DS-UA 301
Advanced Topics in Data Science
Advanced Techniques in ML and Deep Learning

LECTURE 1

Class Introduction

- *Instructor*: Parijat Dube pd2216@nyu.edu
  Senior Research Staff Member at IBM Research, NY
  machine learning/deep learning systems, performance optimization

- *Course Administrator:* Elaine Yang yy2970@nyu.edu

- *Section leader:* Sandeep Menon snm6477@nyu.edu
  Rushabh Musthyala rm6416@nyu.edu
  Elaine Yang yy2970@nyu.edu

- *Graders:* Michael Han zh2033@nyu.edu
  Kaustubh Mishra km5939@nyu.edu
  Savinay Shukla ss16924@nyu.edu
  Wang Xiang wx2008@nyu.edu

- *Prerequisites:*
  - Knowledge of machine learning acquired through introductory coursework
  - Working knowledge of Python and sklearn

What is expected from you?

- Punctuality
- Integrity
- Hard work
- Determination to succeed

Today's Agenda

- Course Overview
  - Course Information and Learning Objectives
  - Syllabus
  - Assignments and Grading
  - Logistics

Course Information

- *What this course will cover ?*
  - Basics of Deep Learning (DL) and DL architectures
  - DL training, frameworks, hyperparameters
  - Tools and techniques for performance optimization of DL
  - Automated ML, adversarial training, distributed training
  - Programming assignments involving training and optimizing DL models
  - Paper readings
- *What this course will not cover ?*
  - Details about specific ML algorithms
  - Mathematical analysis of ML/DL algorithms

Learning Objectives

- Identify different components of DL system stack and their interdependencies
- Knowledge of ML model lifecycle and steps in making a trained model production ready
- Ability to train and optimize DL models using GPUs
- Performance considerations, tools, techniques at different stages of model lifecycle: development, testing, deployment.
- Optimizing models through model selection, feature design, and hyperparameter tuning.

Course Contents: Module 1

AI revolution, ML concepts, Introduction to ML tooling

- AI revolution and success factors
- ML viewpoints: algorithmic vs system
- ML system and its constituents
- ML performance metrics: algorithmic and system Level
- ML performance concepts/techniques: overfitting, generalization, bias, variance, regularization
- ML lifecycle and different stages
- Importance of tooling in ML

Course Contents: Module 2

Introduction to Deep Learning (DL)

- Artificial neural network
- Single layer neural network
- Activation functions
- Loss functions
- Multi-layer neural network
- Neural network training: gradient descent, backpropagation, data preprocessing, SGD, optimizers
- DL hyperparameters: learning rate, batch size, momentum
- Learning rate schedules
- Regularization techniques in DL Training: dropout, early stopping, data augmentation
- Contrastive loss

Course Contents: Module 3

DL Training Tools and Techniques

- DL datasets: MNIST, FashionMNIST, CIFAR10/100, ImageNet
- Introduction to DL frameworks: PyTorch
- Designing and training neural networks in PyTorch
- Training-logs and their analysis
- Checkpointing: framework specific support, restarting from checkpoint
- ML platforms on cloud: AWS, Microsoft, Google, and IBM
- DL training on cloud platforms
- Learning with limited labels

Course Contents: Module 4

Specialized DL Architectures

- Convolutional neural networks (CNNs)
- Recurrent neural networks (RNNs)
- Long short-term memory networks (LSTMs)
- Word embeddings
- Attention networks and transformers
- Generative adversarial networks (GANs)
- Siamese neural networks
- Large Language Models (LLMs)

Course Contents: Module 5

Hyperparameter Optimization and Feature Engineering

- Challenges in hyperparameter optimization
- Hyperparameter optimization techniques
- Hyperband algorithm for hyperparameter optimization
- Tooling for hyperparameter optimization
- Challenges in feature engineering
- Tooling for feature engineering

Course Contents: Module 6

Automated Machine Learning

- Machine learning pipeline and its automation
- Automated ML tools: auto-sklearn, auto-weka, Tpopt, Hyperopt-sklearn
- Neural Network Intelligence (NNI) and H20 AutoML
- Overview of AutoML platforms: IBM AutoAI, Google AutoML
- Open Neural Network Exchange (ONNX)

Course Contents: Module 7

Robust Machine Learning

- Model brittleness
- Adversarial attacks
- Adversarial defenses
- Adversarial training
- Analysis using Adversarial Robustness Toolbox

Course Contents: Module 8

Distributed Training and Federated Learning

- Parallelism in DL training: model parallelism, data parallelism
- Synchronous and asynchronous SGD, straggler problem, stale gradients, variants of synchronous and asynchronous SGD
- Pytorch data parallelism support: DataParallel, DistributedDataParallel, DistributedDataSampler
- Federated learning, FedAvg; Adversarial training in federated setting.

Course Contents: Module 9

Model drift and Continual learning

- Model performance drift
- Data drift and concept drift
- Monitoring and drift detection
- Model retraining
- Tools for continual learning

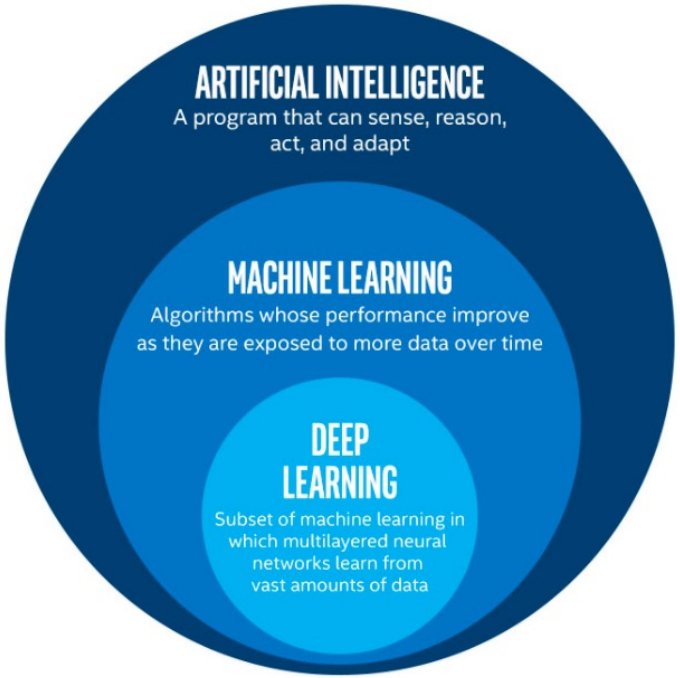
Course Overview

- Course home: https://brightspace.nyu.edu/d2l/home/314740
- Syllabus: https://brightspace.nyu.edu/d2l/le/lessons/314740/lessons/9100287
- Grade distribution
  - Assignments (30%)
  - Exam (30%)
  - Project (30%)
  - Quizzes (10%)
  - Class Participation (±5%)
  - Online Discussion: (+2%)
- Quizzes: 5 in total. Start from week 2. Schedule on Brightspace.
- Assignments: Quantitative questions to be answered with data and code, you will submit both your answers and your source code
- Project: Done in a team of 3 students. You need to submit a project proposal with the dataset you will work with by midterm. Final project presentations will happen at the end of the semester
- Attendance policy
  - **Lectures:** Not mandatory. Upto 5% extra credit.
  - **Labs:** Mandatory. Upto 5% penalty.

Recommended Text

- Aggarwal, Charu C. (2023). Neural Networks and Deep Learning: A Textbook, 2nd edition, Springer.
- Raschka, S.and Mirjalili, V. (2019). Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and Tensorflow 2, 3d edition. Packt Publishing.
  - Code examples Jupyter notebooks available for free download at https://github.com/rasbt/python-machine-learning-book-3rd-edition
- Geron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. 2nd Edition. O'REILLY.
- Goodfellow, Bengio, Courville, "Deep Learning", available at http://www.deeplearningbook.org
- "Dive into Deep Learning" (online) available at https://d2l.ai
- Reading materials will be shared as course progresses

