
outrageously
AMBITIOUS
Module 4:
ML System Design &
Technology Selection
Duke
PRATT SCHOOL of
ENGINEERING

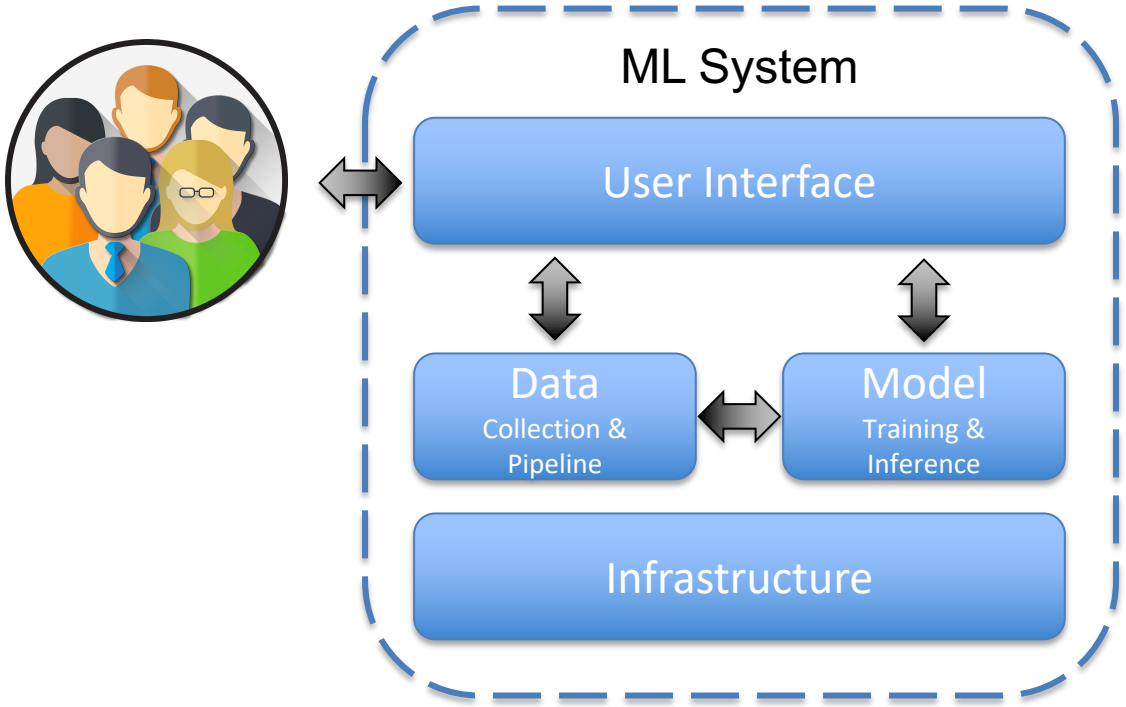
# Module 4 Objectives:

**At the conclusion of this module, you should be able to:**

1) Describe the key technology decisions involved in designing ML systems
2) Identify the main criteria to consider in making technology selection decisions
3) Explain the commonly used tools among data scientists and ML practitioners

