Machine Learning:
Past, Present and Future

Michael Soltys

March 2, 2022 @ RDP 21

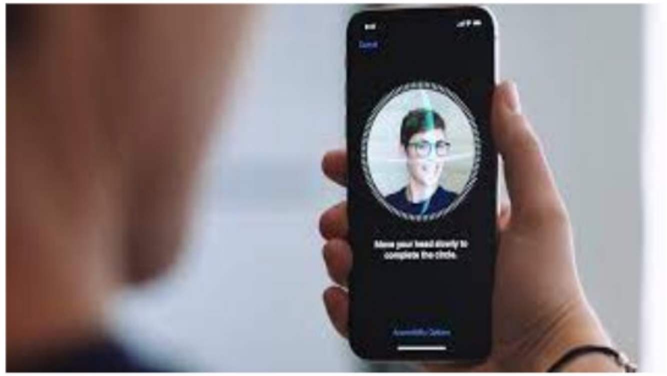
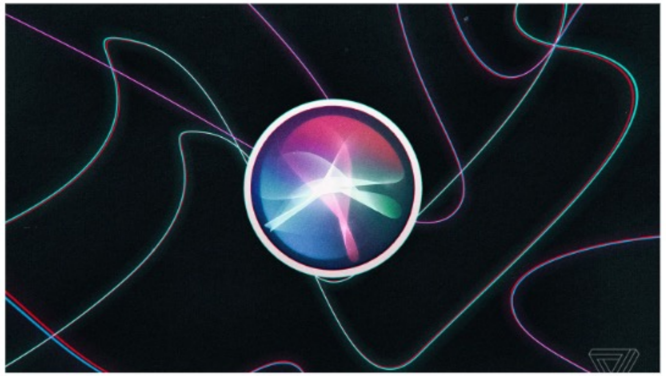
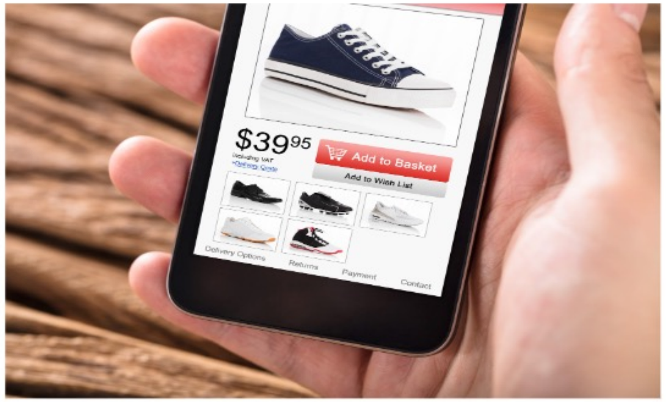


