## Machine Learning: Past, Present and Future

Michael Soltys

March 14, 2022 @ CSUCI

Sabbatical Presentation



Image Recognition



**Speech Recognition** 



**Fraud Detection** 



**Self-Driving Cars** 



**Product Recommendation** 



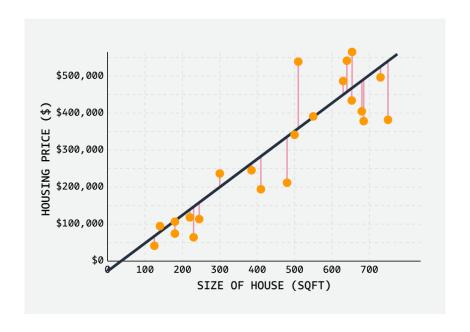
**Traffic Prediction** 

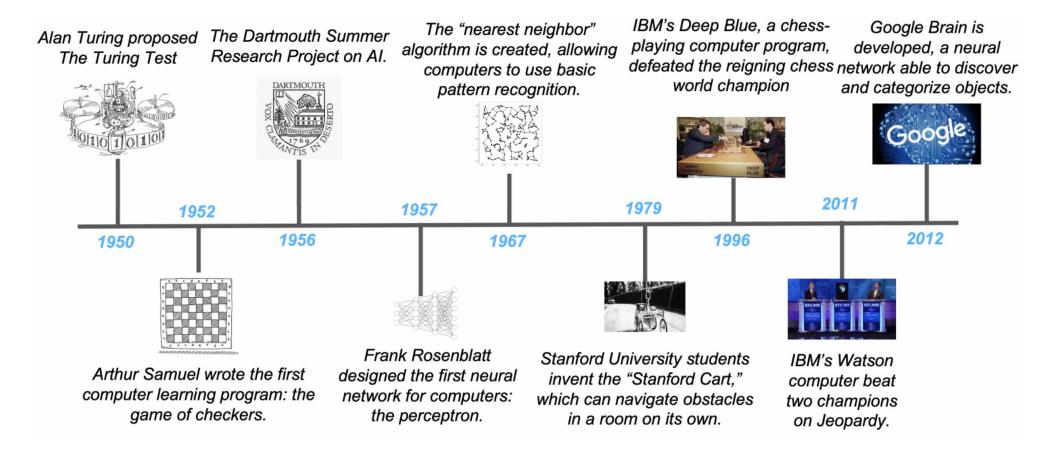
### What is ML?

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

### Example: Linear Regression

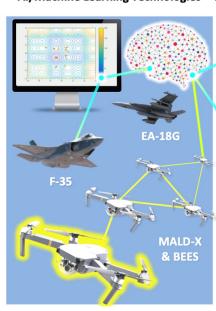
• <a href="https://mlu-explain.github.io/linear-regression/">https://mlu-explain.github.io/linear-regression/</a>