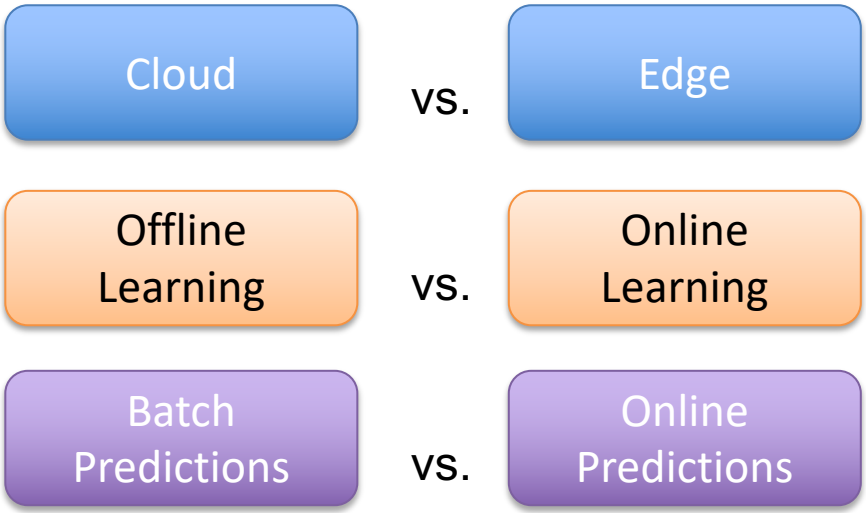

outrageously
AMBITIOUS
ML System Design Considerations
Duke
PRATT SCHOOL of ENGINEERING

# What is a ML system?



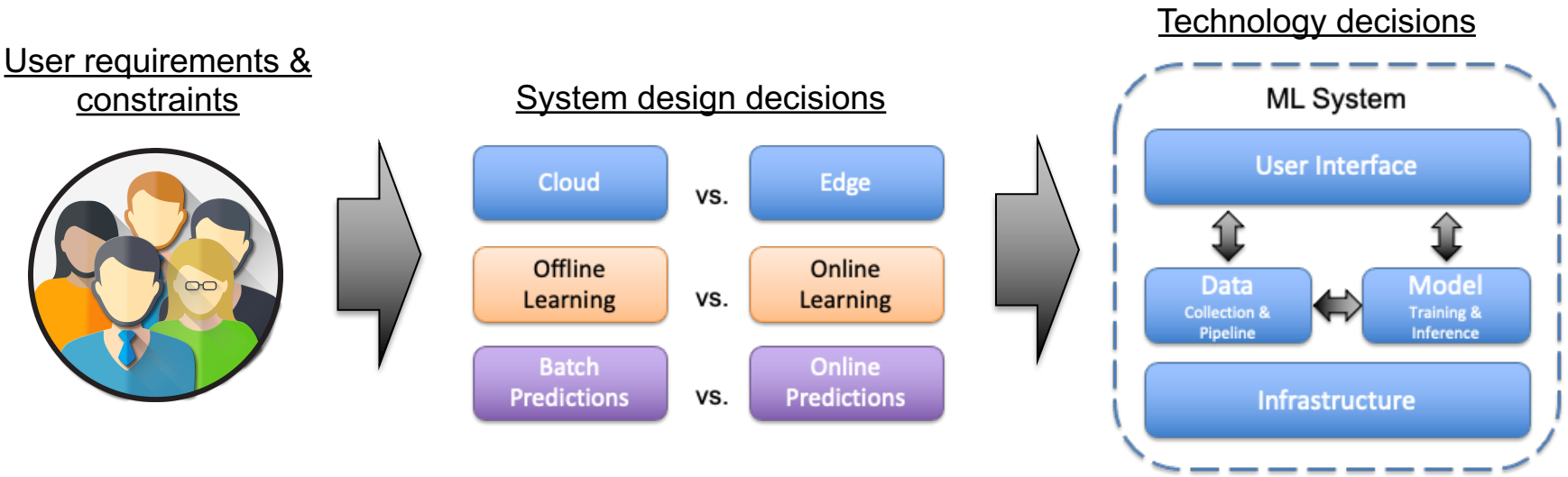
# ML system design decisions

There are several system design decisions which impact the choice of technologies:

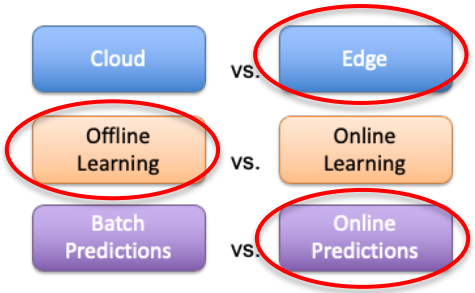
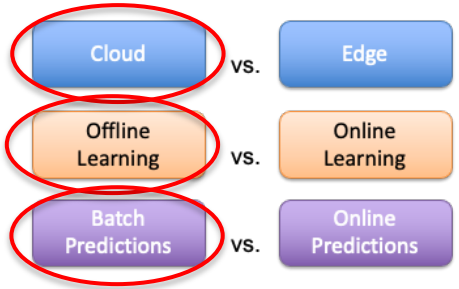