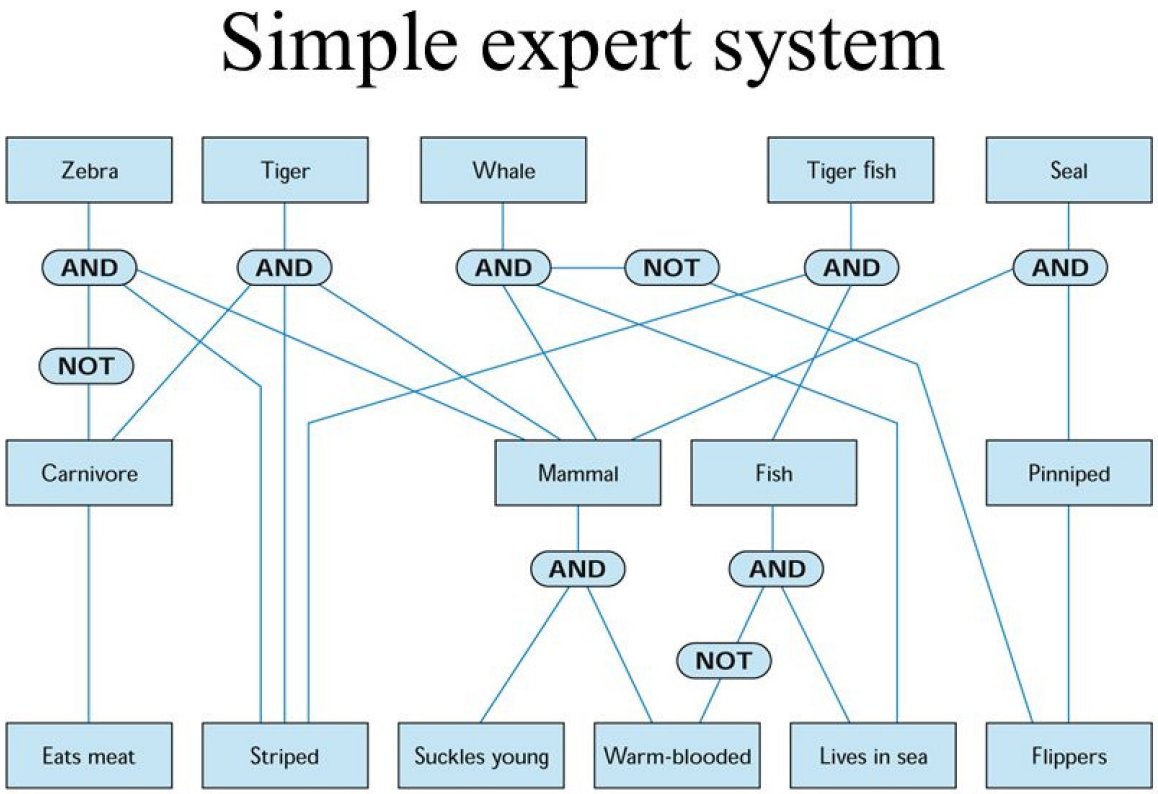
AI, ML, DL



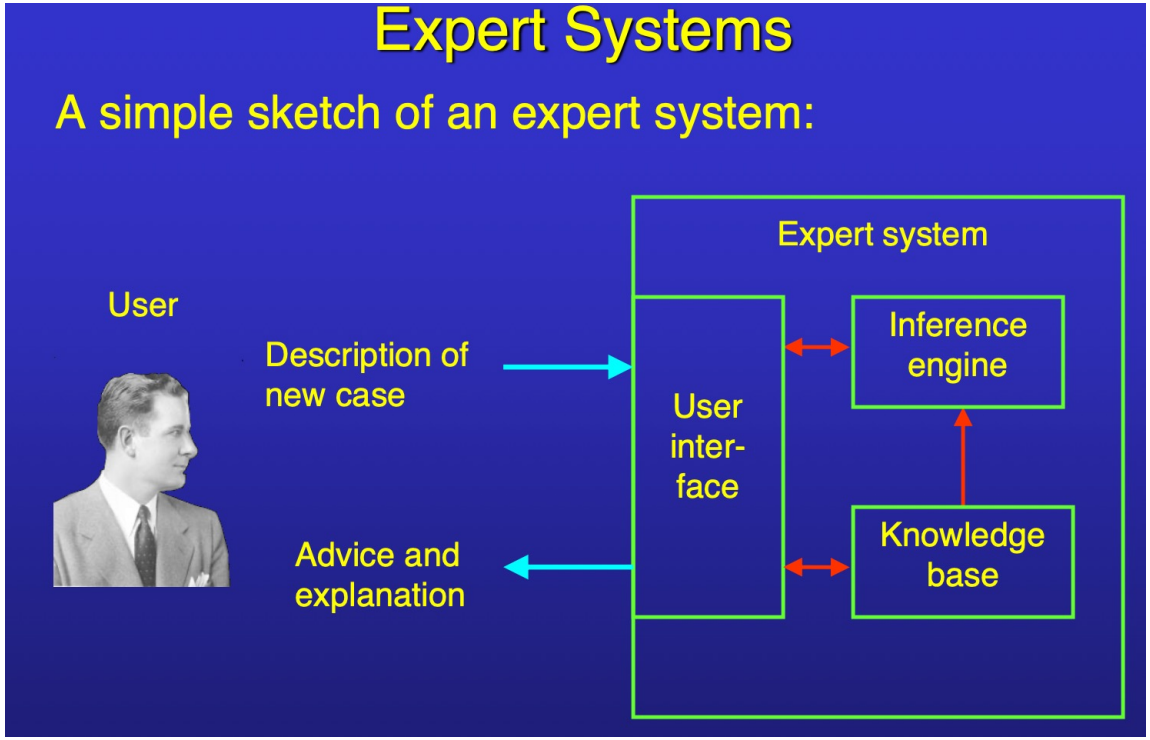
Source: Intel. "How to Get Started as a Developer in AI." October 2016

- AI is dubbed as the Fourth wave of Industrial Revolution
- DL is a subset of ML and involves use of Deep Neural Networks (DNNs)

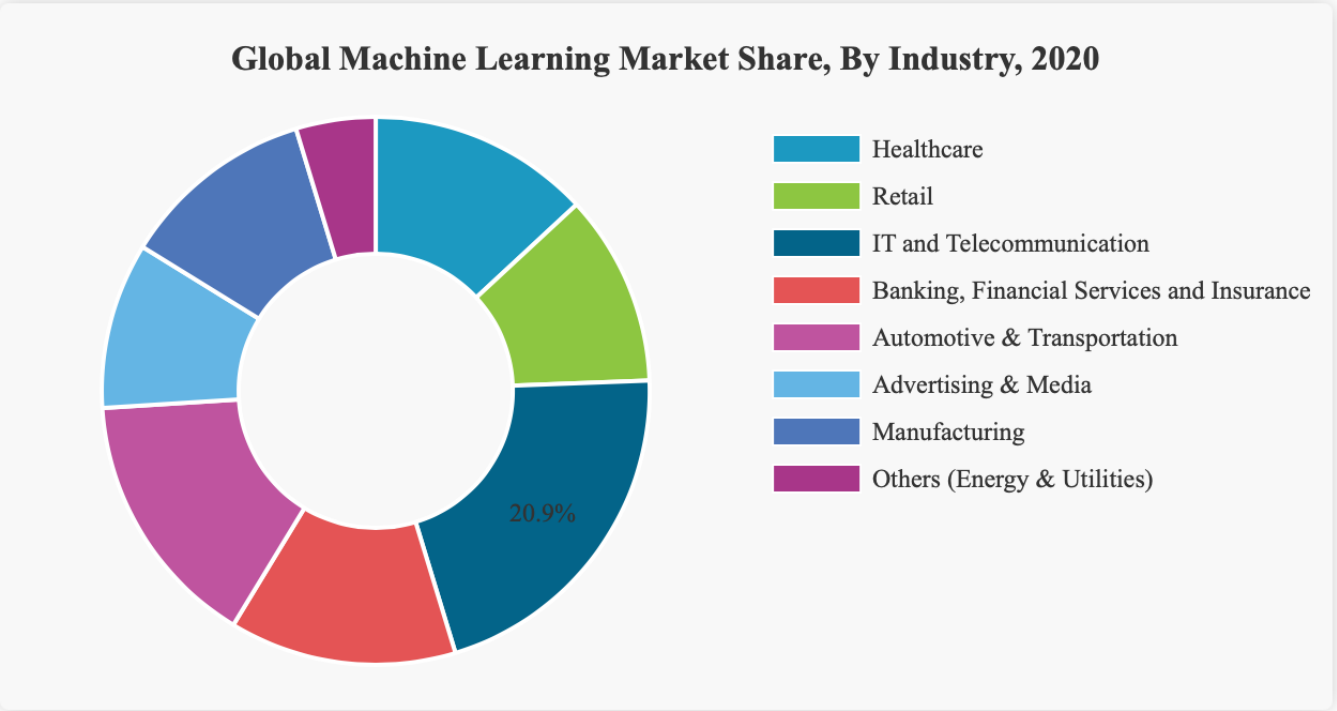
Non-ML based AI



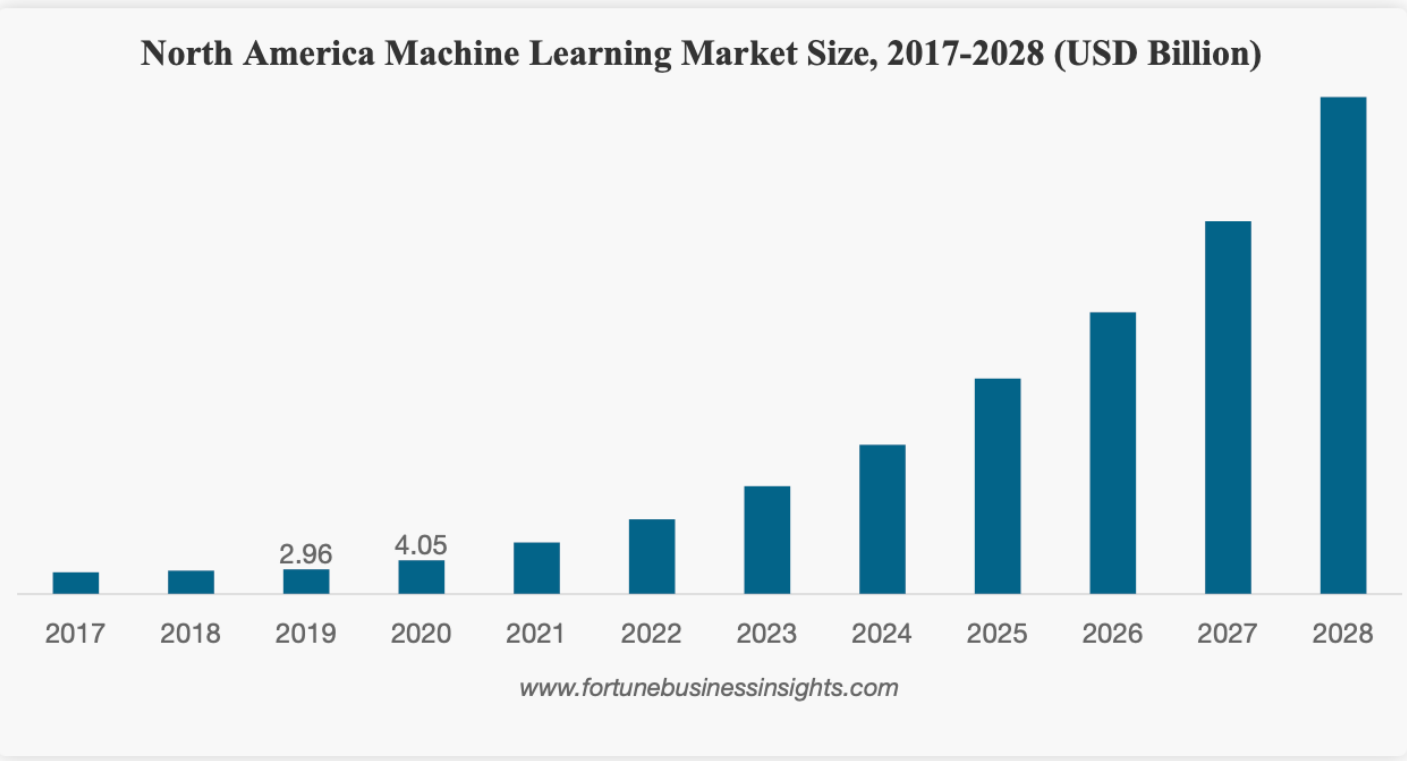
Knowledge base and Inference engine needs to be programmed



Machine Learning Market Share, By Industry



Machine Learning Popularity



Inflection point in AI adoption are closely tied to innovations in computing and availability of big data

AI Timeline

Cloud computing was coined

2006: Amazon elastic cloud and S3 was launched

Factors Contributing to AI Success

- **Data, Compute, Algorithms, Applications**
  - Data growing at exponential rate; Internet, Social media, IoT
  - Compute power growth with specialized hardware
  - Innovations in scaling of machine learning training algorithms
  - Development of innovative applications based on processing of big data
  - Triggered ”Cambrian Explosion” in machine learning technologies

  “*Neural networks are growing and evolving at an extraordinary rate, at a lightening rate,….What started out just five years ago with AlexNet…five years later, thousands of species of AI have emerged*.”

