring data
7. Conducting analysis
8. Validating results
9. Communicating findings

Data Science Workflow

1. Formalizing problem statement and approach
2. Setting up / building data pipeline
3. Collecting data
4. Cleaning data
5. Preprocessing data
6. Exploring data
7. Conducting analysis
8. Validating results
9. Communicating findings

Data Science Workflow

1. Formalizing problem statement and approach
2. Setting up / building data pipeline
3. Collecting data
4. Cleaning data
5. Preprocessing data
6. Exploring data
7. Conducting analysis
8. Validating results
9. Communicating findings

How do we implement ML?

Data Science Workflow

1. Formalizing problem statement and approach
2. Setting up / building data pipeline
3. Collecting data
4. Cleaning data
5. Preprocessing data
6. Exploring data
7. Conducting analysis
8. Validating results
9. Communicating findings

Learning Frameworks

```
import torchvision.transforms as transforms
from torch.utils.data.sampler import SubsetRandomSampler
import matplotlib.pyplot as plt

#Using MNIST
#dataset = datasets.MNIST(root='./data')
#idx = dataset.train_labels==1
#dataset.train_labels = dataset.train_labels[idx]
#dataset.train_data = dataset.train_data[idx]

# Using CIFAR-10
# Load training data
transform_train = transforms.Compose([
    transforms.RandomCrop(32, padding=4),
    transforms.RandomHorizontalFlip(),
    transforms.ToTensor(),
    transforms.Normalize((0.4914, 0.4822, 0.4465), (0.2023, 0.1994, 0.2010)),
])
trainset = torchvision.datasets.CIFAR10(root='./data', train=True,
                                        download=True,
                                        transform=transform_train)
#Edit and Load validation data
validset = torchvision.datasets.CIFAR10(root='./data', train=True,
                                        download=True,
                                        transform=transform_train)
valid_size = 0.2
indices = list(range(len(trainset)))
split = int(np.floor(valid_size * len(trainset)))

if True:
    np.random.shuffle(indices)

train_idx, valid_idx = indices[split:], indices[:split]
train_sampler = SubsetRandomSampler(train_idx)
valid_sampler = SubsetRandomSampler(valid_idx)

trainloader = torch.utils.data.DataLoader(trainset, batch_size=128,
                                          sampler=valid_sampler, shuffle=False,
```

ML Services and Tools

IBM Cloud

Microsoft Azure

amazon web services

Google Cloud Platform

Real world use case…

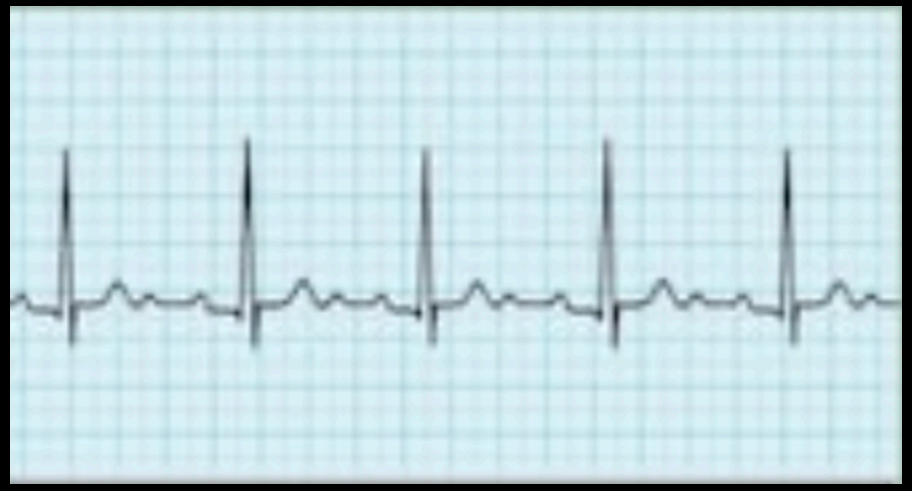
Predicting heart attacks

- Need
  - Predict if person is at risk of heart attack
- Given
  - ECG rhythm (electric signals)