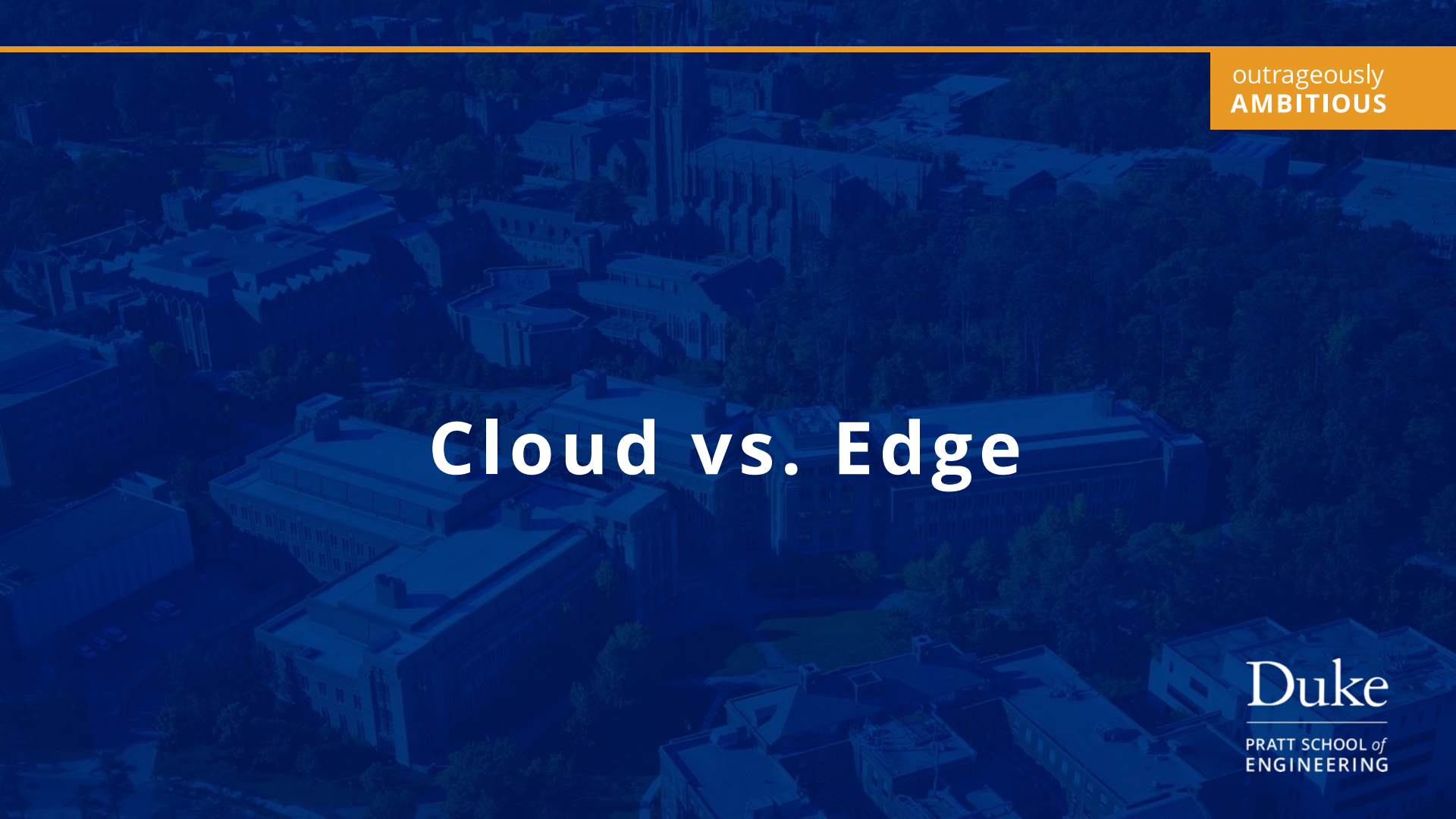
# System design process

- User requirements and constraints drive system design
- System design drives selection of technologies