Examples of Deep Learning

- Visual Recognition
- Handwriting Recognition
- Self Driving Cars: Intel Mobileye, Waymo
- Conversational agents: IBM Watson Assistant
- Transportation: Uber AI
- Art: Generation of artworks

AI/ML Blogs of Major Companies

- Facebook AI: https://ai.facebook.com/blog/
- Google AI Blog: https://ai.googleblog.com
- IBM Research Blog AI:

  https://www.ibm.com/blogs/research/category/ai/
- Microsoft The AI Blog: https://blogs.microsoft.com/ai/
- AWS Machine Learning Blog: https://aws.amazon.com/blogs/machine-learning/

Real World Examples of Machine Learning

- Machine Learning: 6 Real-World Examples
- Top 10 applications of Machine Learning

AI for Agriculture

- Facebook AI Blog: PyTorch drives next-gen intelligent farming machines
- Chris Padwick medium article: AI for AG: Production machine learning for agriculture

Search for Exoplanets with Machine Learning

- Earth to exoplanet: Hunting for planets with machine learning
- Data from NASA's Kepler Mission: 200,000 stars over 4 years; image take every 30 seconds
- Roughly 14 billion data points
- Selected **15,000 labeled data points** across 677 stars to train a **Tensorflow model**
- **Test accuracy: 96%**
- Discovered 2 new planets: Kepler-90i and Kepler-80g
- Details at Google AI blog: Opensourcing the hunt for planets

AI for Healthcare

- Read article: Opinion article: Artificial Intelligence for Healthcare in Africa
- Stanford news: Deep learning algorithm does as well as dermatologists in identifying skin cancer
- Read article: The Real-World Benefits of Machine Learning in Healthcare
- Video: https://youtu.be/i8CQ8Ud0HN0

DeepMind's AlphaGo Zero

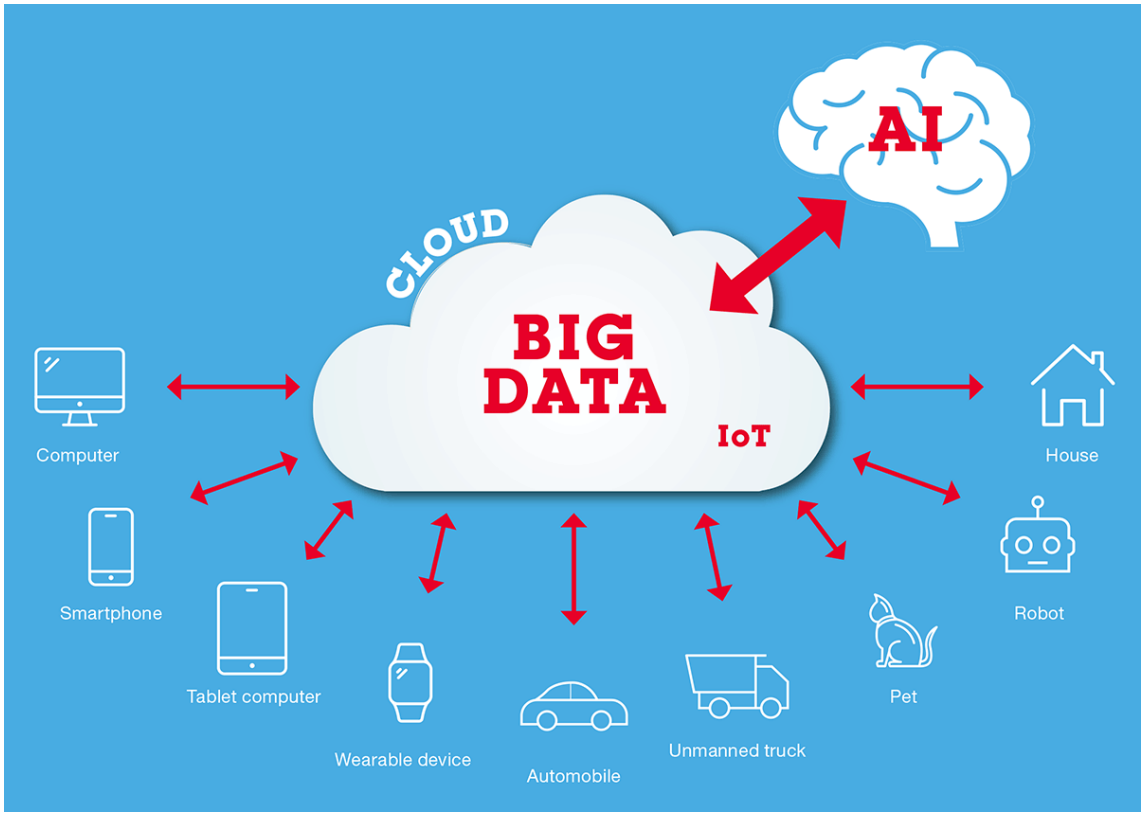
- AlphaGo Zero: Starting from Scratch
- 2017 NIPS Keynote by David Silver
- Nature article: Mastering the game of Go without Human Knowledge
- David Foster medium article: AlphaGo Zero Explained in one Diagram
- AlphaGo Zero Cheat Sheet

AI in Public Policy

- Many challenges associated with Bias, Fairness, and Explainability of machine learning based solutions.
- Machine Learning for Policy Makers: What is it and Why it Matters. (Links to an external site.)
- When governments turn to AI: Algorithms, trade-offs, and trust (Links to an external site.)
- The tensions between explainable AI and good public policy

Cloud and AI

- AI
  - Harness power of Big Data and compute
- Cloud
  - Access to Big Data
  - Platform to quickly develop, deploy, and test AI solutions
  - Ease in AI reachability
- Cloud + AI is the winning combination



Inhibitors in Successful Implementation of ML Solutions

- Deployment and automation
- Reproducibility of models and predictions
- Diagnostics
- Governance and regulatory compliance
- Scalability
- Collaboration
- Monitoring and management

Think!

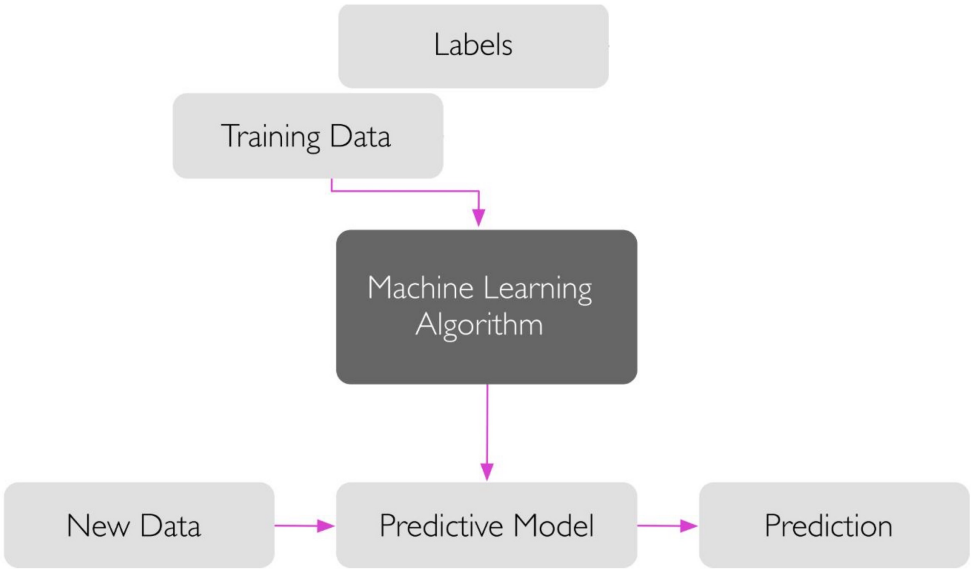
- What are the driving forces behind the wide applicability of machine learning in today's world?
- What are 4 challenges in building machine learning solutions?