Image Recognition

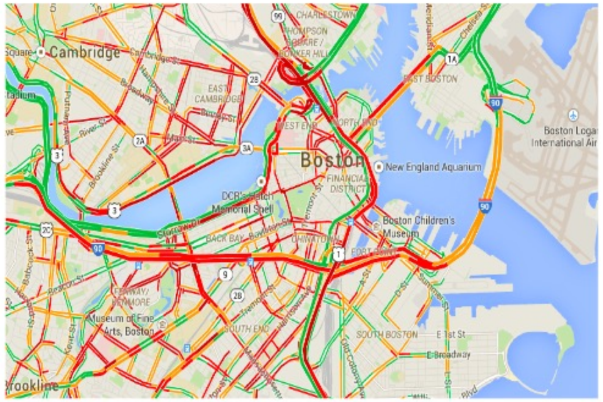

Speech Recognition

Fraud Detection


Self-Driving Cars



Product Recommendation



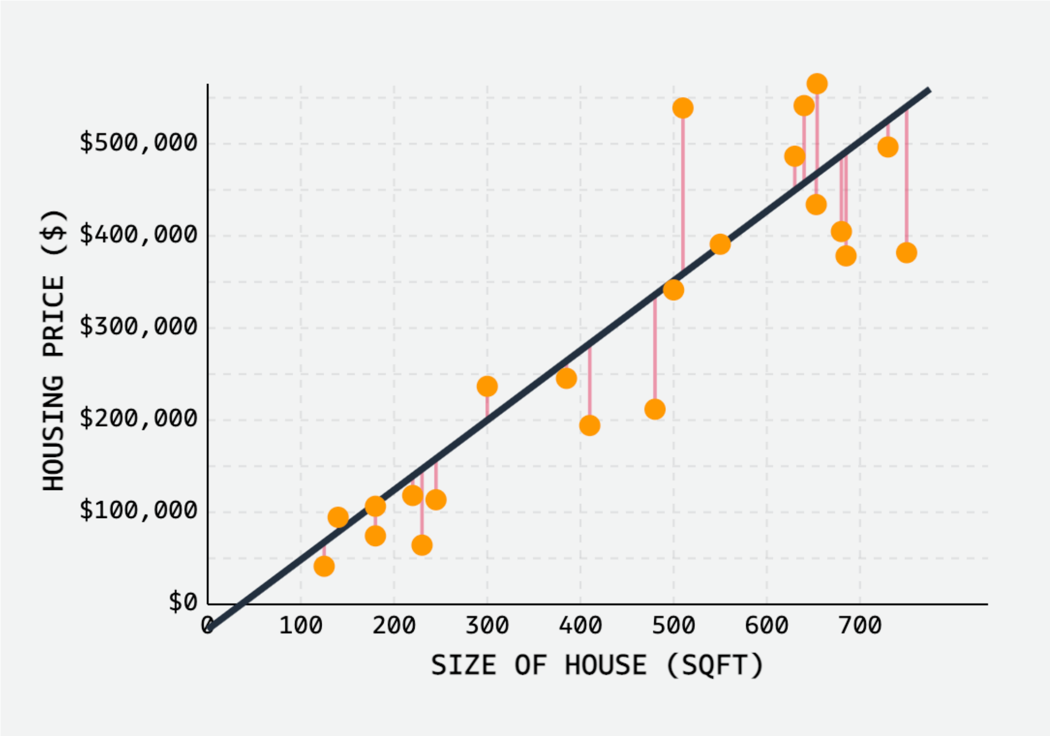
Traffic Prediction

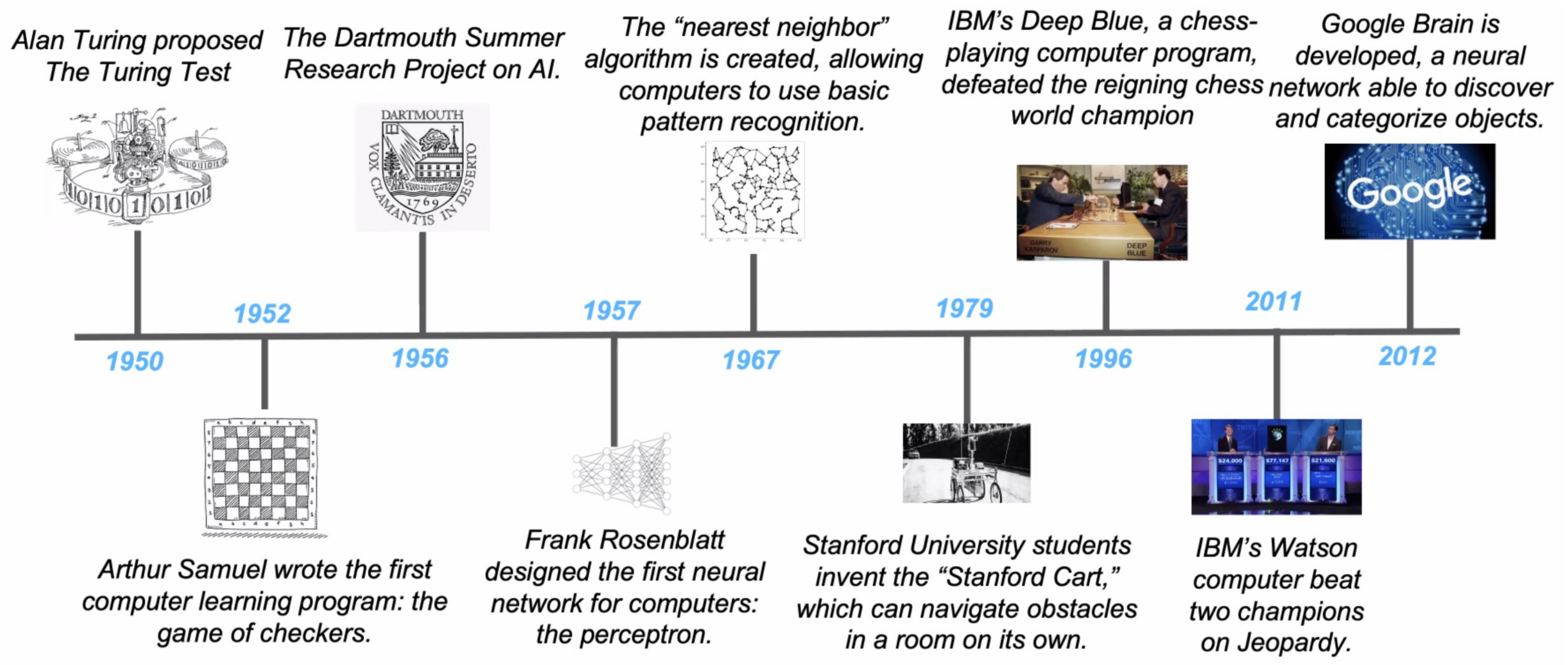
What is ML?

- ML is data-driven (as opposed to rule-driven) computation
- It is a subfield of AI (Artificial Intelligence)

Example: Linear Regression

- https://mlu-explain.github.io/linear-regression/



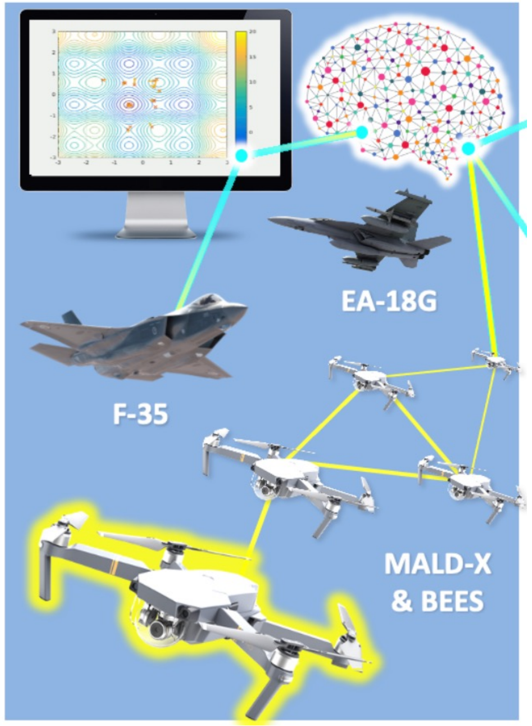


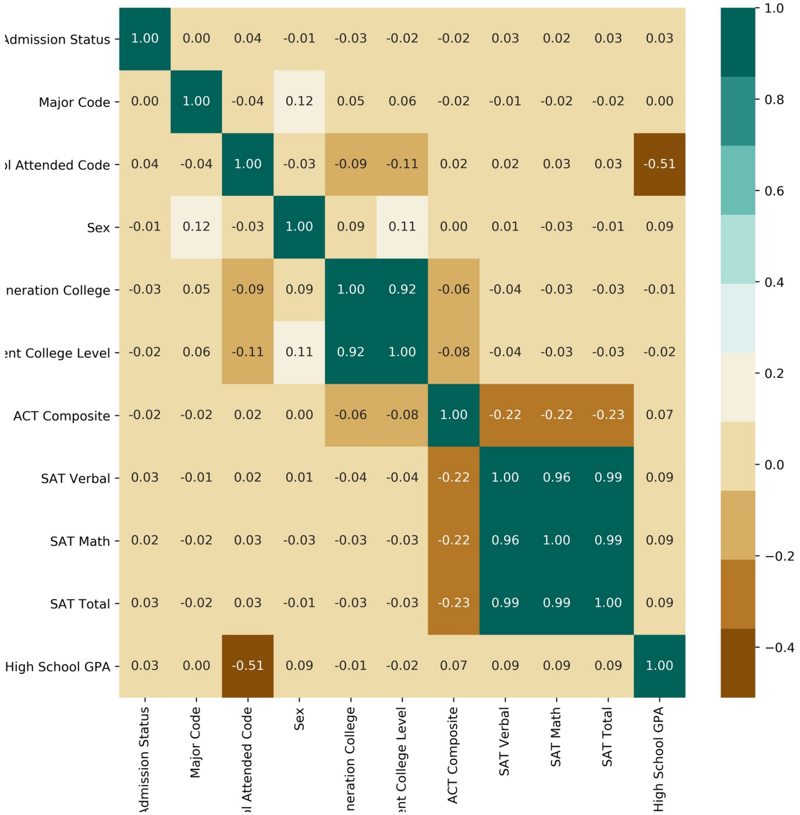
Alan Turing proposed The Turing Test
1950
1952
Arthur Samuel wrote the first computer learning program: the game of checkers.
The Dartmouth Summer Research Project on AI.
1956
1957
Frank Rosenblatt designed the first neural network for computers: the perceptron.
The "nearest neighbor" algorithm is created, allowing computers to use basic pattern recognition.
1967
1979
Stanford University students invent the "Stanford Cart," which can navigate obstacles in a room on its own.
IBM's Deep Blue, a chess-playing computer program, defeated the reigning chess world champion
1996
2011
IBM's Watson computer beat two champions on Jeopardy.
Google Brain is developed, a neural network able to discover and categorize objects.
2012