# System design examples

| | Use Case | Requirements / Constraints | System design |
|---|---|---|---|
| Example 1 | • Unlocking a phone using a facial recognition model | • Low latency<br>• Cannot require connectivity<br>• Privacy concerns |  |
| Example 2 | • Movie recommendation engine | • Assume connectivity<br>• High throughput<br>• Low latency |  |


outrageously
AMBITIOUS
Cloud vs. Edge
Duke
PRATT SCHOOL of
ENGINEERING

# Edge AI

- Running ML on devices themselves
- Predicted to grow by 25% per year[1]
- Enabled by hardware & software advances
  - Solving power constraints
  - Increased edge computational power
  - Smaller models designed for the edge
- Why edge AI?  Eliminates latency
  - Every 100 miles distance from datacenter introduces a latency of  > 1.6 milliseconds[2]

1) *Valuates, https://reports.valuates.com/market-reports/QYRE-Auto-4139/global-edge-ai-software*
2) *Techwalla, https://www.techwalla.com/articles/network-latency-milliseconds-per-mile*

# Cloud vs. edge

| | Cloud ML | Edge ML |
|---|---|---|
| **Description** | Computations done on cloud and result delivered to end device | Computations done directly on device (phone, sensor, etc) |
| **Requirements** | Network connectivity | Sufficient compute power, memory |
| **Benefits** | High throughput | Low latency, privacy, no need for connectivity |
| **Examples** | • Chatbots<br>• Demand prediction | • Quality control<br>• Autonomous driving |

# Cloud ML example

Movie recommendation system

# Edge ML example

Intelligent security system

# Hybrid approaches

- Initiate cloud ML through trigger generated by edge AI
- Store common pre-computed predictions on device
- Exploit many local datacenters/servers to minimize latency

# Hybrid example

Smart spaker with voice assistant

# Cloud vs. edge AI

- How much does latency matter?
  - A lot -> Edge
  - Not much -> Cloud
- Is reliance on internet connectivity acceptable?
  - No -> Edge
  - Yes -> Cloud
- Are users comfortable sending their data to the cloud?
  - No -> Edge
  - Yes -> Cloud


outrageously
AMBITIOUS
Online Learning & Inference
Duke
PRATT SCHOOL of
ENGINEERING

# Offline vs. online models

An important design consideration is whether model training & prediction can be scheduled or must be real-time

| | Scheduled | Real-time |
|---|---|---|
| **Model re-training** | Offline learning | Online learning |
| **Prediction** | Batch prediction | Online prediction |

# Offline vs. online learning

| | Offline learning |
|---|---|
| **Description** | Model re-training done on a schedule (weeks/months) using datapoints in many iterations |
| **Benefits** | • Easier to implement in production<br>• Easier to evaluate |
| **Challenges** | • Slower to adapt to changes in environment or data distribution |
| **Examples** | • Most current applications |

# Offline vs. online learning

| | Offline learning | Online learning |
|---|---|---|
| **Description** | Model re-training done on a schedule (weeks/months) using datapoints in many iterations | Continual re-training as new data arrives (mins/hours) using each new datapoint once |
| **Benefits** | • Easier to implement in production<br>• Easier to evaluate | • Handles big data<br>• Real-time adaptation to changing environment |
| **Challenges** | • Slower to adapt to changes in environment or data distribution | • Harder to implement & evaluate performance |
| **Examples** | • Most current applications | • Flagging spam in social media |

# Online learning example

News site with personalized recommendations

# Batch vs. online prediction

| | Batch prediction |
|---|---|
| **Description** | Generate predictions on batch of observations on a recurring schedule |
| **Benefits** | • Leverage more efficient operations and technologies<br>• Easier monitoring of drift |
| **Challenges** | • Predictions not immediately available for new data |
| **Examples** | • Recommendation systems<br>• Demand prediction |

