THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

Applied Machine Intelligence and Reinforcement Learning

Professor Hamza F. Alsarhan

SEAS 8505 – DA2

Lecture 1

June 15, 2024

Welcome to SEAS Online at George Washington University

Class will begin shortly

Audio: To eliminate background noise, please be sure your audio is muted. To speak, please click the hand icon at the bottom of your screen (Raise Hand). When instructor calls on you, click microphone icon to unmute. When you've finished speaking, *be sure to mute yourself again*.

Chat: Please type your questions in Chat.

Recordings: As part of the educational support for students, we provide downloadable recordings of each class session to be used exclusively by registered students in that particular class for their own private use. Releasing these recordings is strictly prohibited.

Agenda

- Course Overview
- Machine Learning Landscape
- End to End Machine Learning Project
- Classification and Machine Learning Metrics
- Intro to Google Colab
- Assignment Overview for This Week

About The Professor

Some things about me

- Academic Background
  - D.Eng. in EM, GWU
  - MEM, Dartmouth College
  - BE, Computer Engineering, Dartmouth College
  - BA, Computer Science, Middlebury College
- Professional Background
  - Founder of an AI-based tech startup
  - VP of Customer Experience & Programs, Unirac
  - Director of Digital Lending, GLS
  - Financial Engineer, CMBS, Moody's Analytics
  - iOS Engineer, iDialogs

THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

Course Overview

Professor Hamza F. Alsarhan
SEAS 8505

Course Schedule and Logistics

Class Time and Dates:

- Day and Time: Saturdays, 9:00 am – 12:00 pm (Eastern)
- All Class Meeting Dates: Jun. 15, 22, 29; Jul. 6, 13, 20, 27; Aug. 3, 10, 17
- Online classes are conducted via Zoom; Links are provided in Blackboard
- Zoom link for Office Hours: https://gwu-edu.zoom.us/my/alsarhan

Assignments:

- Weekly Discussions on Blackboard
- Final Discussion Report
- Homework Assignments – eight (8) assignments
- Exams – a midterm and a final exam, administered on Blackboard outside the class meeting time

Grading:

- Homework, totaling: 20% (due by 9:00 am Eastern on Saturdays per the class schedule)
- Discussion Board and Final Report: 5% (due by 9:00 am Eastern on Saturdays per the class schedule)
- Midterm Exam: 35%
- Final Exam: 40%
- No extra credit assignments will be available

Office Hours:

- Mondays and Tuesdays, 6:00 pm – 7:30 pm ET

Exams

- Midterm and Final Exams:
  - The mid-term will be released at 8 pm ET on Saturday, July 13 and must be started no later than 5 pm Eastern on the following Monday.
    - There is a mandatory HonorLock practice exam designed to make sure that your computer is compatible with HonorLock, and you have no IT issues accessing the midterm/final exam. It will take approximately 5 minutes, and you must complete it before the end of Week 4.
  - The final exam will be released at 8 pm ET on Saturday, August 17, the last week of classes, and must be started no later than 5 pm Eastern on the following Monday.
- Students may use only native calculators on the PC or Mac.
- Students may use one formula sheet (letter size 8.5”x11”, front and back)

Exam Policy Violation Examples and Consequences (please see syllabus for full details)

- **Minor Violations** – radio/TV in the background, someone enters the room, sitting on a couch, any part of face out of camera view briefly (less than 5 minutes in total), second monitor (off) on the desk, improper lighting, using headphones, wearing hats, sunglasses, etc.
  - If you are flagged for a minor violation, you receive a warning for the first offense. Students who commit minor violations after being warned will be penalized 10% on the exam.
  - Subsequent minor violations could result in referral to the office of academic integrity. Minor violations will be counted cumulatively across the entire program.
- **Major Violations** - browsing the web, using the phone or other devices, using additional screens, any part of face out of camera view (more than 5 min), communicating with another individual by any means.
  - If you are flagged for a major violation, you will receive a 20% reduction on the exam and may be referred to the office of academic integrity.
- Written work must comply with the Academic Integrity Policy of the George Washington University policy. Any plagiarized material will receive a grade of 0.