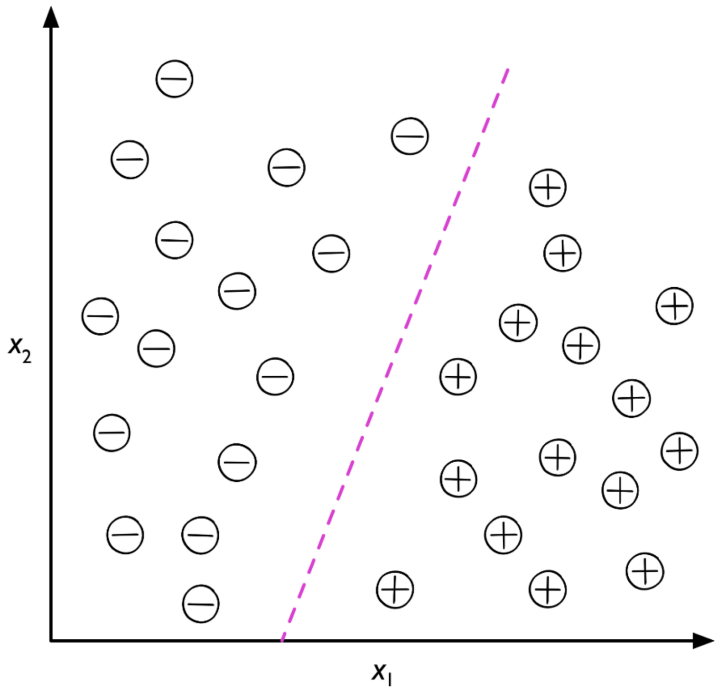
ML Model Training and Inferencing

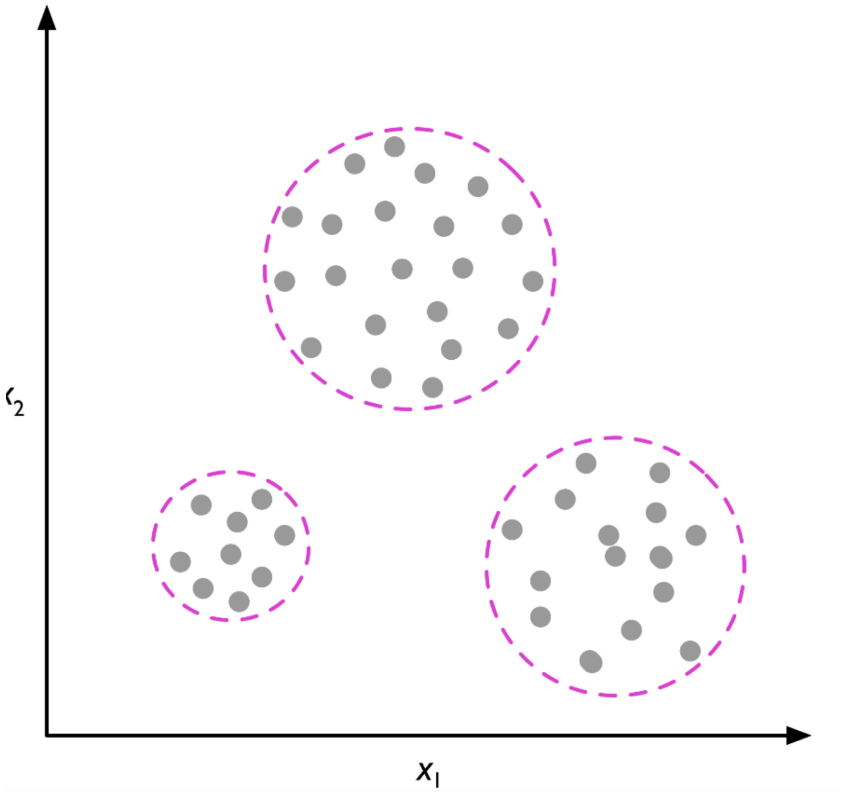
- What is ML model ?
  - A computer program trained to learn from data and perform tasks requiring human intelligence
- What is ML model training ?
  - Process of using data to learn the ML model
- What is inferencing ?
  - Process of using the trained model to make predictions on test data
- Why is data needed and important ?
  - ML is data-driven machine intelligence; ML model knowledge is all learned from the data
- Type of data for ML training ➔ Type of training (Supervised vs Unsupervised)

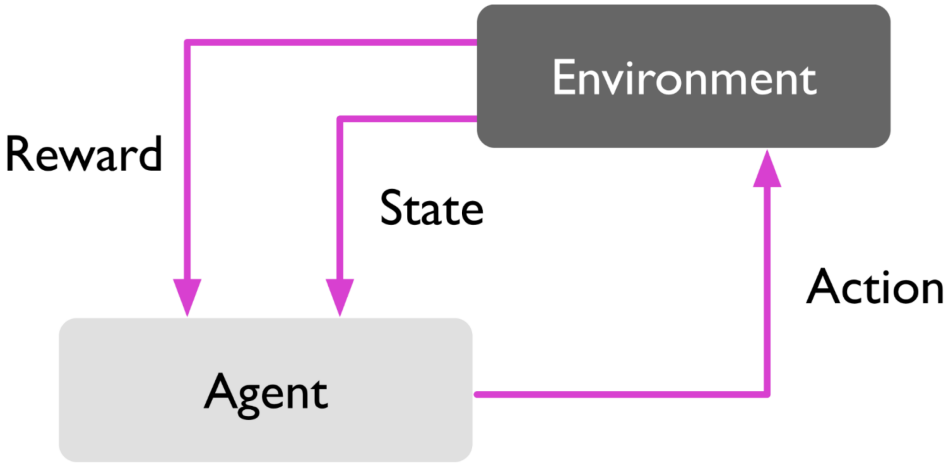
Types of Learning Algorithms

- **Supervised**: learning using labeled training dataset
  - Goal is to infer a function mapping input to the output using labeled input-output pairs
    - Inferred function correctly predicts output labels for other inputs
  - Example Algorithms
    - Linear and Logistic regression
    - Support Vector Machine
    - Backpropagation based neural network
    - Tree based: Decision Trees, Random Forest
- **Unsupervised**: learning using un-labeled training dataset
  - Goal is to infer hidden structure in the input dataset that can help in your understanding of the data
    - Infer clusters/patterns in data based on input features
  - Examples: Clustering, Autoencoder
- **Reinforcement**: learning by sensing, acting, and evaluating reward
  - Goal is to learn the best set of actions which maximize long-term reward
  - Examples: self-driving car, video-games
- Also **Semi-supervised**


Labels
Training Data
Machine Learning Algorithm
New Data
Predictive Model
Prediction


x_2
x_1


x_2
x_1


Environment
Reward
State
Action
Agent

Creating a real-world ML Solution

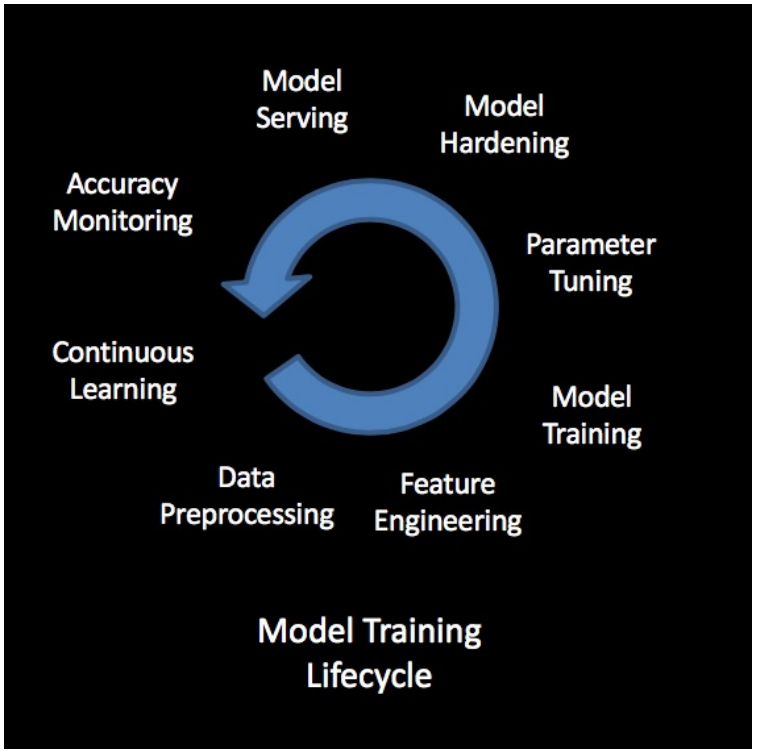
- **System Objective:** What problem will my system solve ? What is the target deployment scenario? What are the performance objectives ?
- **Approach**: What is my solution and its components ?
- **Data Collection**
  - Identifying the data sources: type, schema
  - Collecting the data
- **Data Preparation**
  - Preparing the data: Is my data business ready ?
  - Ingesting the data: What is the right storage for my data ?
- **Model Development**
  - Data preprocessing
  - Identification and training
  - Model evaluation
  - Hyperparameter Tuning
- **Model Deployment**
  - Model optimization (if needed) for the deployment infrastructure
  - Model packaging and deployment
- **Monitoring and Feedback**
  - Is my deployed model performing as expected ?
  - Is there a drift in model performance which requires re-training ?

Operationalization of Machine Learning

Develop ML model → Deploy ML model → Generate business value

- Operationalizing a machine learning model involves deploying the model in production environments where it makes prediction on real-world data
- Challenges:
  - Model performance is data dependent; mismatch between training data distribution and production data distribution can cause model performance to deteriorate in production environments
  - Continuous monitoring of model performance, model adjustments, collecting new data, re-training the model are needed to make sure model performance is maintained
  - Model size needs to be adapted to target environment where it is deployed, e.g., edge devices, mobile phones, self-driving cars, cloud, desktop. Goal is to make the model fit the resources available at target environments. This might require sacrificing model accuracy to improve efficiency.
- Read this article: Key considerations for operationalizing machine learning

ML Model Lifecycle in Production Environment

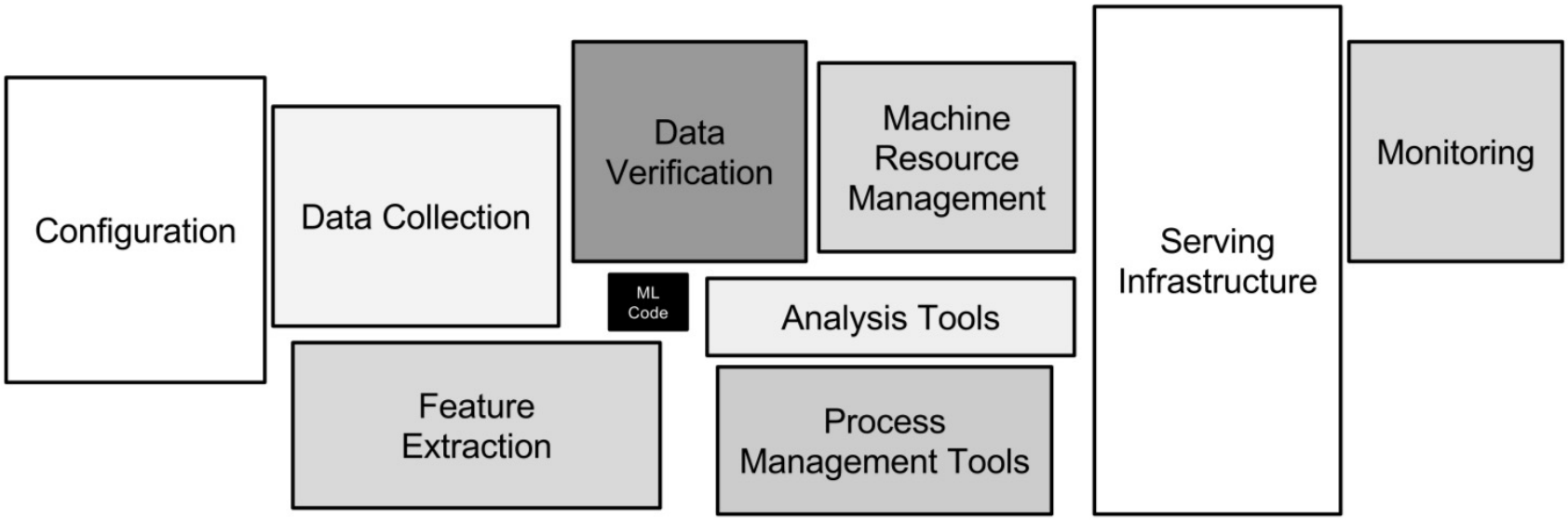


- Data preprocessing: de-noising, de-biasing, train/test set creation
- Feature engineering: search efficient data transformations
- Model training: model identification/synthesis, hyperparameter tuning, regularization
- Model hardening: making model robust against malicious attacks
- Model serving: hardware, model pruning (making the model smaller/simpler without losing performance) and compression
- Monitoring: response time, detect degradation in model performance
- Continuous learning: model adaptability, retraining

Think!