A ML Solution at a High Level

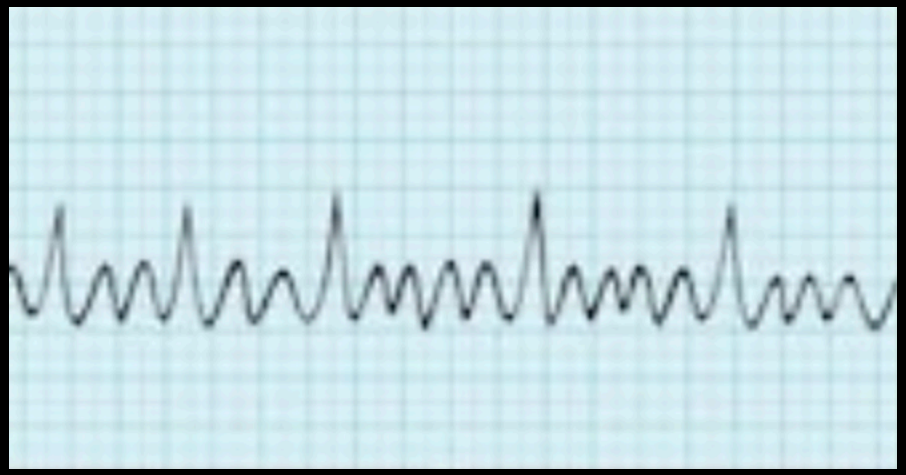
- ML model to classify ECGs as either “at risk” or “healthy”
  - E.g. AdaBoost or Convolutional neural network
- High level solution overview
  - Collects ECGs, labeled with either “at risk” or “healthy”
  - Looks at a piece of data: ECG + label
  - Finds patterns between ECGs
  - Trains model based on this data
  - Updates model
  - Repeat

Variables: Beats per minute



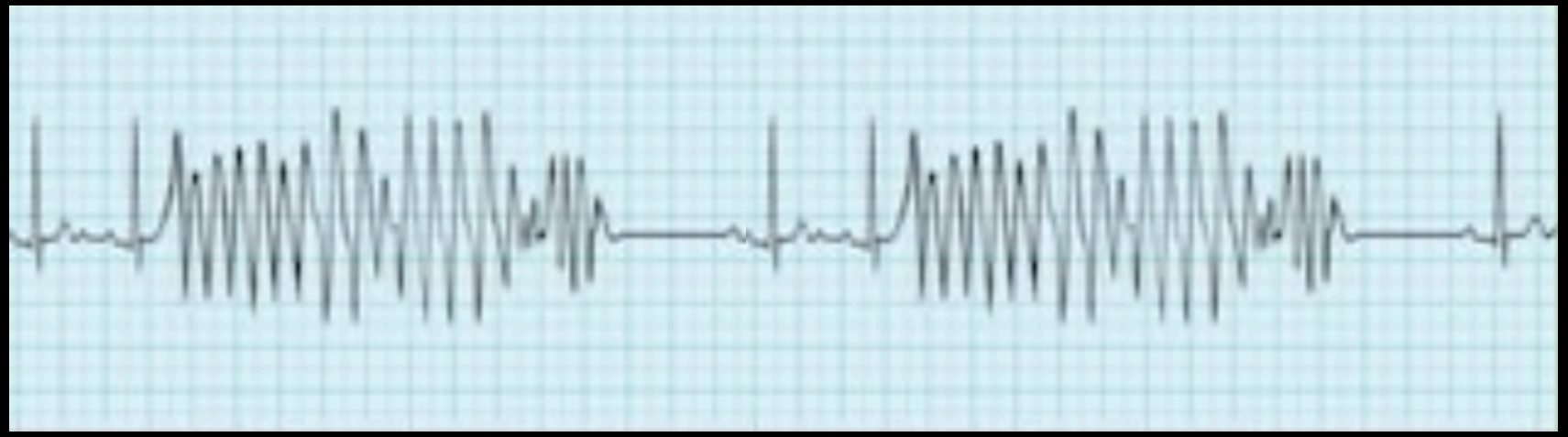
Normal

Variables: Beats per minute, seconds between beats



At risk
(atrial fibrillation)