Course Overview

- A broad introduction to fundamental concepts and techniques in machine learning from the perspective of the systems engineer.
- The field of machine learning explores algorithms that can learn from examples (e.g. experience) without pre-programmed rules or that can make predictions based on automated analysis of prior data.
- This course provides students with knowledge of the theory and practice of machine learning leveraging an open-source framework to explore the ideas, algorithms and techniques, without a prior background in programming.

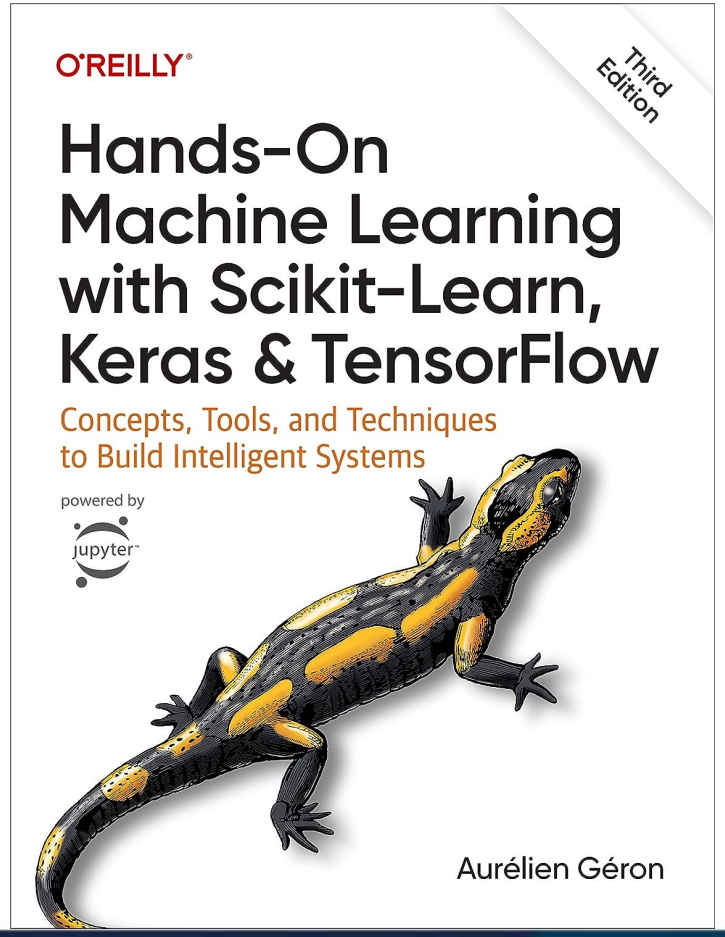
Class Schedule and Assignments

Class	Topic/Activity	Assignment Due
1	Course Overview, How to Apply Machine Learning to Projects (from the beginning of a project to the end of a project) Chapters 1, 2, and 3	None
2	Model training and Decision Trees Chapters 4 and 6	HW #1 (Due by 9:00 am ET on Jun. 22) Discussion #1
3	Support Vector Machines (SVMs) Chapter 5	HW #2 (Due by 9:00 am ET on Jun. 27) Discussion #2
4	Ensembles (Applications and Random Forests) Chapter 7	HW #3 (Due by 9:00 am ET on Jul. 6) Discussion #3
5	Dimensionality Reduction and Clustering Chapters 8 and 9	HW #4 (Due by 9:00 am ET on Jul. 13) Discussion #4 Miderm Exam: Jul. 13, 8 pm ET to Jul. 15, 5pm ET
6	Neural Networks and Deep Learning Applications Chapters 10 and 11 Introduction to Tensorflow	No HW or Discussion
7	Sequential Data Processing with Recurrent Neural Networks (RNNs) and LSTMs Chapter 15	HW #5 (Due by 9:00 am ET on Jul. 27) Discussion #5
8	Auto-Encoders and Attention/Transformers Chapters 16 and 17	HW #6 (Due by 9:00 am ET on Aug. 3) Discussion #6
9	Generative Adversarial Networks (GANs) and Reinforcement Learning Chapters 17 and 18	HW #7 (Due by 9:00 am ET on Aug. 10) Discussion #7
10	Emerging AI Topics/Applications	HW #8 (Due by 9:00 am ET on Aug. 17) Discussion Summary Final Exam: Aug. 17, 8 pm ET to Aug. 19, 5pm ET

Course Overview

- Email works best for communication with me. Please include SEAS 8505 in your email subject.
- If you must skip a class, please communicate with me as soon as possible to discuss a plan for you to make up for the missed class.
- Office hours are for you to ask questions and seek help. No new material will be covered during office hours. If you need to meet outside of these office hours, we can discuss scheduling meetings by appointment.