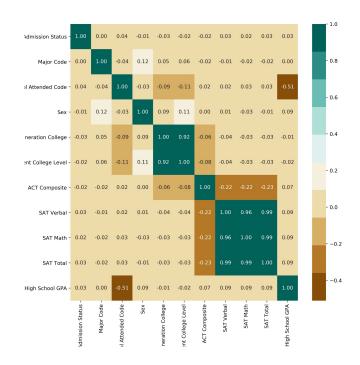




### Example

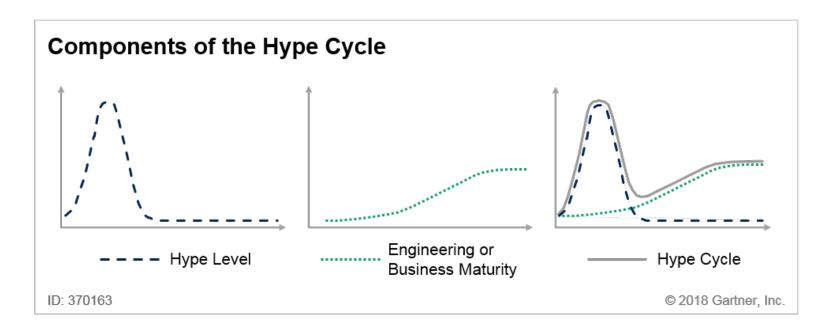




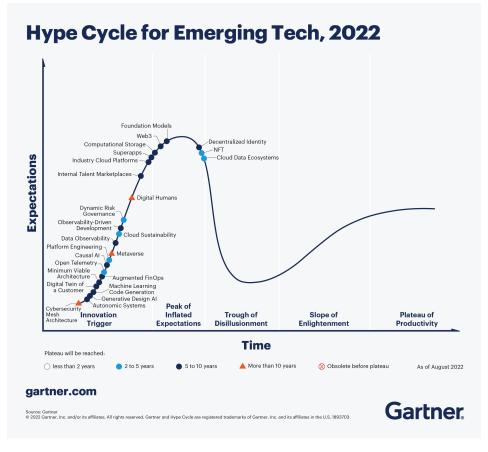


https://github.com/michaelsoltys/sagemaker-enrollment

### 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.

VOL. LIII...NO. 16,774.

NEW YORK, FRIDAY. OCTOBER 9, 1903.-FOURTEEN PAGES.

### Have Arrived at Masam-Pho.

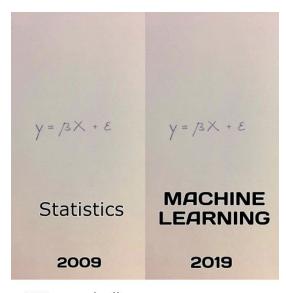
### MAY SUSPEND BALKAN REVOLT. TO EXTRADITE GRAFTERS \$150,000,000 MORTGAGE FILED.

Shipbuilding Trust.

1903 NYT weeks before Wright brothers flew for first time:

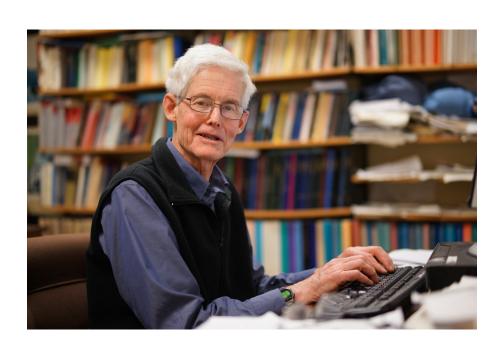
man will not fly for 10 million years

### Mathematics



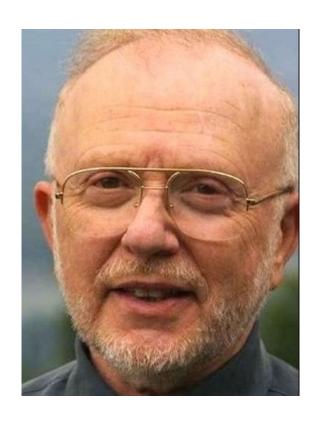
#10yearchallenge

### Stephen Cook





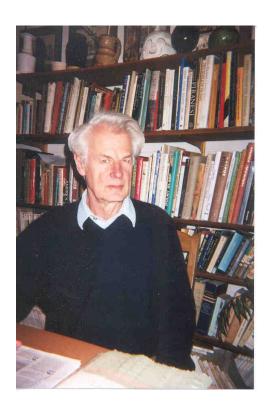
### McMaster University

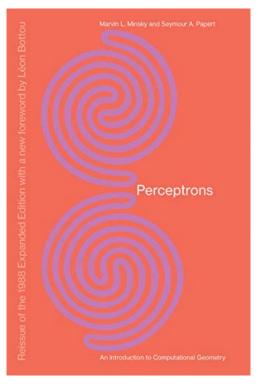


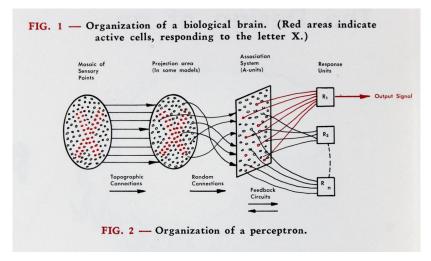
Copy and paste is a design error.
-David Parnas



### 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

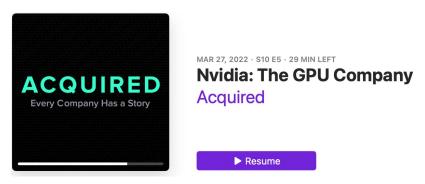
- Code was bespoke
  - Written de novo each time
- But by the early 2000s:
  - Shared theoretical core of knowledge:
    - Backpropagation
    - Statistical Learning Theory
- What was taught:
  - Theory of neural networks and limitation of learning algorithms
  - How to code them by hand

```
return 1.0 / (1.0 + math.exp(-x))
def derivative_sigmoid(x):
   inputs = [[1, 2], [2, 3], [3, 1], [4, 3], [5, 3], [6, 2]]
   targets = [[0], [0], [0], [1], [1], [1]]
   n_hidden = 2
  n_outputs = 1
   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]]
   for i in range(500):
      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]
      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 <u>this post</u> 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