Example

SaigeTEC

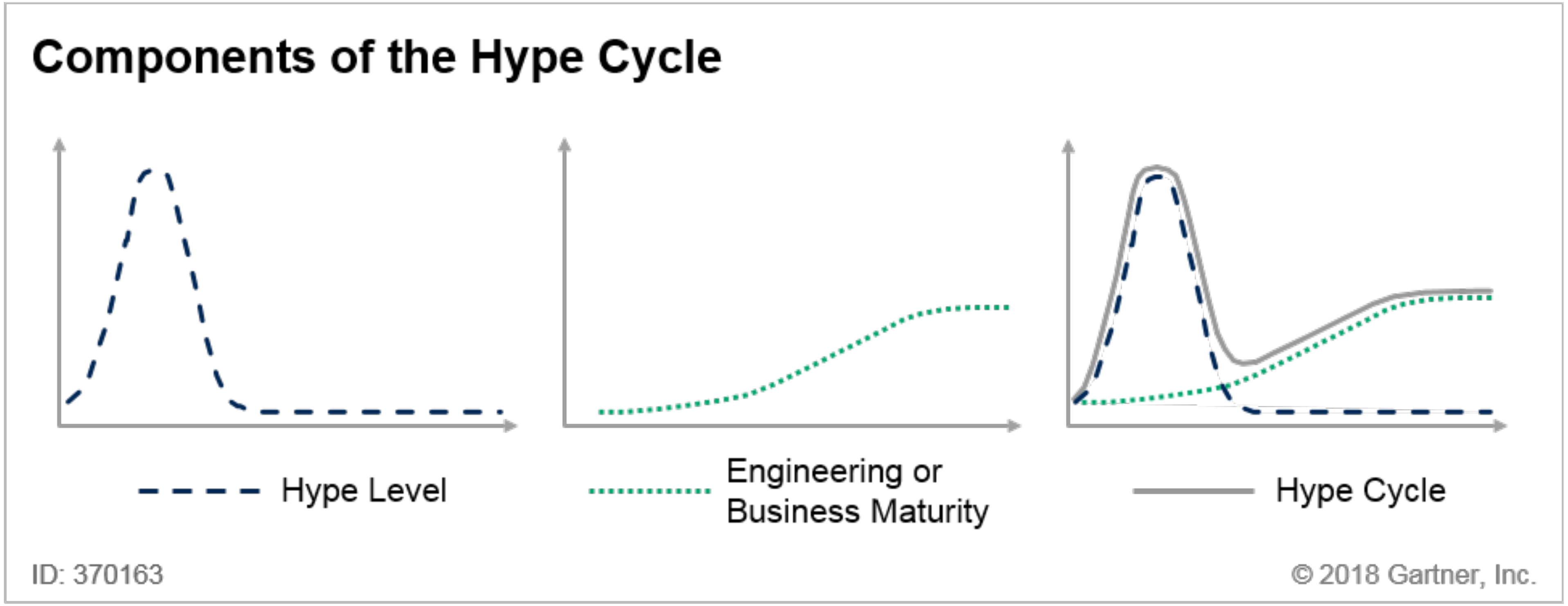
AI/Machine Learning Technologies



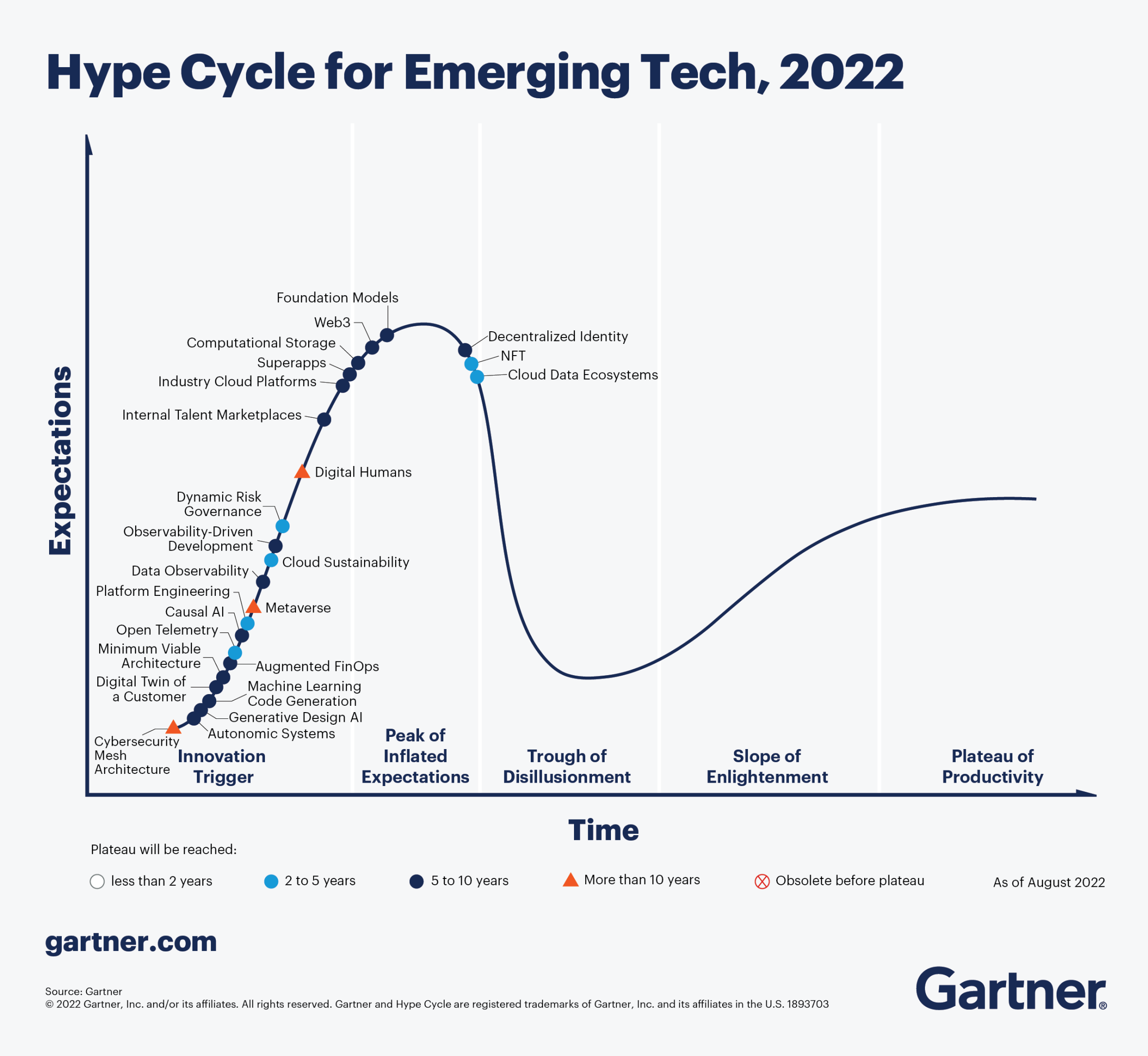


https://github.com/michaelsoltys/sagemaker-enrollment

Gartner Hype Cycle



Where is ML in the hype cycle?



Past
Theory and bespoke code

University of Toronto

Geoffrey Hinton was pioneering deep learning (1990-2015)

"All the News That's Fit to Print."

The New York Times.

THE WEATHER. Rain to-day; fair to-morrow.

VOL. LIII...NO. 16,774. NEW YORK, FRIDAY, OCTOBER 9, 1903.—FOURTEEN PAGES. ONE CENT

RUSSIAN FLEET AT KOREA?

Report That Ninety Warships Have Arrived at Masam-Pho.

Yesterday Part of Manchuria Should Have Been Evacuated, but No Move Has Been Made by the Russians There.

MAY SUSPEND BALKAN REVOLT.

LIFE INSURANCE AGENCY SOLD.

PRESIDENT GIVEN RESIGNS.

TO EXTRADITE GRAFTERS

Treaties Covering Bribery to be Negotiated at Once.

$150,000,000 MORTGAGE FILED.

MARRIED ON STOLEN MONEY.

CANADA'S EMERGENCY TROOPS.

DENIAL BY MORGAN & CO.

Statement of Connection With Shipbuilding Trust.

HOW SHARES WERE SOLD

M'LAUGHLIN DECLARES WAR ON M'CARREN

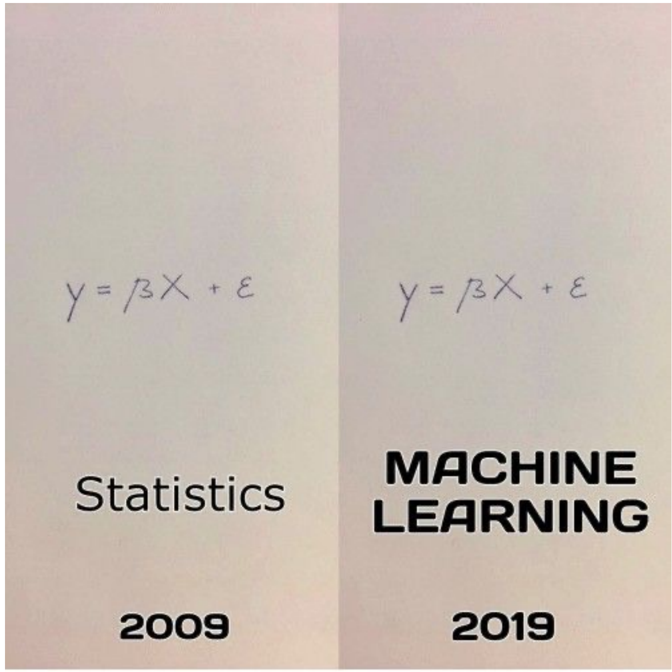
James Shevlin Calls the Senator a Political Acrobat.