Course Textbook



Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 3rd Edition.

Aurélien Géron
Published by O'Reilly Media, Inc.

Available through the Gelman Library
https://wrlc-gwu.primo.exlibrisgroup.com/discovery/fulldisplay?context=L&vid=01WRLC_GWA:live&search_scope=DN_and_CI&isFrbr=true&tab=Everything&docid=alma9918628036850410 7

Course Topics (first half of course)

Class	Topic/Activity	Readings
1	Course Overview, How to Apply Machine Learning to Projects (from the beginning of a project to the end of a project)	Chapters 1, 2, and 3
2	Model training and Decision Trees	Chapters 4 and 6
3	Support Vector Machines (SVMs)	Chapter 5
4	Ensembles (Applications and Random Forests)	Chapter 7
5	Dimensionality Reduction and Clustering	Chapters 8 and 9

Course Topics (second half of course)

Class	Topic/Activity	Readings
6	Neural Networks and Deep Learning Applications Introduction to Tensorflow	Chapters 10 and 11
7	Sequential Data Processing with Recurrent Neural Networks (RNNs) and LSTMs	Chapter 15
8	Auto-Encoders and Attention/Transformers	Chapters 16 and 17
9	Generative Adversarial Networks (GANs) and Reinforcement Learning	Chapters 17 and 18
10	Emerging AI Topics/Applications	Assigned by Instructor

Other Potential References

1. The Hundred-Page Machine Learning Book. Andriy Burkov, 2019
   - http://themlbook.com/wiki/doku.php
2. Machine Learning Engineering. Andriy Burkov, 2020
   - http://www.mlebook.com/wiki/doku.php
3. Deep Learning. Ian Goodfellow, Yoshua Bengio, Aaron Courville, 2016
   - https://www.deeplearningbook.org/
4. Artificial Intelligence: A Modern Approach, 4th Ed. Stuart Russell and Peter Norvig, 2021
5. Introduction to Machine Learning. Ethem Alpaydin, 2014
6. Many other machine learning references …

Discussions

- At the beginning of the course, I will post an assignment prompt on the discussion board and you will be randomly assigned to a discussion group.
- You are responsible for spending at least one hour each week collaborating within your group, and individually posting a one-paragraph response on Blackboard discussion board for your discussion group to see.
- During the final week of class, you will submit a 1-2 page summary on an assigned topic.
- Mandatory. Calculated as part of your grade (includes your weekly posting as well as your end of semester report).
- Responses due by 9:00 am ET every Saturday.
- **Late submissions will receive a grade of 0.**

Homework

- Homework is comprised of eight (8) assignments:
  - Provide an opportunity for you to apply principles learned in class related to applied machine learning for engineers.
  - HW assignments involve running a Python script and using its results to answer some questions.
  - For each assignment, the requirement is to submit your work on time, and it will be graded based on correctness and completeness.
  - All homework assignments are **due by 9:00 am ET on Saturdays** per the class schedule.
  - **Late submissions will receive a grade of 0.**

Grading

- Homework 20%
- Discussion Board 5%
- Midterm Exam 35%
- Final Exam 40%

- GW's grading system for graduate students is: A, Excellent; B, Good; C, Satisfactory; F, Fail; other grades that may be assigned are A-, B+, B-, C+, C-.

Overall Approach for the Course

- While we will cover some theory and mathematics, our focus will be on how to correctly apply machine learning from a systems perspective.

- Material for exam questions will come from class slides and assigned readings. The focus will be on concepts, how to apply machine learning, and what would be the best response for a given scenario.

THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

Why Machine Learning???

What is Machine Learning?

- Machine Learning is the science (and art) of programming computers, so they can learn from data.