```
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
```

```
return 1.0 / (1.0 + math.exp(-x))
def derivative_sigmoid(x):
    return x * (1.0 - x)
  n_outputs = 1
  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]]
       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]
               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]
           output_errors[j] = error * derivative_sigmoid(output_layer_out[j])
           for k in range(n_hidden):
              error = weights hidden to output[i][k] * error
              weights_hidden_to_output(j)[k] += hidden_layer_out(k) * error * derivative_sigmoid(hidden_layer_out(k))
       hidden_errors = [0, 0]
```

### **Impact**

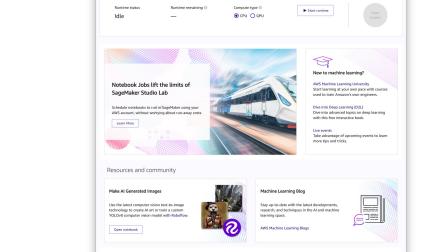
- New tools allowed practitioner to go up one level of abstraction:
  - Before: "How do I take all this math and write it in code?"
  - Now: "How can I structure this network to solve my problem?
  - Or Even: "How do I organize my data/problem so a model can train on it?"
- Entry bar was high (PhD!), but now:
  - Moving ML from research to production with emphasis on tooling
  - Open Source tools like AutoGluon: <a href="https://auto.gluon.ai">https://auto.gluon.ai</a>

```
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



0

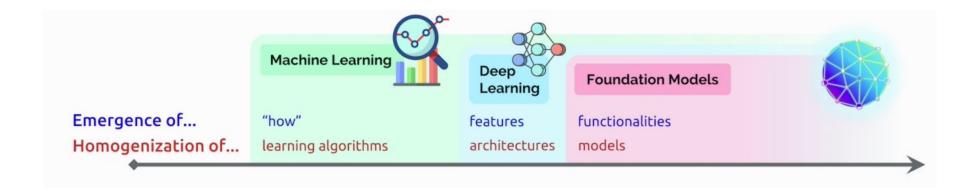
My project

G ~ 0 \* 0 @ 4 0 m \* 0 0

毟

# Future Foundation Models

### Foundation models

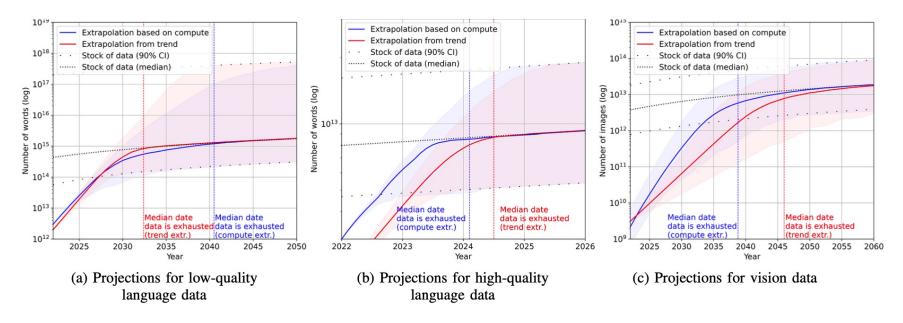


- Increased standardization of models:
  - Code Whisperer
  - GPT-3
  - Stable Diffusion
  - Chat GPT

### 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



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

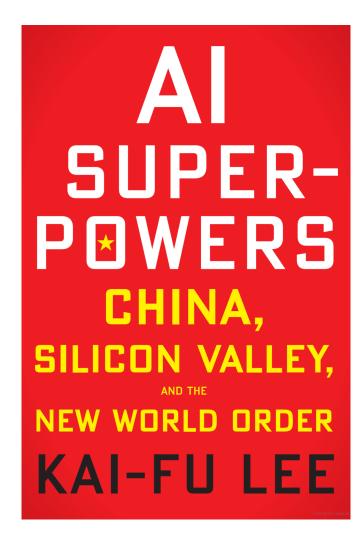
Pablo Villalobos\*, Jaime Sevilla\*†, Lennart Heim\*§, Tamay Besiroglu\*‡, Marius Hobbhahn \*¶, Anson Ho\*

### 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





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



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