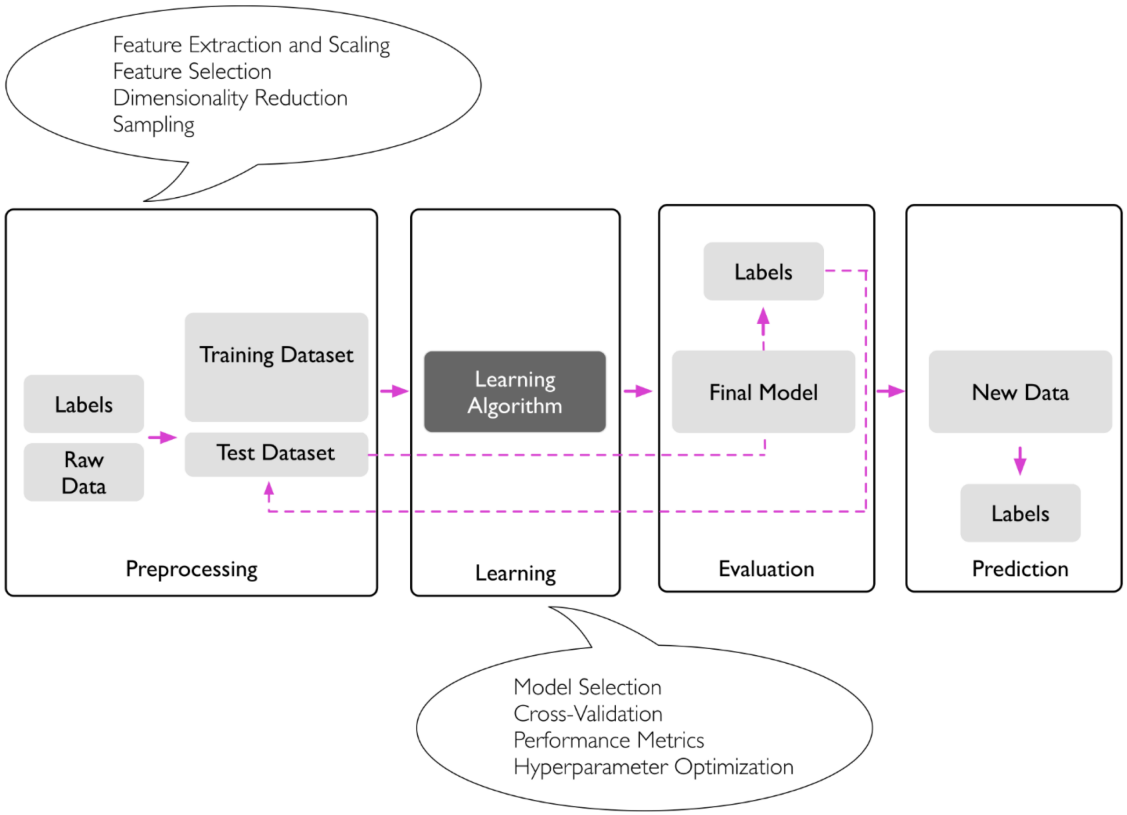
- What are the different stages in model lifecycle?
- What does it mean to operationalize machine learning?
- List 2 challenges in operationalizing machine learning?

Practical Machine Learning Systems



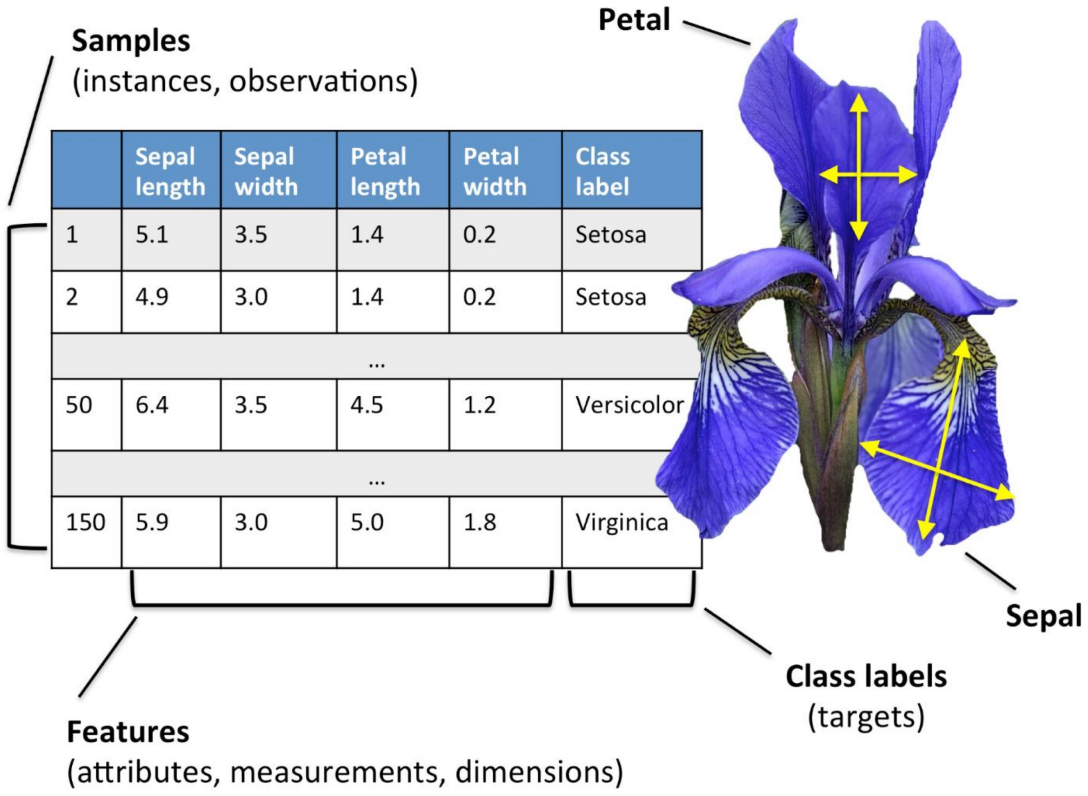
The portion of ML code in a real-world ML system is a lot smaller than the infrastructure needed for its support.

Machine Learning Vanilla Pipeline



Machine Learning Terminology

- Training data
- Feature
- Target
- Loss/cost function



	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
			...		
50	6.4	3.5	4.5	1.2	Versicolor
			...		
150	5.9	3.0	5.0	1.8	Virginica

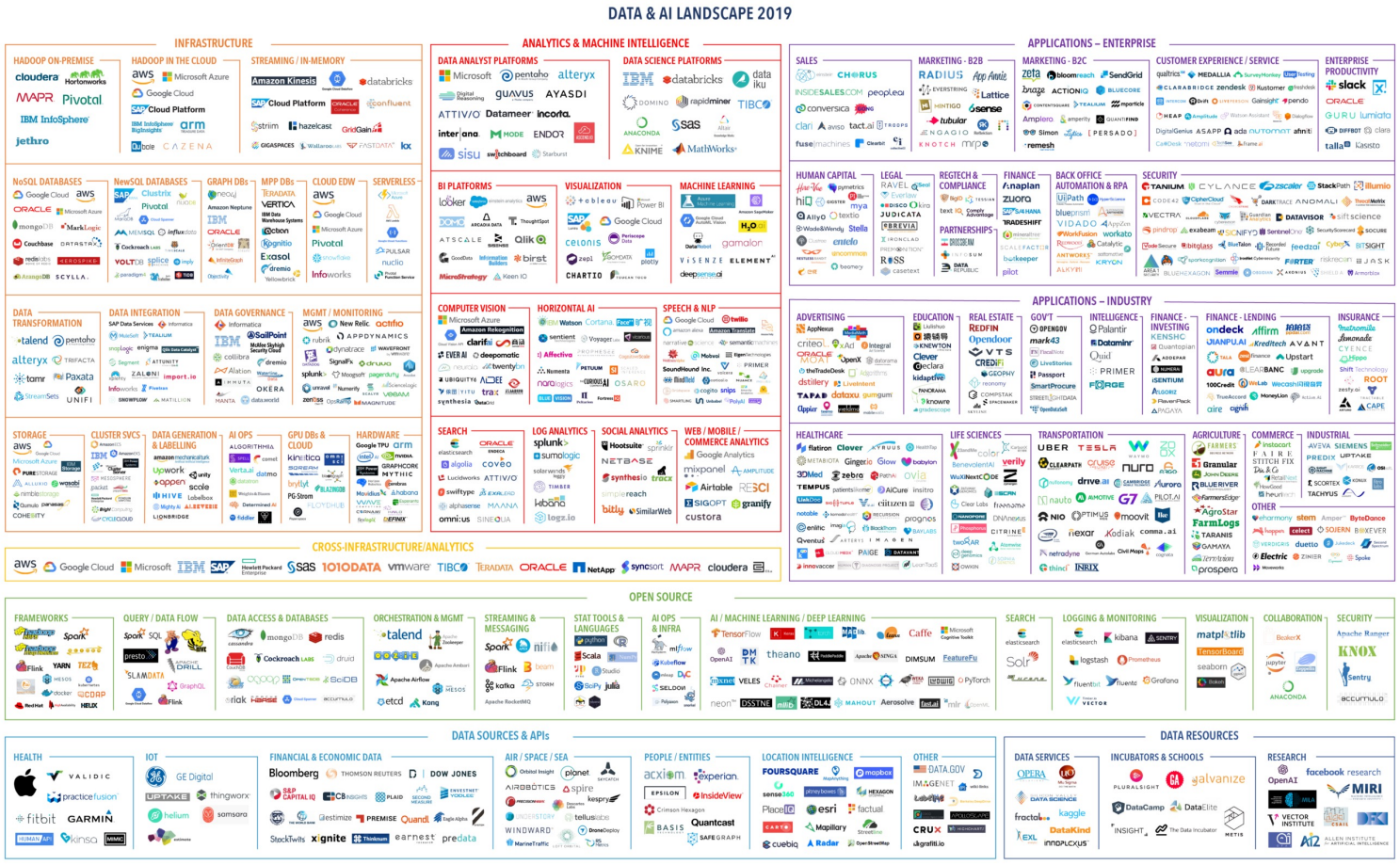
ML Tools

- Why we need ML tools?
- Tools help to design, develop, and deploy **full-stack Machine Learning** application
- Features of a good tool:
  - Intuitive interface
  - Best practice
  - Trusted resource
- Platforms, libraries