Variables: Beats per minute, spacing between beats, irregularities per minute



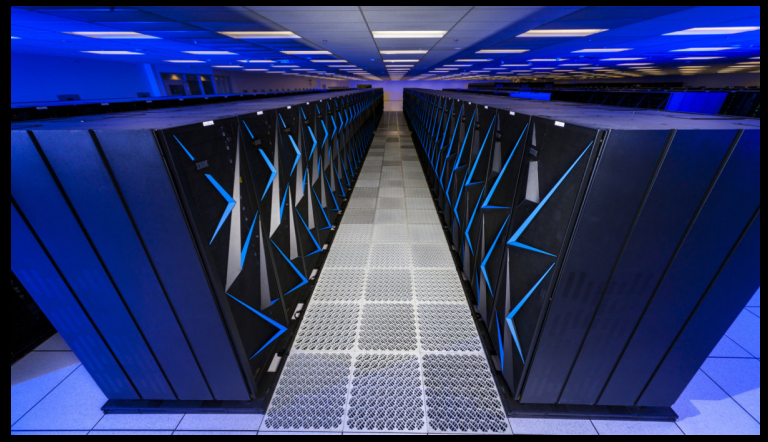
At risk
(ventricular tachycardia)

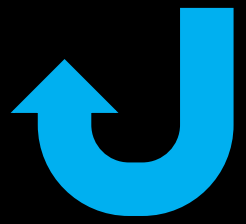
Variables: Beats per minute, seconds between beats, irregularities per minute

Normal
(ventricular tachycardia)

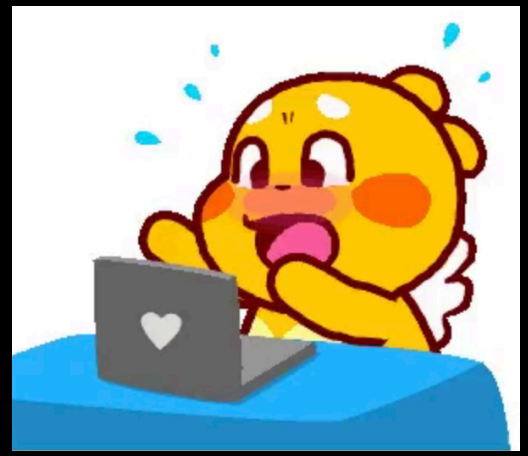
Central Server Model

Data





Model Training



Programmer


10:09
Back
ECG Detail
0s
1s
2s
Sinus Rhythm
88 BPM Average
This ECG does not show signs of atrial fibrillation.
Export a PDF for Your Doctor
If you believe you're having a heart attack or a medical emergency, call emergency services.
SAMPLE DETAILS
Start Time
Sep 12, 2018 at 10:08:31 AM
End Time
Sep 12, 2018 at 10:09:01 AM
Source
ECG
Today
Health Data
Sources
Medical ID
76 BPM
10:09
24sec
It helps to rest your arms on a table or your legs.

Predicting heart attacks

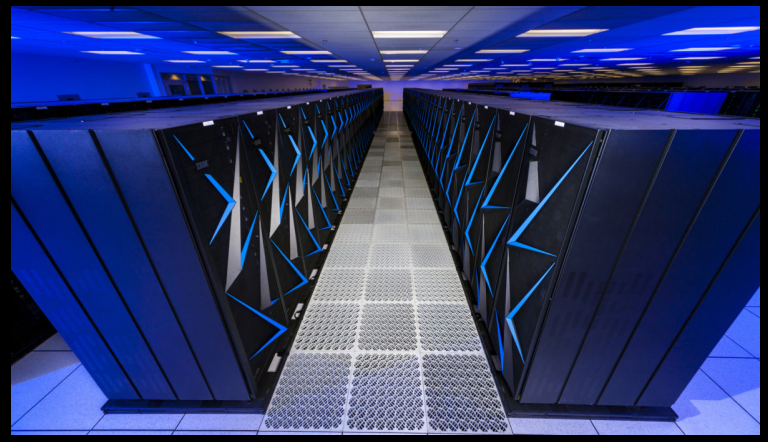
- Need
  - Predict if person is at risk of heart attack
- Given
  - ECG rhythm (electric signals)