EFFECT OF LITTLETON SPEECH

1903 NYT weeks before Wright brothers flew for first time:

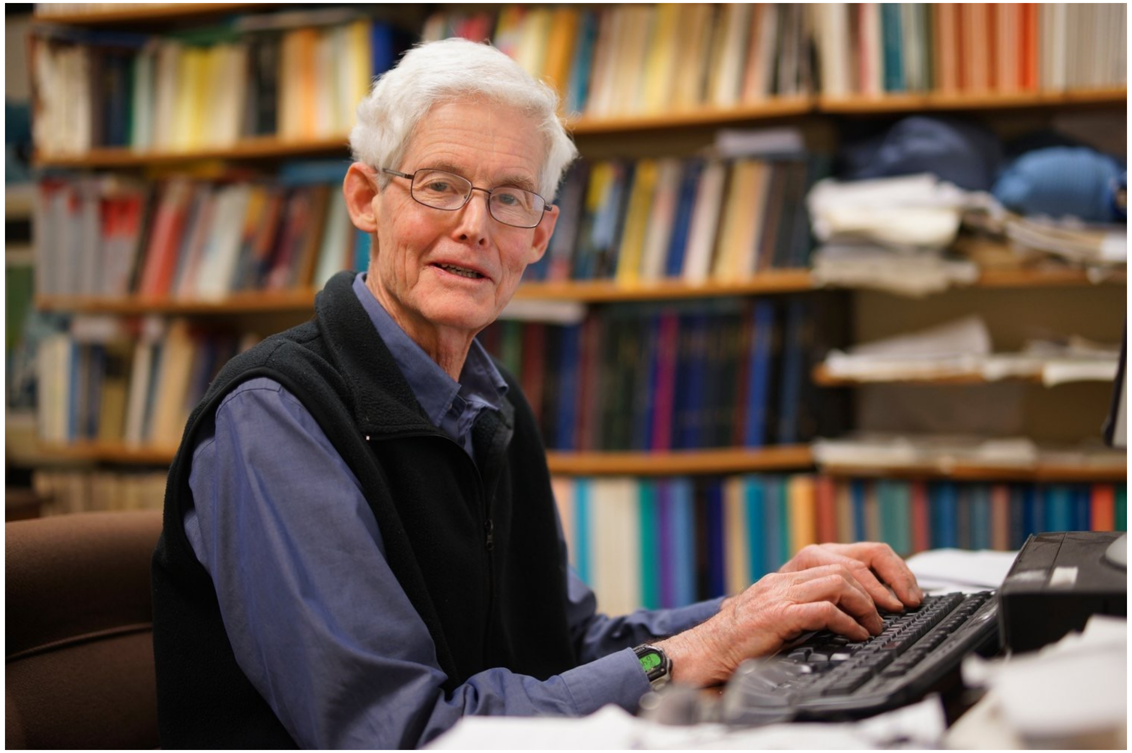
man will not fly for 10 million years

Mathematics



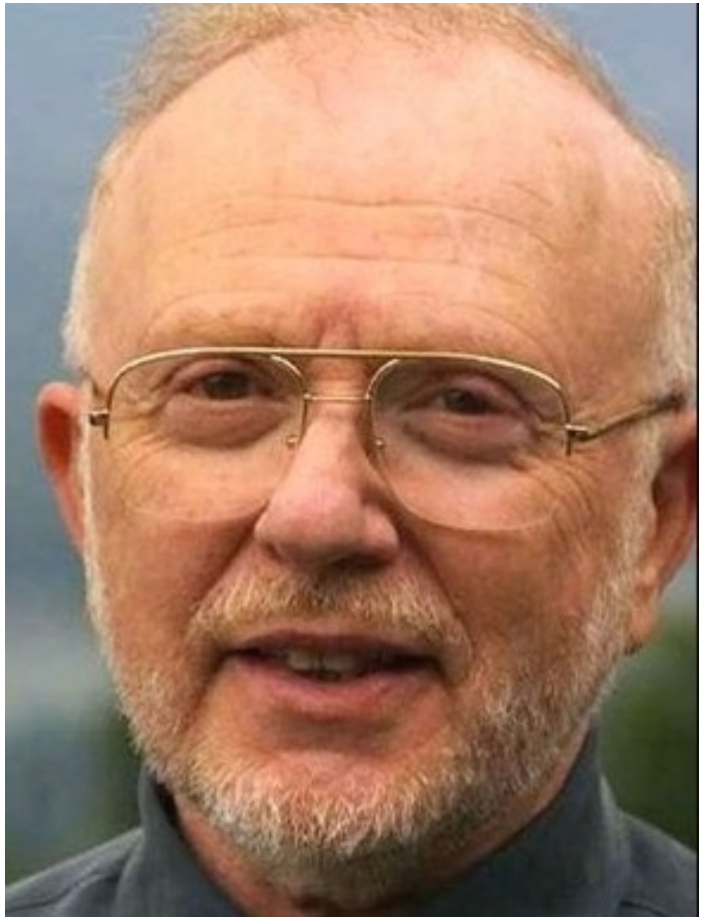
#10yearchallenge

Stephen Cook



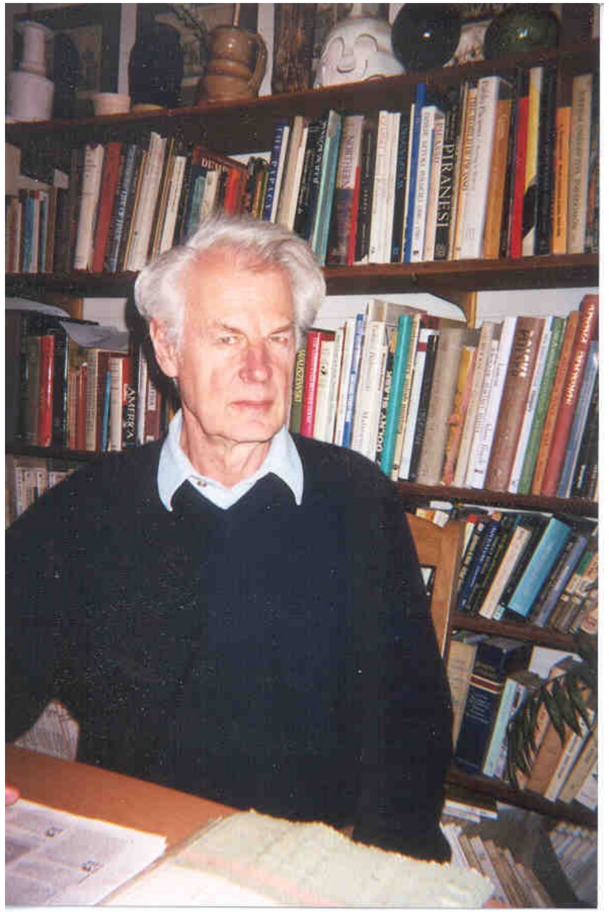
A.M. TURING AWARD

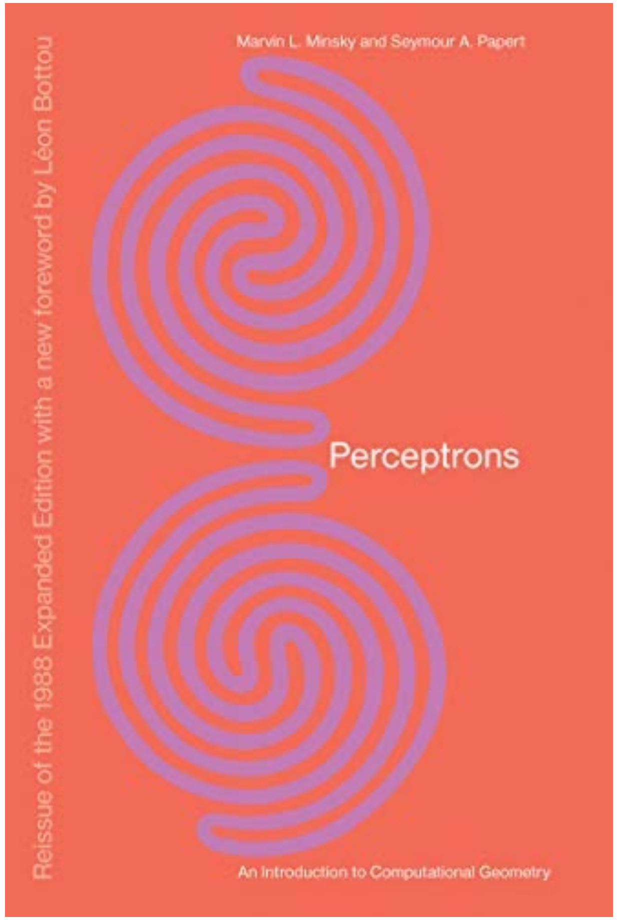
McMaster University

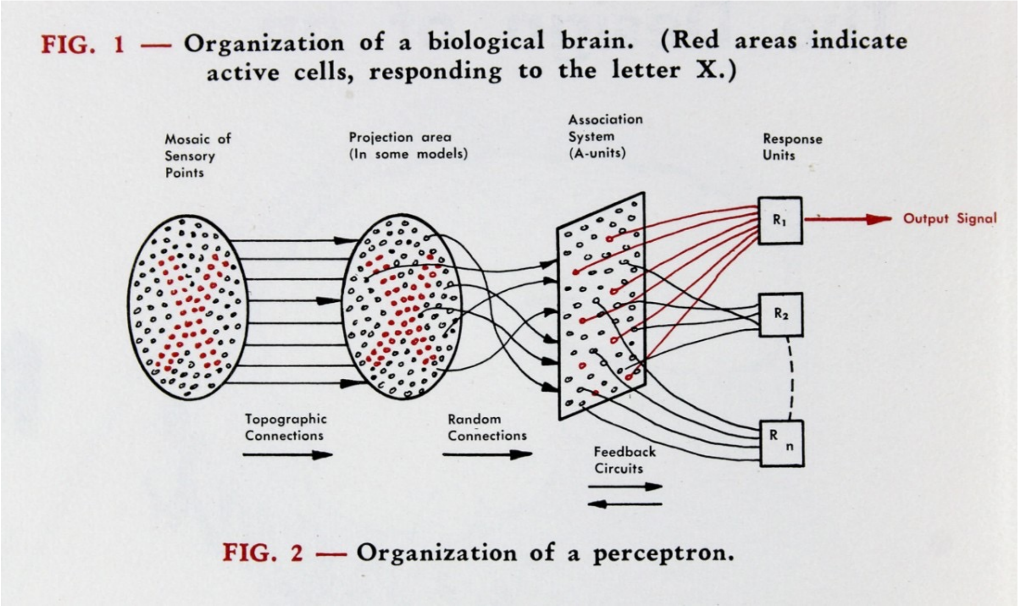


> Copy and paste is a design error.
>
> -David Parnas

Jan Mycielski







Perceptrons: an intro to computational geometry
by Marvin Minsky and Seymour Papert, 1969.
An edition with handwritten corrections released in the early 1970s.

How was ML done

- Code was *bespoke*
  - Written *de novo* each time
- But by the early 2000s:
  - Shared theoretical core of knowledge:
    - Backpropagation
    - Statistical Learning Theory
- What was taught:
  - Theory of neural networks and limitation of learning algorithms
  - How to code them by hand

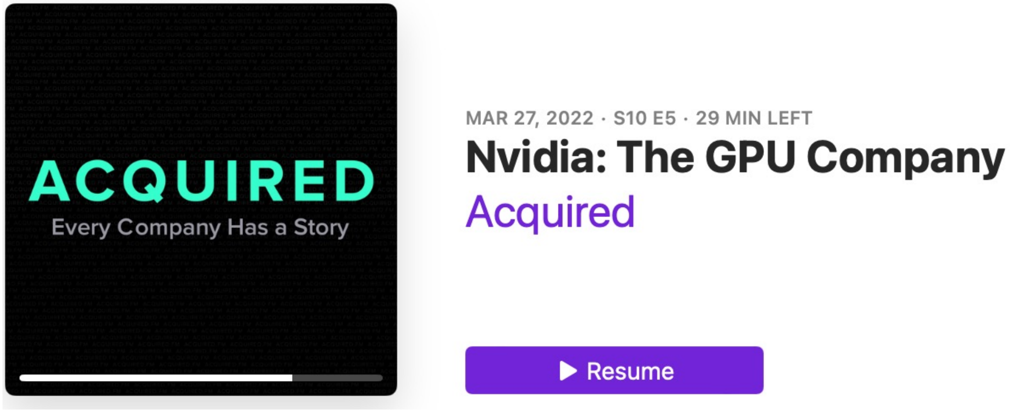
```python
# This code uses for loops to implement backpropagation for a two-layer fully connected sigm
# The network has 2 inputs, 2 hidden units, and 1 output unit

def sigmoid(x):