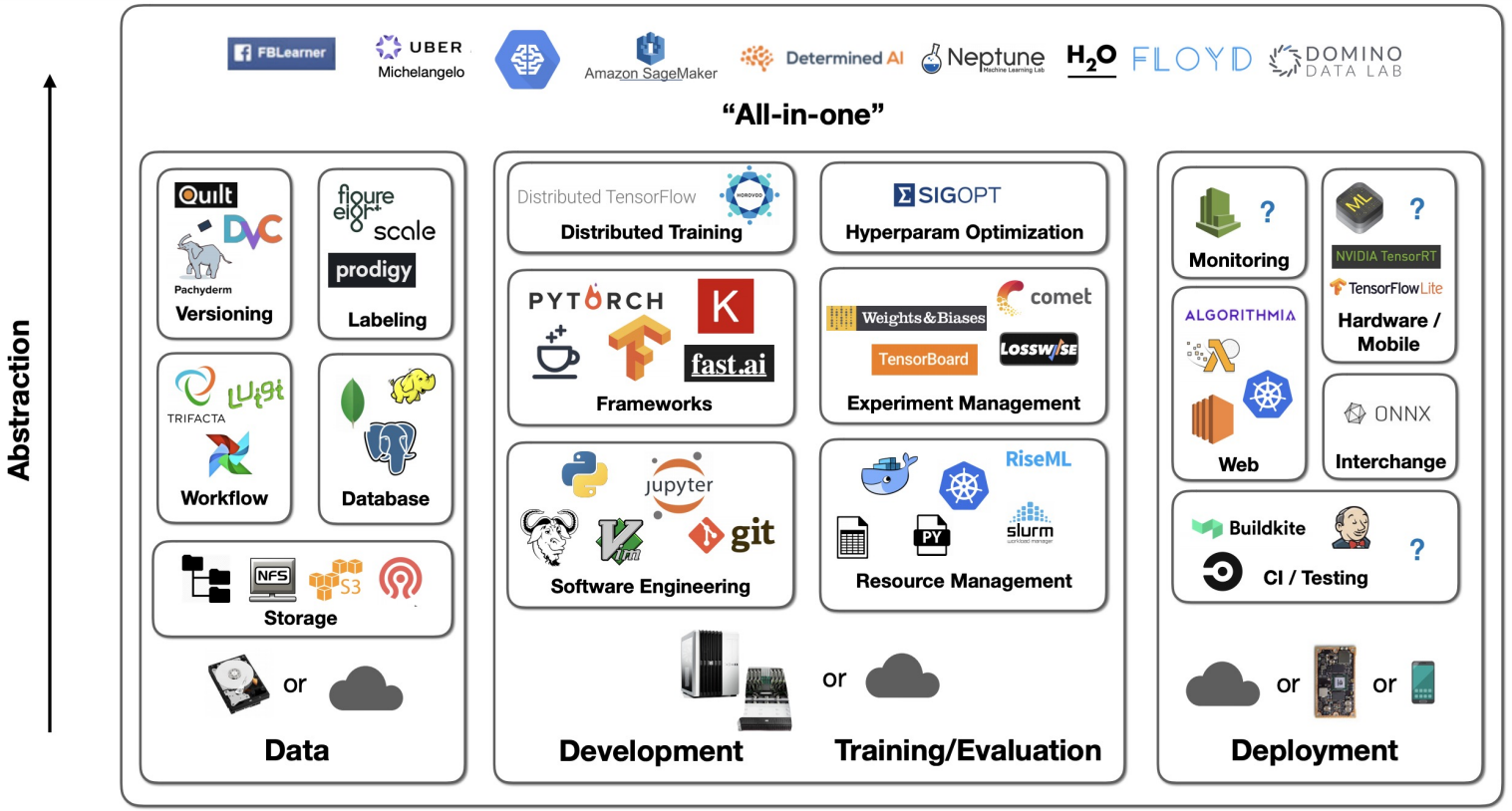
Machine Learning with Python

- Specialized packages/library support for efficient numerical, scientific, and data science computations
  - NumPy: library with support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays
  - Pandas: library providing data structures and tools for efficient data analysis built on NumPy
  - SciPy: library used for scientific computing built on NumPy
  - Matplotlib: visualization library for static, dynamic, interactive plotting
  - Scikit-learn: machine learning library built on NumPy, SciPy, and Matplotlib
- Deep Learning frameworks: Tensorflow, Keras, Pytorch

ML Tool Landscape: Overwhelming!



Tools for different stages



Democratization of AI

- Harnessing of AI potentials should be wide-spread and easy
- Not only restricted to big companies having resources (capital and AI expertise)
- https://www.h2o.ai/democratizing-ai/

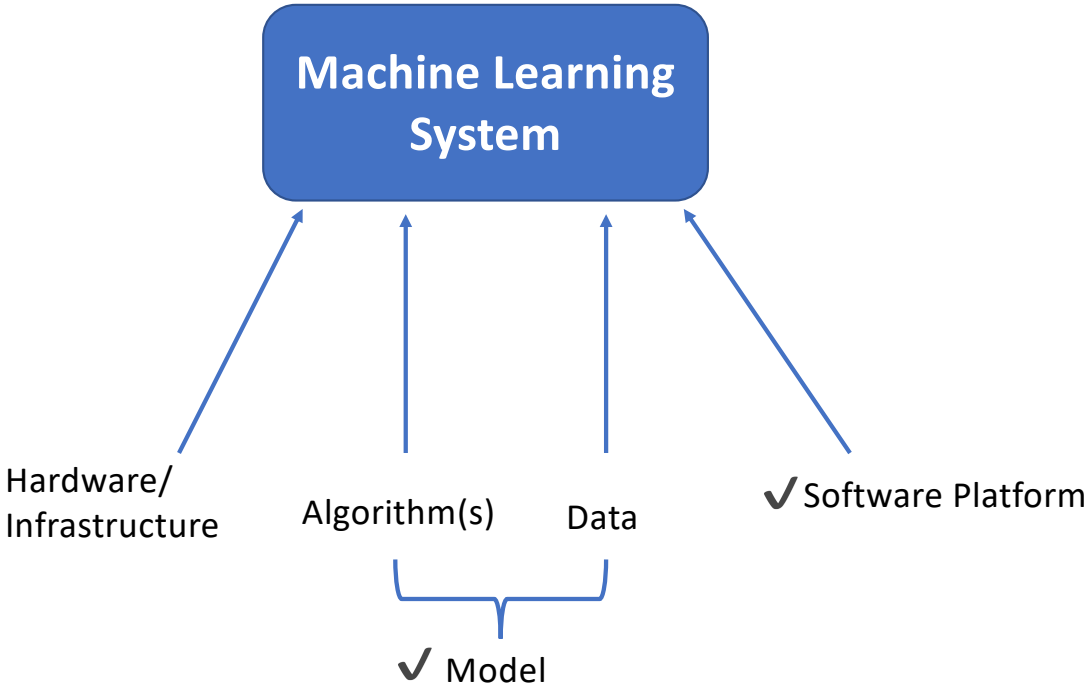
Factors Contributing to Democratization of AI

- Cloud based AI solutions
- Cloud based ML platforms
  - Heterogenous compute (CPUs, GPUs, TPUs, specialized accelerators)
  - Immense and cheap storage (Amazon S3)
  - ML lifecycle management service
- Tools for different stages of ML lifecycle
- Tools to automate the end-to-end ML pipeline
- Open source effort (tools, frameworks, solutions)
  - Easy customization of AI solutions
  - Prevents vendor lock-in
- Standardization efforts making deep AI models portable
  - Shorten time to move to production

Machine Learning System

A system is a composition of one or more software components, with possible interactions, deployed on a hardware platform with the purpose of achieving some performance objective.

A Machine Learning system is a system where one or more software components are machine learning based.



Infrastructure

- Compute units and accelerators, Memory, Storage, Network
- Resources can be acquired as bare metal, VMs/Containers on cloud
- Hardware can help improve performance pretty much everywhere in the pipeline
- Design better hardware
  - Adapt existing architectures to ML tasks.
  - Develop brand-new architectures for ML.
- Hardware compute precision affects performance
- Software: ML frameworks, libraries (MKL, CUDNN), drivers

(Learning) Algorithm

- General and domain specific architectures
- Hyperparameter tuning to extract the best performance
- Effects the resource requirements: compute (FLOPS), memory
- Performance (runtime) and scalability of an algorithm depends on:
  - Hardware/Infrastructure
  - Software platform (frameworks, libraries, drivers)

Data

- Data as a critical element; Data is the King in ML
- Different modalities: audio, video, images, text
- Data sources, collection, labeling, quality, data storage
- Data type determines the choice of learning algorithm
- Making the data *business ready* is challenging
- Many data-driven organizations are spending 80 percent of their time on data preparation and find it a major bottleneck.
- DataOps

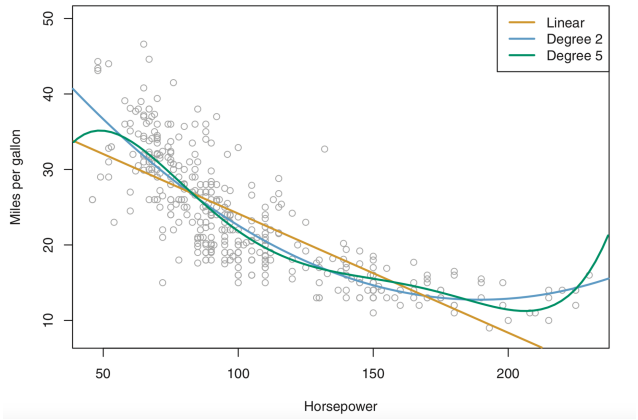
ML Tasks: Classification and Regression

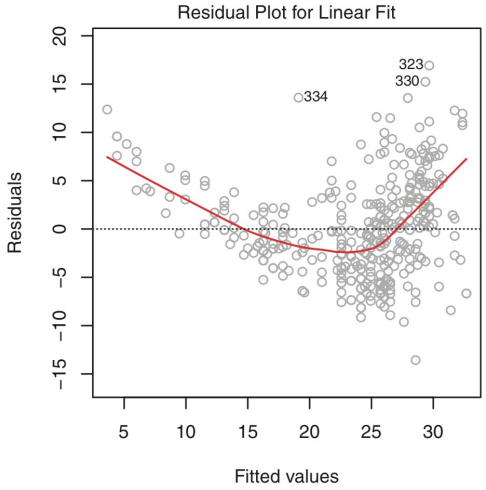
- **Regression**: Output is a continuous valued real variable
  - Stock price prediction
  - House value price prediction