- [Machine Learning is the] field of study that gives computers the ability to learn without being explicitly programmed. (Arthur Samuel, 1959)

- A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E. (Tom Mitchell, 1997)

Hands-On Machine Learning, Ch1, p2

What does it mean to learn?

- Class definition of learning

- Class definition of knowledge

What does it mean to learn?

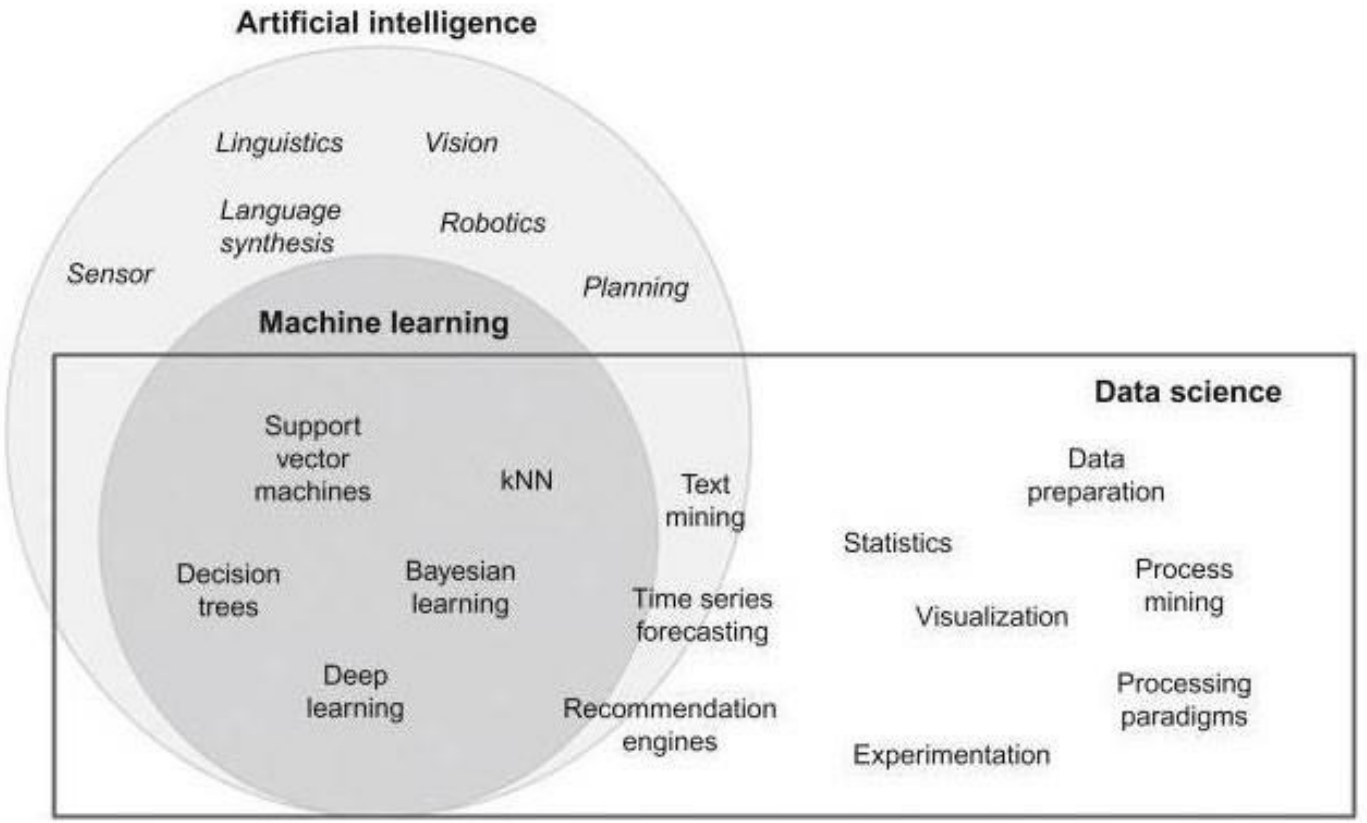
- Dictionary definition of learning
  - The acquisition of knowledge or skills through experience, study, or by being taught.