    return 1.0 / (1.0 + math.exp(-x))

def derivative_sigmoid(x):
    return x * (1.0 - x)

def learn():
    # Inputs
    inputs = [[1, 2], [2, 3], [3, 1], [4, 3], [5, 3], [6, 2]]
    targets = [[0], [0], [0], [1], [1], [1]]

    # Define network
    n_inputs = 2
    n_hidden = 2
    n_outputs = 1

    # Initialize weights
    weights_input_to_hidden = [[0.15, 0.2, 0.25], [0.4, 0.45, 0.5]]
    weights_hidden_to_output = [[0.6, 0.7], [0.65, 0.8], [0.8, 0.9]]

    # Train network
    for i in range(500):
        # Forward pass
        hidden_layer_in = [0, 0]
        for j in range(n_inputs):
            for k in range(n_hidden):
                hidden_layer_in[k] += inputs[j][k] * weights_input_to_hidden[j][k]
        hidden_layer_out = [sigmoid(x) for x in hidden_layer_in]
        output_layer_in = [0, 0]
        for j in range(n_hidden):
            for k in range(n_outputs):
                output_layer_in[k] += hidden_layer_out[j] * weights_hidden_to_output[j][k]
        output_layer_out = [sigmoid(x) for x in output_layer_in]
        # Backward pass
        output_errors = [0, 0]
```