# Batch vs. online prediction

| | Batch prediction | Online prediction |
|---|---|---|
| **Description** | Generate predictions on batch of observations on a recurring schedule | Real-time predictions generated upon request |
| **Benefits** | • Leverage more efficient operations and technologies<br>• Easier monitoring of drift | • Predictions available immediately |
| **Challenges** | • Predictions not immediately available for new data | • Minimizing latency<br>• Monitoring of model drift |
| **Examples** | • Recommendation systems<br>• Demand prediction | • Translation app<br>• Autonomous vehicles |

# Online prediction example

Food delivery “time-to-arrive”


outrageously
AMBITIOUS
ML on Big Data
Duke
PRATT SCHOOL of
ENGINEERING

# Examples of Big Data

Sensor/device data

Social media data

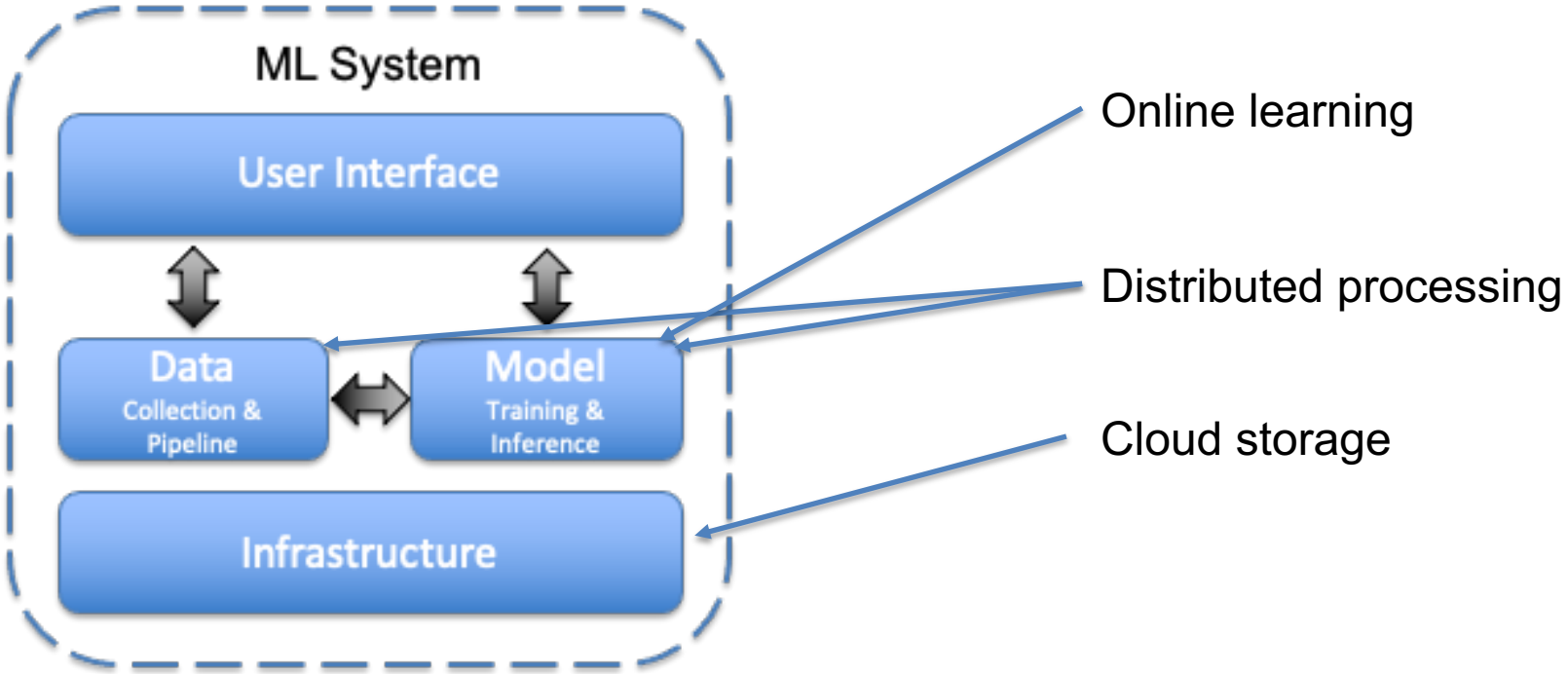

Healthcare data

# ML on Big Data

Working with Big Data has unique challenges:

- **Storage & processing**
  - Massive storage needs
  - Processing computational costs
- **Exploration**
  - Understanding quality, patterns & relationships
- **Modeling**
  - Data is too large to fit in memory
  - Long model training cycles

# Impacts on ML system design




outrageously
AMBITIOUS
ML Technology Selection
Duke
PRATT SCHOOL of
ENGINEERING

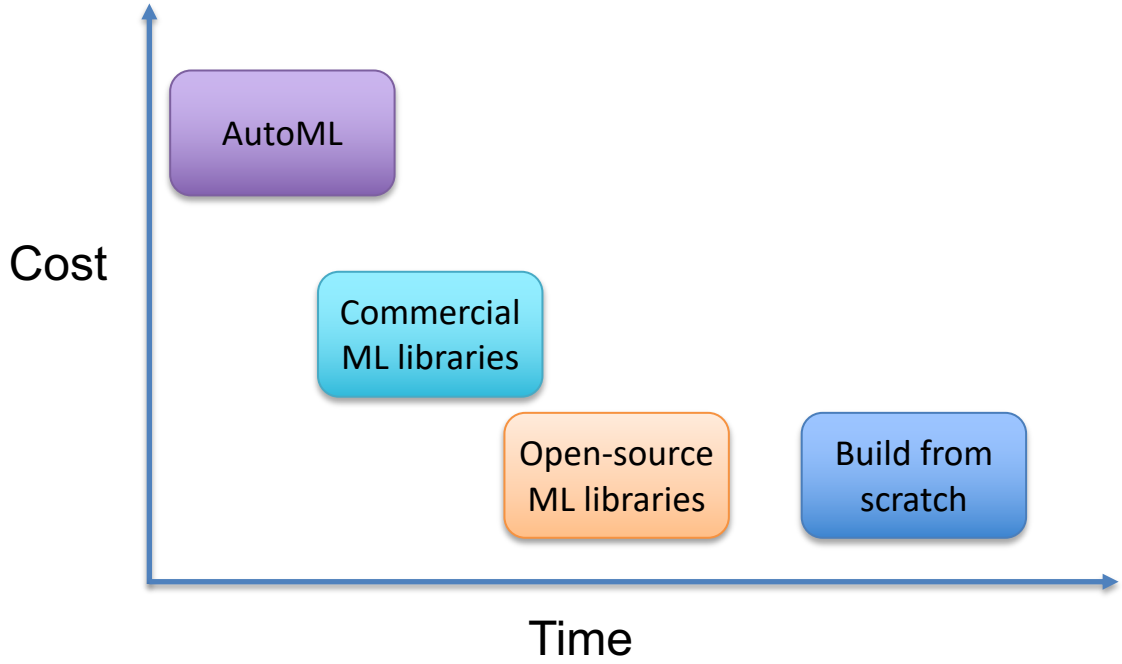
# ML technology decisions

- Programming language
- Data processing toolset
- Modeling toolset
- API / interface

# Factors in technology selection

- Open source vs. proprietary
- Learning curve
- Documentation – docs, tutorials etc.
- Community support & talent
- Flexibility vs. simplicity
- Lifetime cost

# ML technology options




outrageously
AMBITIOUS
Common ML Tools
Duke
PRATT SCHOOL of
ENGINEERING

# Programming languages

## Python

- Easy to learn, highly readable
- Rich ecosystem of data science / ML libraries
- Widely available help resources
- Very strong community

## R

- Specialized for statistical and graphical analysis
- Good ecosystem of data analysis libraries
- Excellent plotting capabilities
- Good community