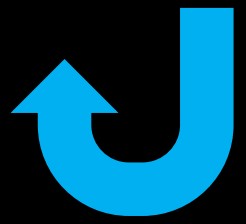
Predicting heart attacks

- Need
  - Predict if person is at risk of heart attack
- Given
  - ECG rhythm (electric signals)
- Obstacle
  - As a patient, I don’t want my data to be stored on Apple’s servers.

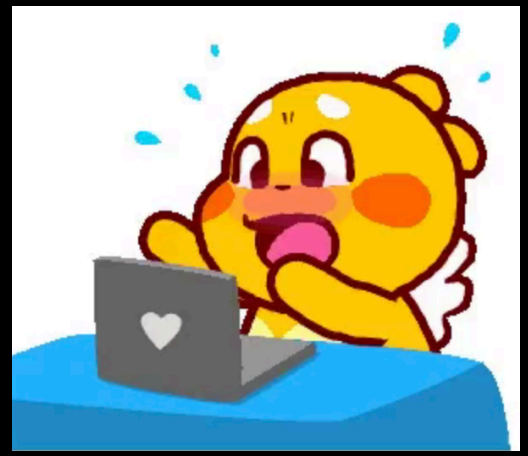
Central Server Model

Data





Model Training

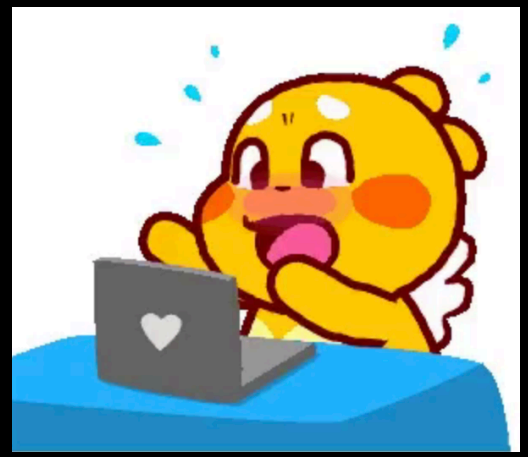