Present

Powerful tools

The Cloud as enabler

- Specs of an AWS SageMaker instance:
  - `ml.g5.48xlarge:`
    - `8 NVIDIA A10G Tensor Core GPUs`
    - `192 vCPUs!`
    - `768GiB storage`



https://podcasts.apple.com/us/podcast/acquired/id1050462261?i=1000558142063

- But Cloud is *not* the solution for everything:
  read [this post](#) on the Stack Overflow architecture

Proliferation of Packages

- 2007 – Theano
- 2010 – Scikit Learn
- 2014 – Jupyter Notebooks
- 2014 – XGBoost
- 2015 – Tensorflow, Keras
- 2016 – PyTorch, MXNet

PyTorch implementation

```python
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.fc1 = nn.Linear(2, 2)
        self.fc2 = nn.Linear(2, 1)

    def forward(self, x):
        x = F.sigmoid(self.fc1(x))
        x = self.fc2(x)
        return x
```

```python
# This code uses for loops to implement backpropagation for a two-layer fully connected sigmoid neural network
# The network has 2 inputs, 2 hidden units, and 1 output unit

def sigmoid(x):
    return 1.0 / (1.0 + math.exp(-x))

def derivative_sigmoid(x):
    return x * (1.0 - x)

def learn():
    # Inputs
    inputs = [[1, 2], [2, 3], [3, 1], [4, 3], [5, 3], [6, 2]]
    targets = [[0], [0], [0], [1], [1], [1]]

    # Define network
    n_inputs = 2
    n_hidden = 2
    n_outputs = 1

    # Initialize weights
    weights_input_to_hidden = [[0.15, 0.2, 0.25], [0.4, 0.45, 0.5]]
    weights_hidden_to_output = [[0.6, 0.7], [0.65, 0.8], [0.8, 0.9]]

    # Train network
    for i in range(500):
        # Forward pass
        hidden_layer_in = [0, 0]
        for j in range(n_inputs):
            for k in range(n_hidden):
                hidden_layer_in[k] += inputs[j][k] * weights_input_to_hidden[j][k]
        hidden_layer_out = [sigmoid(x) for x in hidden_layer_in]
        output_layer_in = [0, 0]
        for j in range(n_hidden):
            for k in range(n_outputs):
                output_layer_in[k] += hidden_layer_out[j] * weights_hidden_to_output[j][k]
        output_layer_out = [sigmoid(x) for x in output_layer_in]
        # Backward pass
        output_errors = [0, 0]
        for j in range(n_outputs):
            error = targets[j][0] - output_layer_out[j]
            output_errors[j] = error * derivative_sigmoid(output_layer_out[j])
            for k in range(n_hidden):
                error = weights_hidden_to_output[j][k] * error
                weights_hidden_to_output[j][k] += hidden_layer_out[k] * error * derivative_sigmoid(hidden_layer_out[k])
        hidden_errors = [0, 0]
        for j in range(n_hidden):
            error = 0
```

Impact

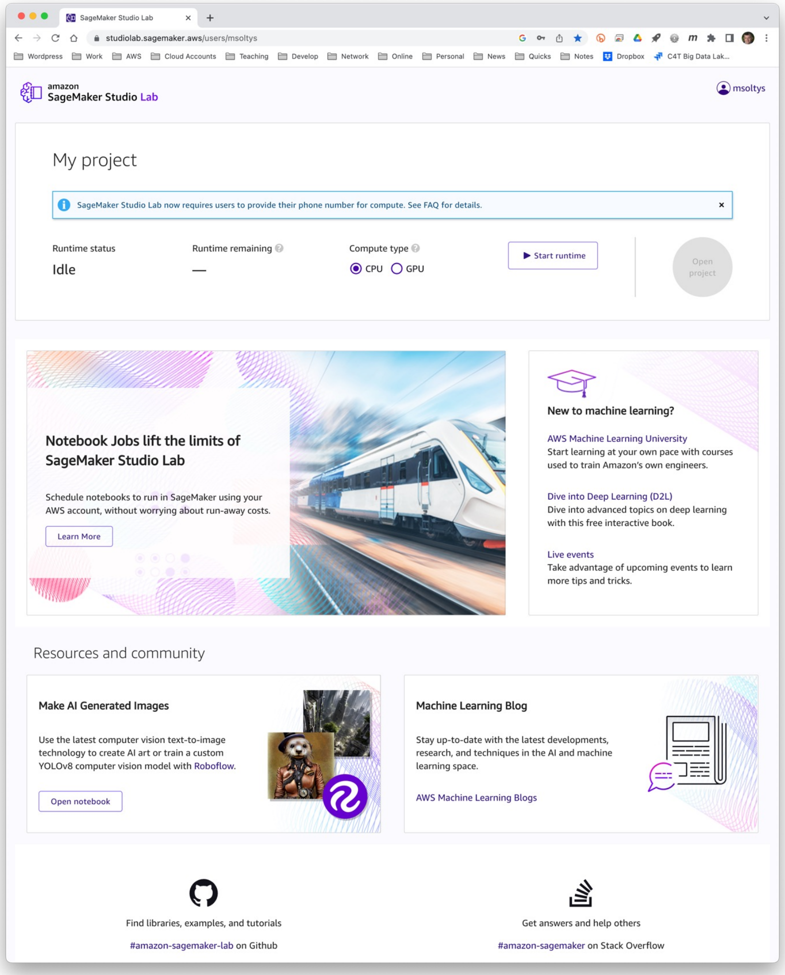
- New tools allowed practitioner to go up one level of abstraction:
  - Before: “How do I take all this math and write it in code?”
  - Now: “How can I structure this network to solve my problem?
  - Or Even: “How do I organize my data/problem so a model can train on it?”