## C/C++

- Common choice for software dev teams
- Faster than other languages
- Higher learning curve
- Less commonly used for ML

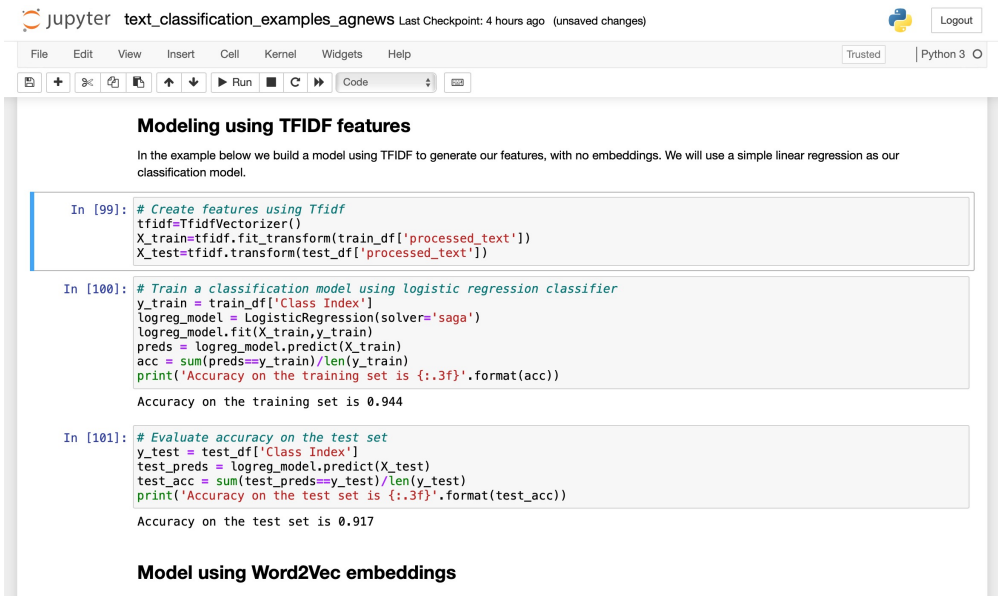
# Language popularity

PYPL Popularity of Programming Index

| Rank | Language | Share | 1yr Trend |
|---|---|---|---|
| 1 | Python | 30.6% | -1.1% |
| 2 | Java | 17.2% | -0.1% |
| 3 | JavaScript | 8.3% | +0.3% |
| 4 | C# | 6.8% | 0% |
| 5 | C/C++ | 6.3% | +0.6% |
| 6 | PHP | 6.2% | +0.2% |
| 7 | R | 3.9% | -0.1% |

*Source: http://pypl.github.io/PYPL.html*

# Jupyter notebooks




- Great for experimentation
- Combine code (>40 languages), text and images in a single notebook
- Enable iterative development through individual cell execution
- More difficult to follow software engineering best practices – testing, versioning

# Python ecosystem

- **Pandas (and NumPy)**: data manipulation
  - Merge, manipulate, clean & analyze data using DataFrames
- **Sci-kit Learn**: modeling
  - Data analysis and classical machine learning
- **Matplotlib**: visualization
  - Visualize data from pandas and NumPy
- **NLTK, SpaCy**: text processing
  - Pre-process text and convert to features for modeling