Programmer

Central Server Model

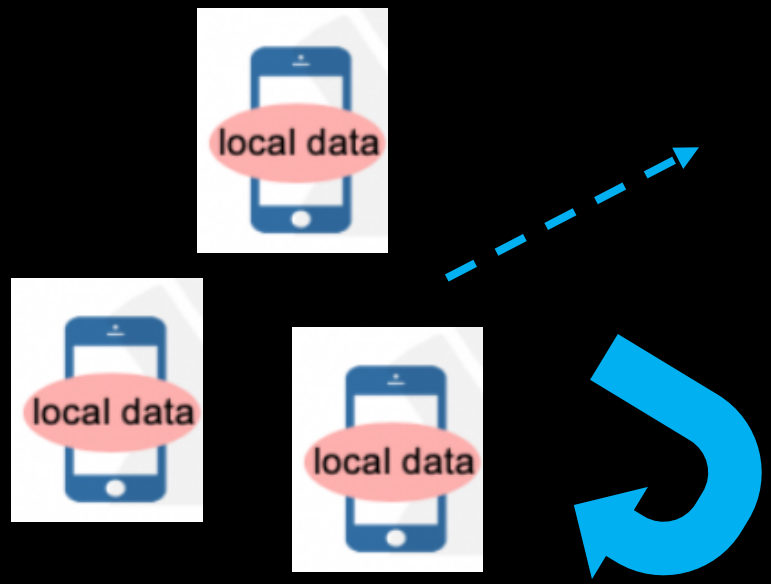
Data

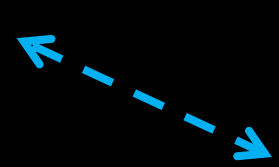
Model Training

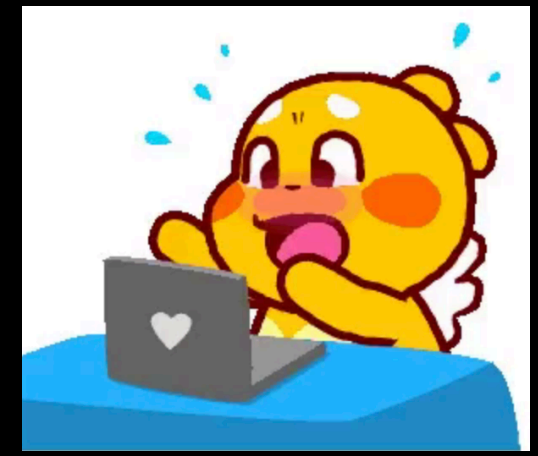


Programmer

Federated Learning Model





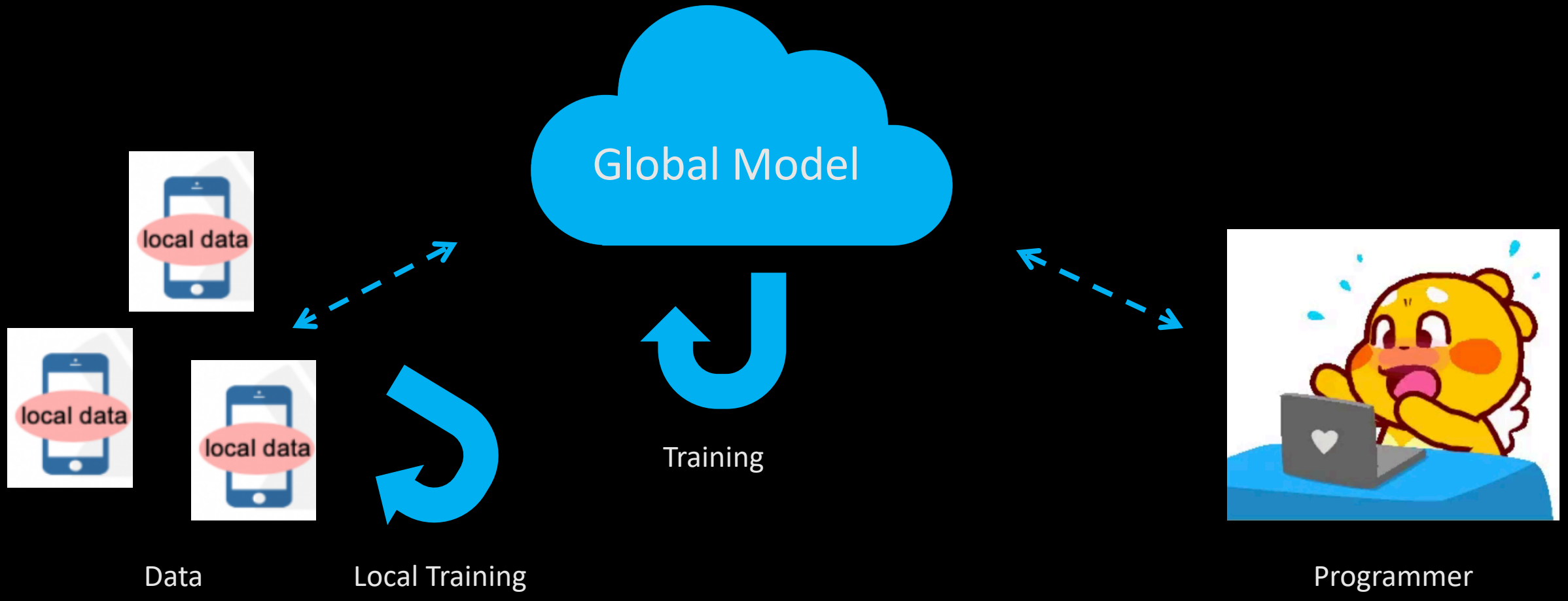


Data

Model Training

Programmer

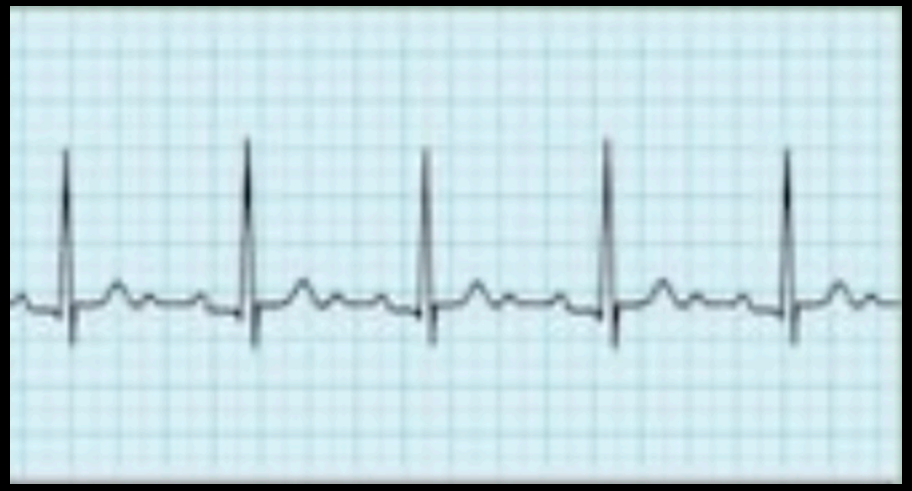
Federated Learning Model