- Entry bar was high (PhD!), but now:
  - Moving ML from research to production with emphasis on tooling
  - Open Source tools like AutoGluon: https://auto.gluon.ai

```
from autogluon.tabular import TabularPredictor

predictor = TabularPredictor(label="label").fit(train_data="train.csv")
predictions = predictor.predict("test.csv")
```

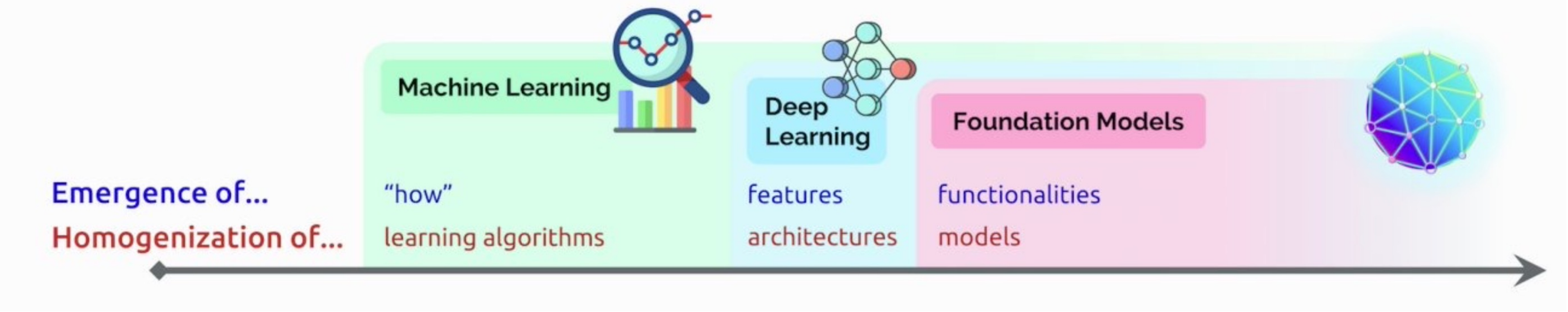
SageMaker Studio Lab

- https://studiolab.sagemaker.aws
- **Free**
- Takes about a week to be approved for account
- Linked to GitHub with lots of examples
- Community on Stack Overflow



Future
Foundation Models

Foundation models

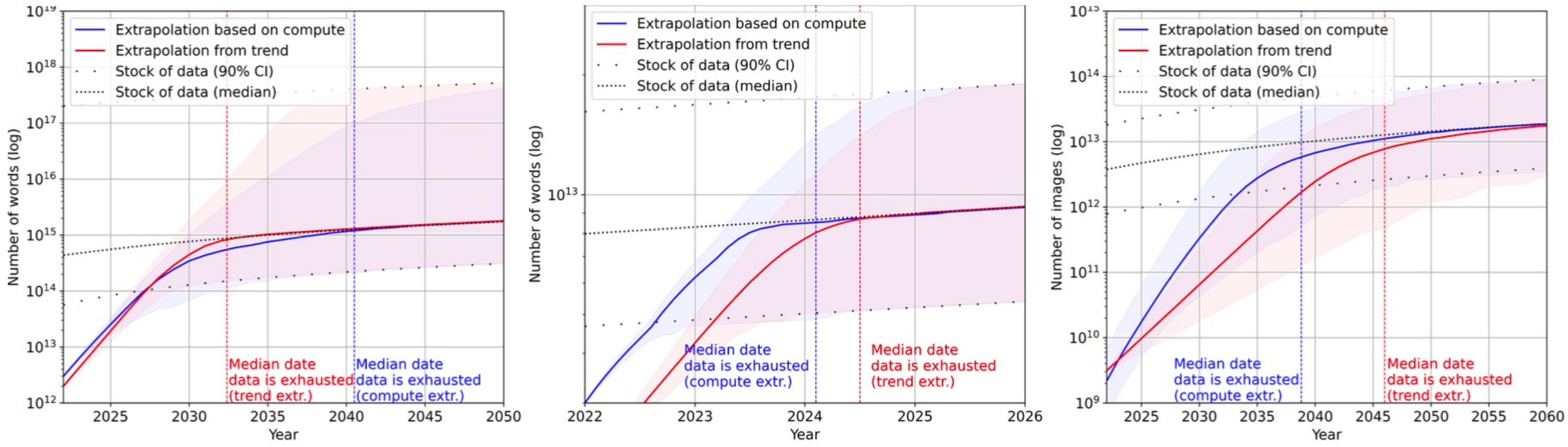


- Increased standardization of models:
  - Code Whisperer
  - GPT-3
  - Stable Diffusion
  - Chat GPT

Characteristics of Foundation Models

- Often trained “self-supervised”
- Predict portions of data from other portions with no explicit labels
  - Eg., fill in blanked out word in text, or fill in missing portion of image
  - Use rich data source (say most text written in history of humanity)
- Expensive, requiring millions $ to train
- Made once, then reused by many without modification of any kind
- Interact by making a sentence where the only way to fill the blank is with answer you want:
  - Eg., “George Washington was barn in the year ___”

Running out of data



(a) Projections for low-quality language data

(b) Projections for high-quality language data

(c) Projections for vision data

Will we run out of data? An analysis of the limits of scaling datasets in Machine Learning

Pablo Villalobos*, Jaime Sevilla*†, Lennart Heim*§, Tamay Besiroglu*‡, Marius Hobbhahn *¶, Anson Ho*

Explain-ability and Ethics

- How to demonstrate (prove) that a model is correct?
  - Why is model training so successful?
- How to demonstrate that a model is not biased?
- How to protect human beings?

Important but not intellectually “elegant”

- CI/CD aspect of ML
  - In industry Git is one of the most important tools
  - Understanding the mathematical foundations is probably the least important
- Documentation has to be superb, and it seldom is
- It doesn’t work for a long time … , until it finally works a little bit
- Interpretation of data – what does 0.3 likelihood of coming to CI mean?
- Communications of methodology and findings – super important! Listen to customer, do not push your fav technology; what is business need?
- Politics of data:
  - No one wants to share their data, even within the same organization; negotiating for data and terms of usage (e.g., access) takes 50% of time of entire effort
  - Hard to reach agreement on ”goodness” of data
  - Even harder to reach agreement on “conclusion” and how to craft policy based on the data

AI SUPER-POWERS

CHINA, SILICON VALLEY, AND THE NEW WORLD ORDER

KAI-FU LEE

Man has made his match … now it's his problem

Skynet is a fictional neural network-based AI system that animates the Terminator