# Deep learning libraries

**TensorFlow** | **PyTorch** | **Keras**

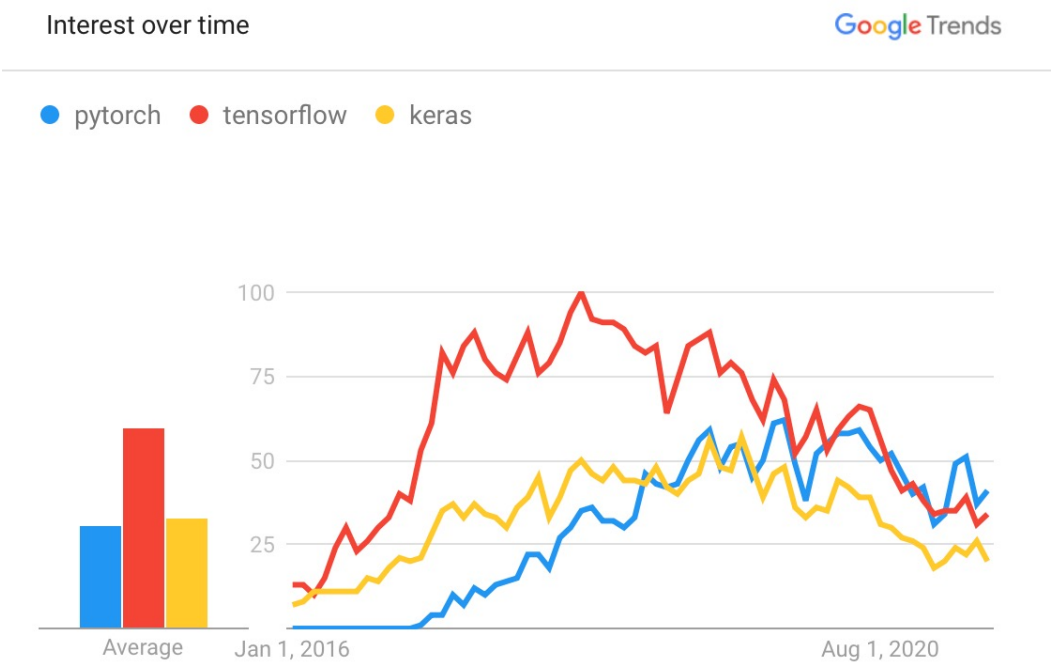
- Open-source Python libraries for building and training deep neural network models
- Can be used on CPUs, GPUs or TPUs for accelerated model training
- Export trained models to servers and devices for external use by applications

- Developed by Google
- More mature than PyTorch
- Historically the largest user community

- Developed by Facebook AI Research Lab
- Slightly easier learning curve than TensorFlow
- Growing popularity and user community

- Deep learning API on top of TensorFlow
- Simplifies model building process to enable quick iteration

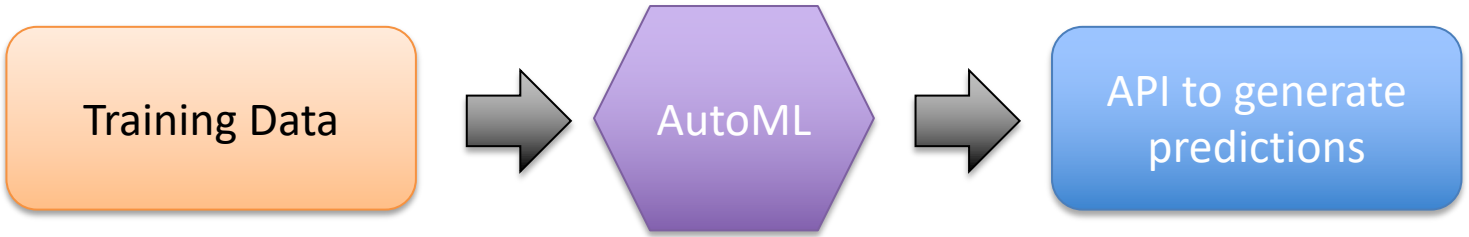
# Deep learning libraries




*Source: https://trends.google.com/trends/explore?date=2016-01-01%202021-06-23&geo=US&q=pytorch,tensorflow,keras*

# AutoML

- Enables developers with limited ML expertise to quickly build models with little/no code
- Automates selection of features, algorithm and hyperparameters for optimal performance
- Available from Google, Microsoft, AWS, H2O, etc.
- Variants for common ML tasks including NLP, computer vision




outrageously
AMBITIOUS
Wrap-up
Duke
PRATT SCHOOL of
ENGINEERING

# Wrap Up

- User requirements -> ML system design -> technology selection
- Important considerations include latency, connectivity, throughput, data size
- Many technology options
  - Open source vs proprietary
  - Manual vs automated