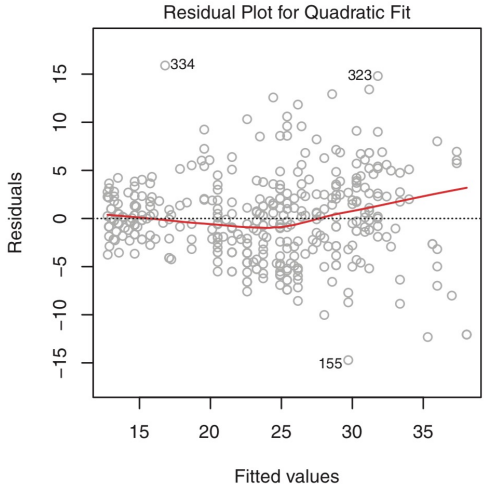
- **Classification**: Output is a categorical, unordered variable
  - Binary classification
    - Tumor: benign vs malignant
    - Food image : Greek vs Non-Greek
  - Multiclass classification
    - Tumor: benign, stage-1, stage-2, stage-3
    - Food image: falafel, salad, pita, …

Linear Regression

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \cdots + \hat{\beta}_p x_p$$







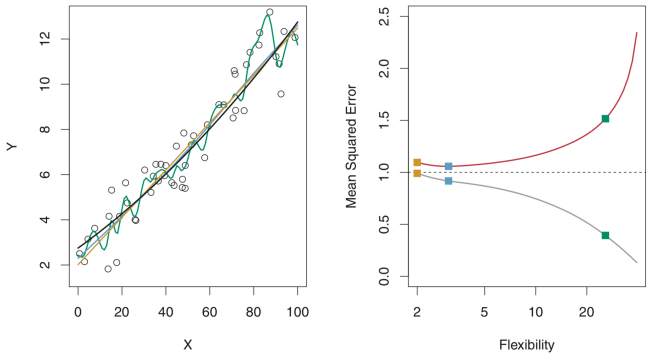
$$\mathrm{TSS} = \sum (y_i - \bar{y})^2$$

$$\mathrm{RSS} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

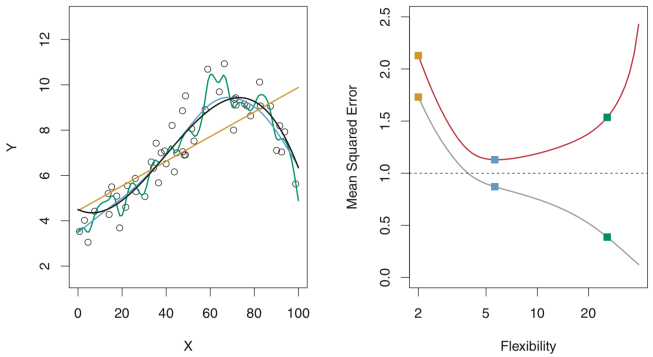
$$R^2 = \frac{\mathrm{TSS} - \mathrm{RSS}}{\mathrm{TSS}} = 1 - \frac{\mathrm{RSS}}{\mathrm{TSS}}$$

Mean Square Error (MSE)

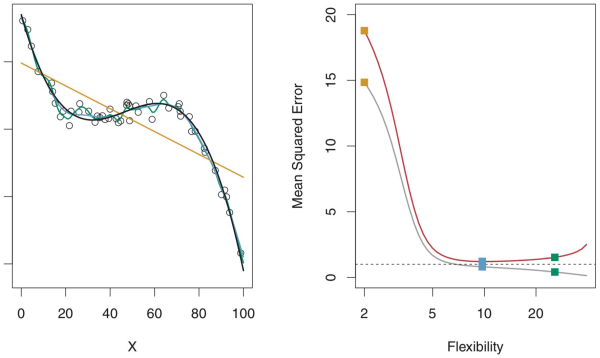
CASE 1



CASE 2



CASE 3



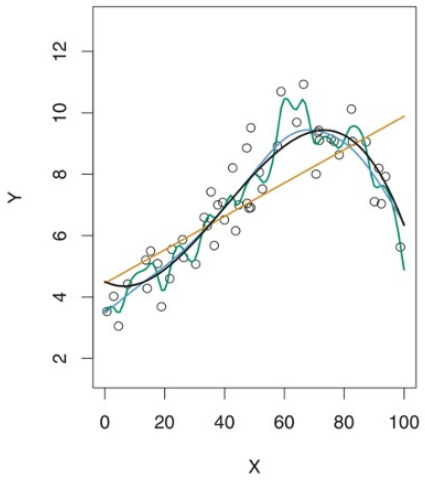
true value $Y = f(X) + \epsilon.$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2$$

predicted $\hat{Y} = \hat{f}(X)$

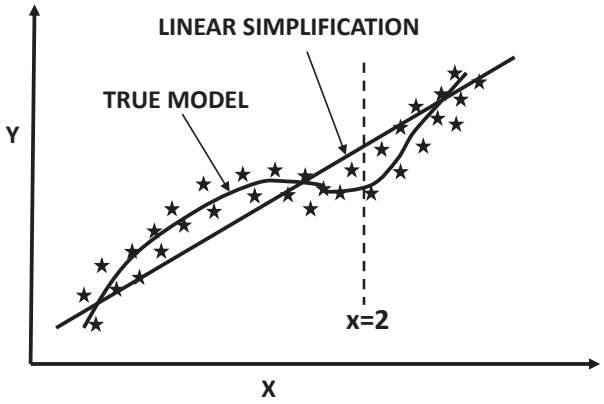
Overfitting and Underfitting

- Overfitting: model performs well on training data but does not generalize well to unseen data (test data)
- Underfitting: model is not complex enough to capture pattern in the training data well and therefore suffers from low performance on unseen data



Bias of a model

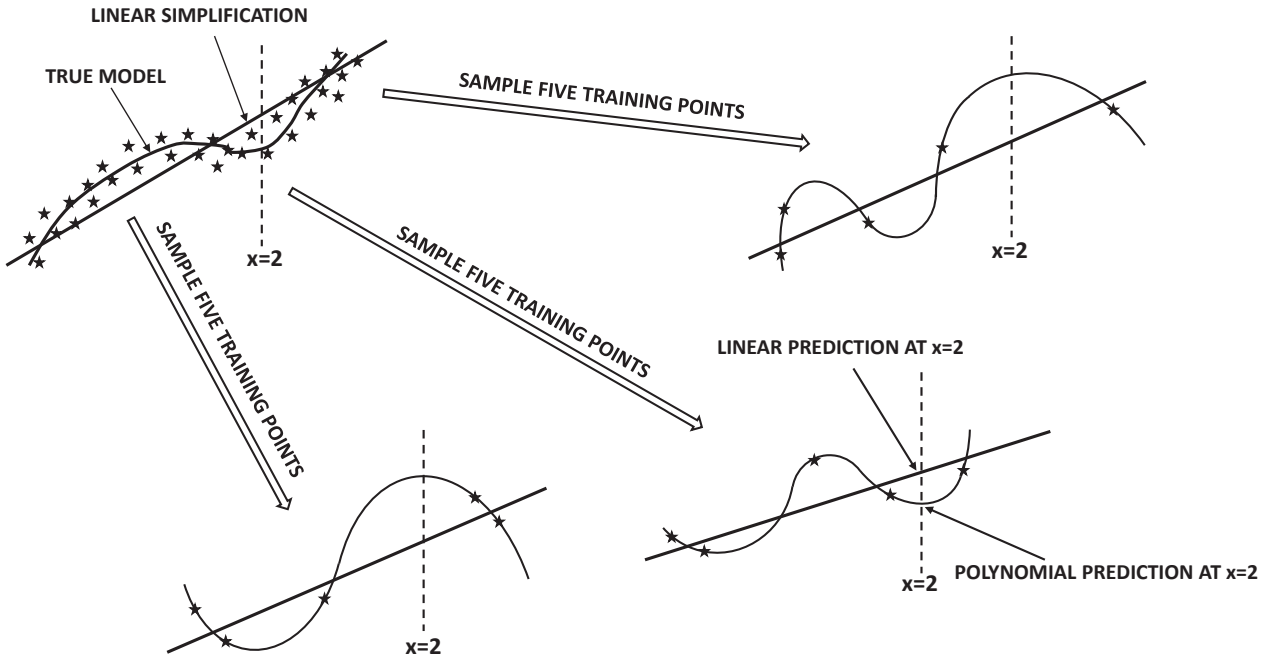
Example: Predict y from x



- **First impression:** Polynomial model such as $y = w_0 + w_1 x + w_2 x^2 + w_3 x^3 + w_4 x^4$ is "better" than linear model $y = w_0 + w_1 x$.

  –

Different Training Data Sets with Five Points



- Zero error on training data but wildly varying predictions of $x = 2$

Observations

- The higher-order model is more complex than the linear model and has less *bias*.
  - But it has more parameters.
  - For a small training data set, the learned parameters will be more sensitive to the nuances of that data set.
  - Different training data sets will provide different predictions for y at a particular x.
  - This variation is referred to as model *variance*.

Noise Component

- Unlike bias and variance, noise is a property of the *data* rather than the model.

- Noise refers to unexplained variations ϵ_i of data from true model $y_i = f(x_i) + \epsilon_i$.

- Real-world examples:

  – Human mislabeling of test instance $\Rightarrow$ Ideal model will never predict it accurately.

  – Error during collection of temperature due to sensor malfunctioning.

- Cannot do anything about it even if seeded with knowledge about true model.