Federated Learning

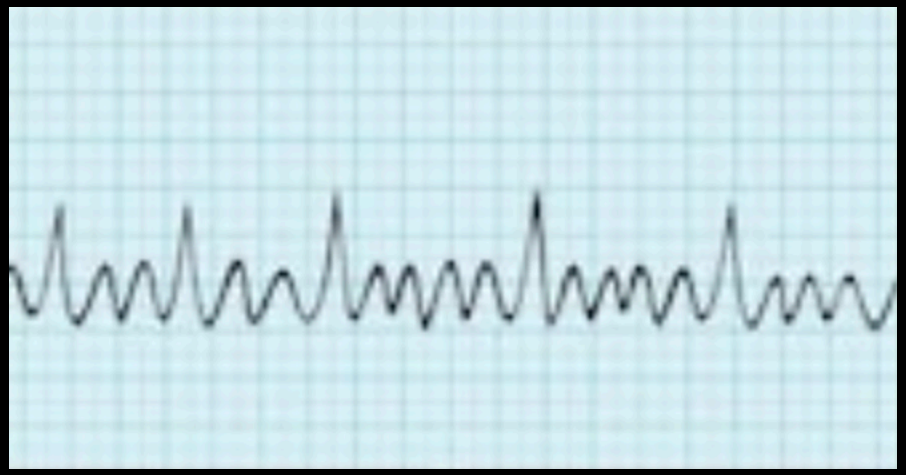
- Data stays on edge devices (e.g. phones)
- Maintain a global model shared among edge devices
- Training occurs on edge devices
- Edge devices send updates to global model
- Global model sends the total updated model to all edge devices
  - All devices benefit from the learnings from all data

Variables: Beats per minute



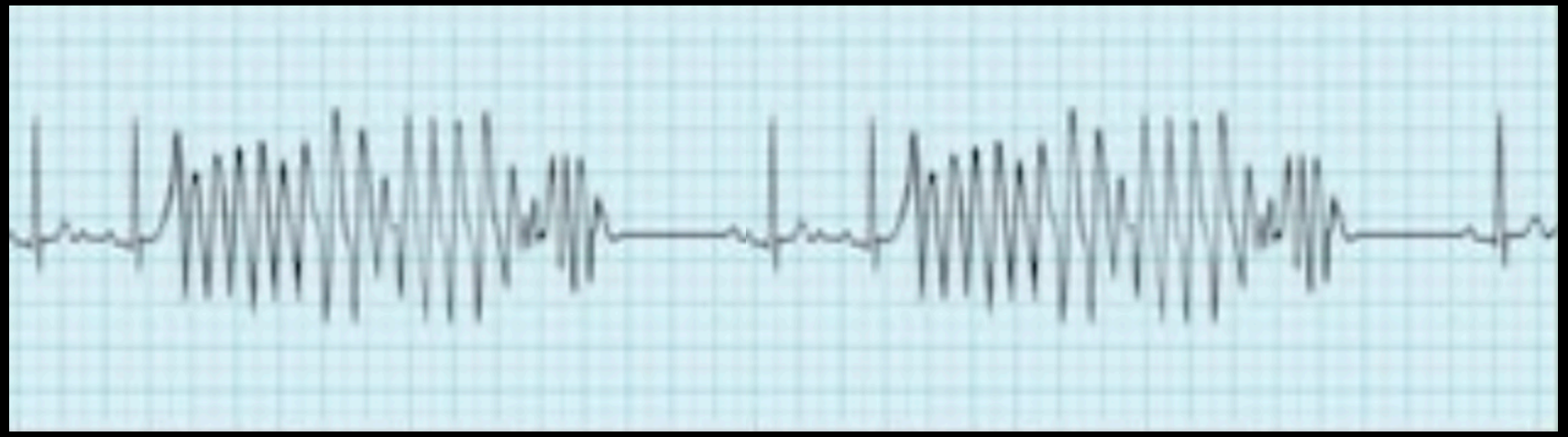
Normal

Variables: Beats per minute, seconds between beats



At risk
(atrial fibrillation)

Variables: Beats per minute, seconds between beats, irregularities per minute



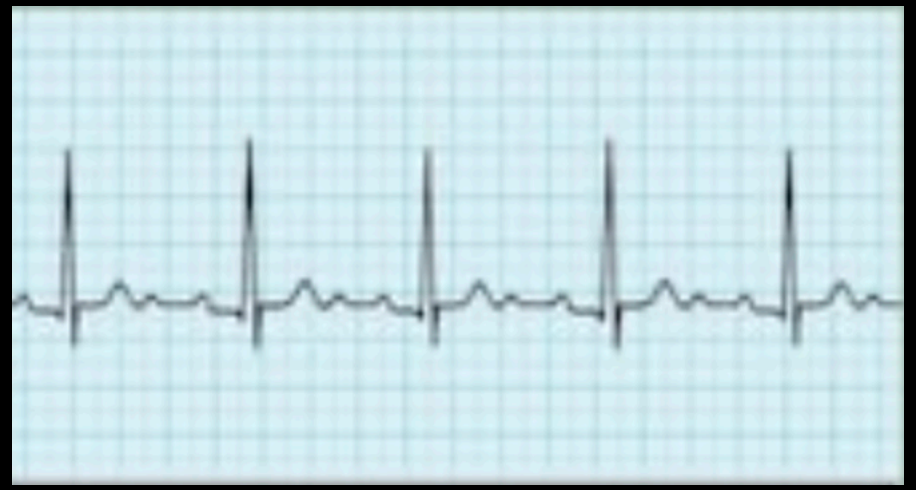
At risk
(ventricular tachycardia)

Variables: Beats per minute, seconds between beats, irregularities per minute

Normal
(ventricular tachycardia)

Federated Averaging

- Variables and weights
  - Beats per minute (w1)
  - Seconds between beats (w2)
  - Irregularities per minute (w3)
- Example model
  - Y = w1 * bpm + w2 * spacing + w3 * irregularities
  - If Y > 1 then “at risk”. Else “healthy”.
- Average weights
  - Device 1: w1 = 0.07, w2 = 0.03, w3 = 0.116
  - Device 2: w1 = 0.09, w2 = 0.01, w3 = 0.122
  - Average: w1 = 0.08, w2 = 0.02, w3 = 0.119



Final Thoughts

- Machine Learning can be incredibly beneficial
- Machine Learning can be incredibly harmful
  - Geo-location tracking

Recap

- Data Science Workflow
- ML Tools and Frameworks
- Example use case
- Federated Learning
- Final thoughts

Questions?