Bias-Variance Trade-off: Setup

- Imagine you are given the true distribution $\mathcal{B}$ of training data (including labels).

- You have a principled way of sampling data sets $\mathcal{D} \sim \mathcal{B}$ from the training distribution.

- Imagine you create an infinite number of training data sets (and trained models) by repeated sampling.

- You have a *fixed* set $\mathcal{T}$ of unlabeled test instances.

  - The test set $\mathcal{T}$ does not change over different training data sets.

  - Compute prediction of each instance in $\mathcal{T}$ for each trained model.

Informal Definition of Bias

- Compute averaged prediction of each test instance x over different training models $g(x, \mathcal{D})$.

- Averaged prediction of test instance will be different from true (unknown) model $f(x)$.

- Difference between (averaged) $g(x, \mathcal{D})$ and $f(x)$ caused by erroneous assumptions/simplifications in modeling $\Rightarrow$ Bias

  - **Example:** Linear simplification to polynomial model causes bias.

  - If the true (unknown) model $f(x)$ were an order-4 polynomial, and we used any polynomial of order-4 or greater in $g(x, \mathcal{D})$, bias would be 0.

Informal Definition of Variance

- The value $g(x, \mathcal{D})$ will vary with $\mathcal{D}$ for fixed x.
  - The prediction of the same test instance will be different over different trained models.
- All these predictions cannot be simultaneously correct $\Rightarrow$ Variation contributes to error
- Variance of $g(x, \mathcal{D})$ over different training data sets $\Rightarrow$ Model Variance
  - **Example:** Linear model will have low variance.
  - Higher-order model will have high variance.

Bias-Variance Equation

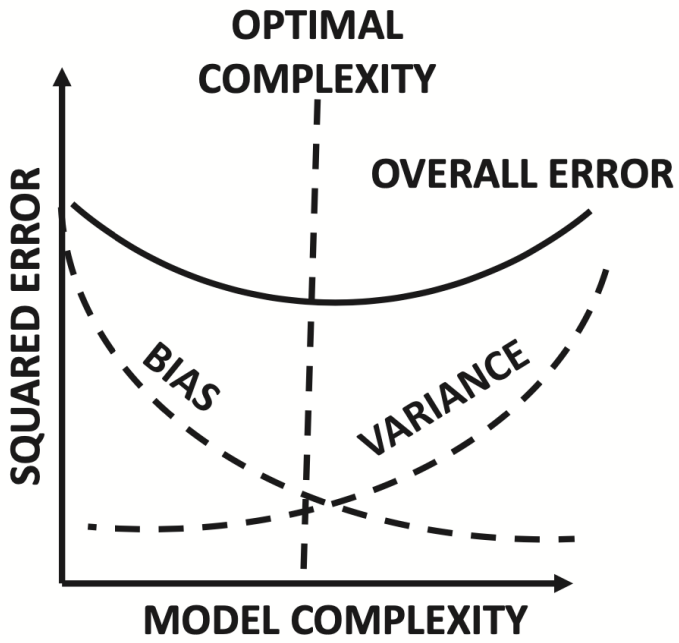
- Let $E[MSE]$ be the expected mean-squared error of the fixed set of test instances over different samples of training data sets.

$$E[MSE] = \text{Bias}^2 + \text{Variance} + \text{Noise} \quad (1)$$

  - In linear models, the bias component will contribute more to $E[MSE]$.
  - In polynomial models, the variance component will contribute more to $E[MSE]$.

- We have a trade-off, when it comes to choosing model complexity!

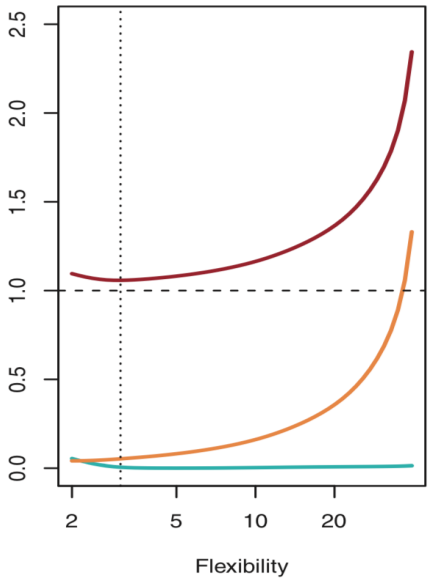
The Bias-Variance Trade-Off



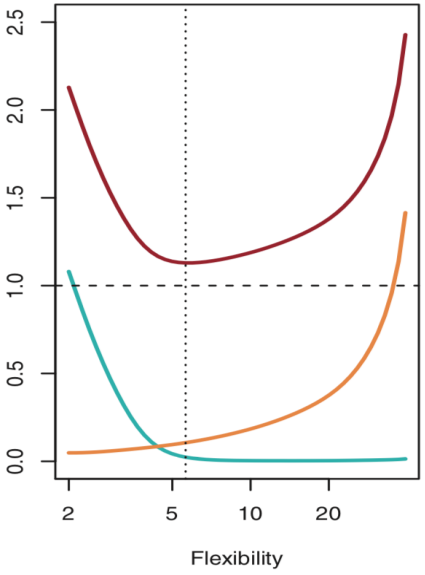
- Optimal point of model complexity is somewhere in middle.

Model Complexity Tradeofffs

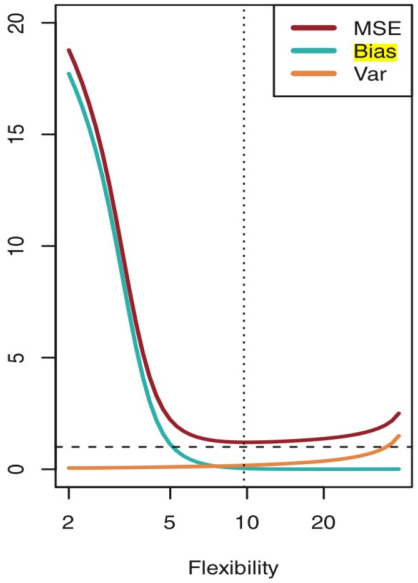
CASE 1



CASE 2



CASE 3

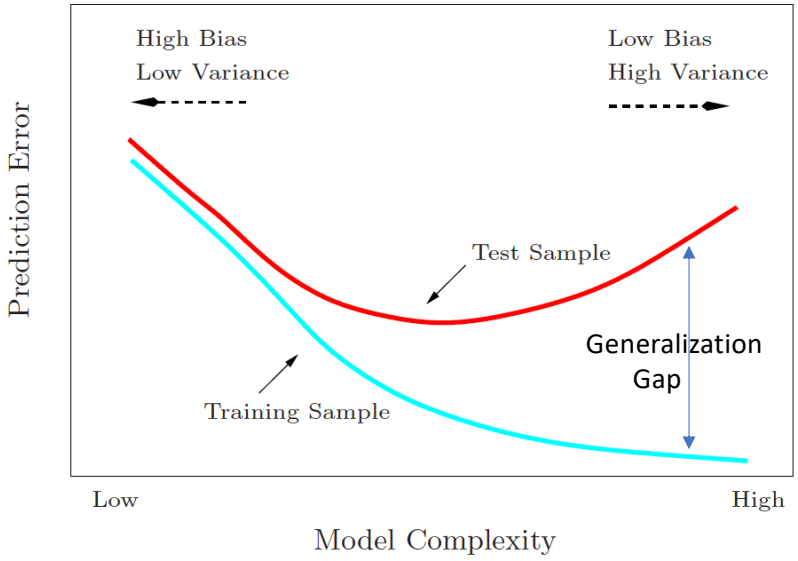


$$E\left(y_0 - \hat{f}(x_0)\right)^2 = \mathrm{Var}(\hat{f}(x_0)) + [\mathrm{Bias}(\hat{f}(x_0))]^2 + \mathrm{Var}(\epsilon)$$

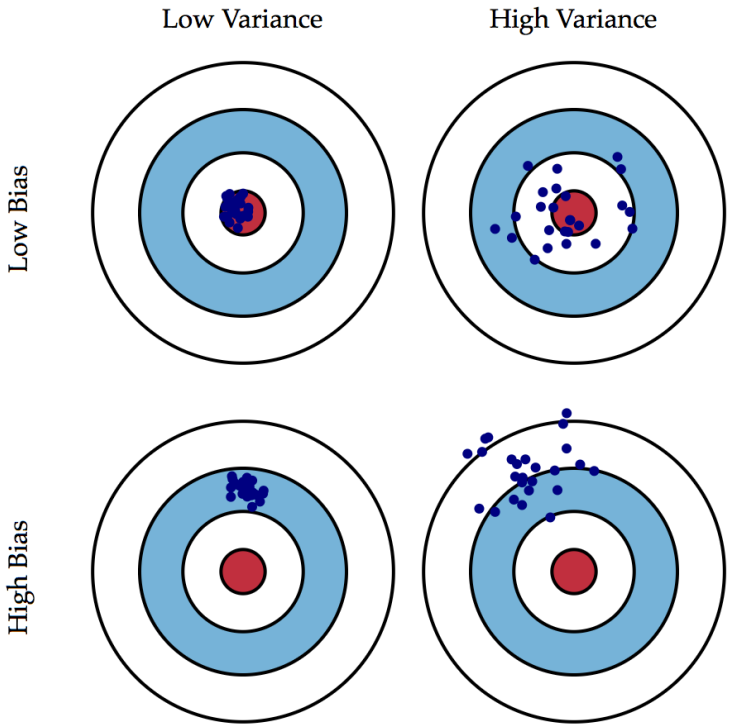
Irreducible

Model Complexity Tradeoffs

- Simple model
  - Fail to completely capture the relationship between features
  - Introduce bias: Consistent test error across different choices of training data
  - Low variance
  - Increasing training data does not help in reducing bias
- Complex model captures nuances in training data causing Overfitting
  - Low bias
  - Train error << Test error
  - With different training instances, the model prediction for same test instance will be very different – High Variance
- Variance does not depend on the true value of the test data



Bias-Variance Tradeoff



$$MSE_{test} = Bias^2 + Variance + Irreducible\ error$$

High Bias

Increase model complexity

Low Bias

High Variance

Increase training data

Low Variance

Bias-variance tradeoff example

- https://dustinstansbury.github.io/theclevermachine/bias-variance-tradeoff

Prepare for Lecture 2

- Take Quiz0
- Attend lab session. Attendance mandatory in lab sessions.
- Attend lectures. You will get credit.