- Class definition of knowledge
  - Facts, information, and skills acquired by a person through experience or education; the theoretical or practical understanding of a subject.

ML, AI, and Data Science (1 of 2)

- **Artificial Intelligence (AI):** Mimic the human brain and create systems that can function intelligently and independently

- **Machine Learning (ML):** Is a subset of AI. Focused on allowing a computer to learn using data and algorithms.

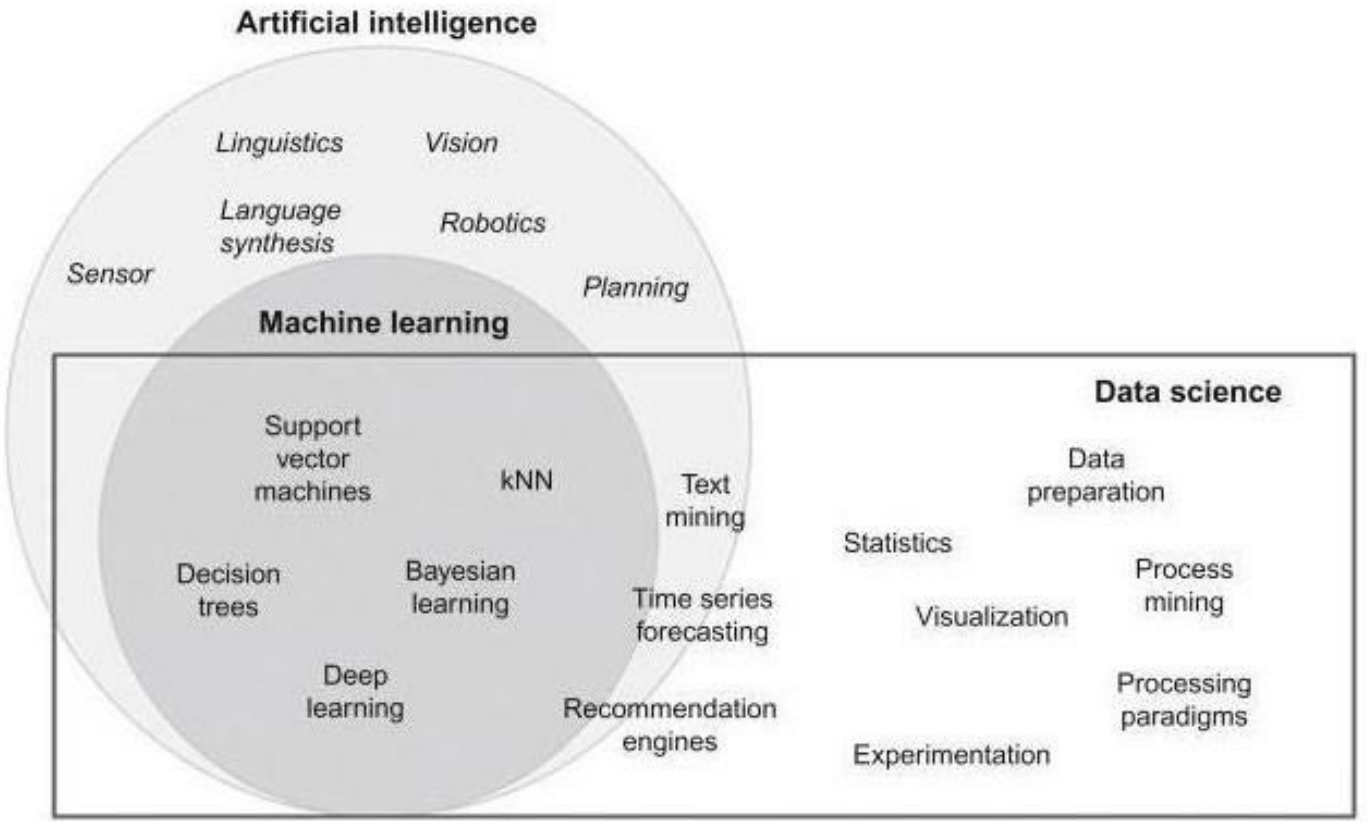


https://towardsdatascience.com/artificial-intelligence-vs-machine-learning-vs-data- science-2d5b57cb025b

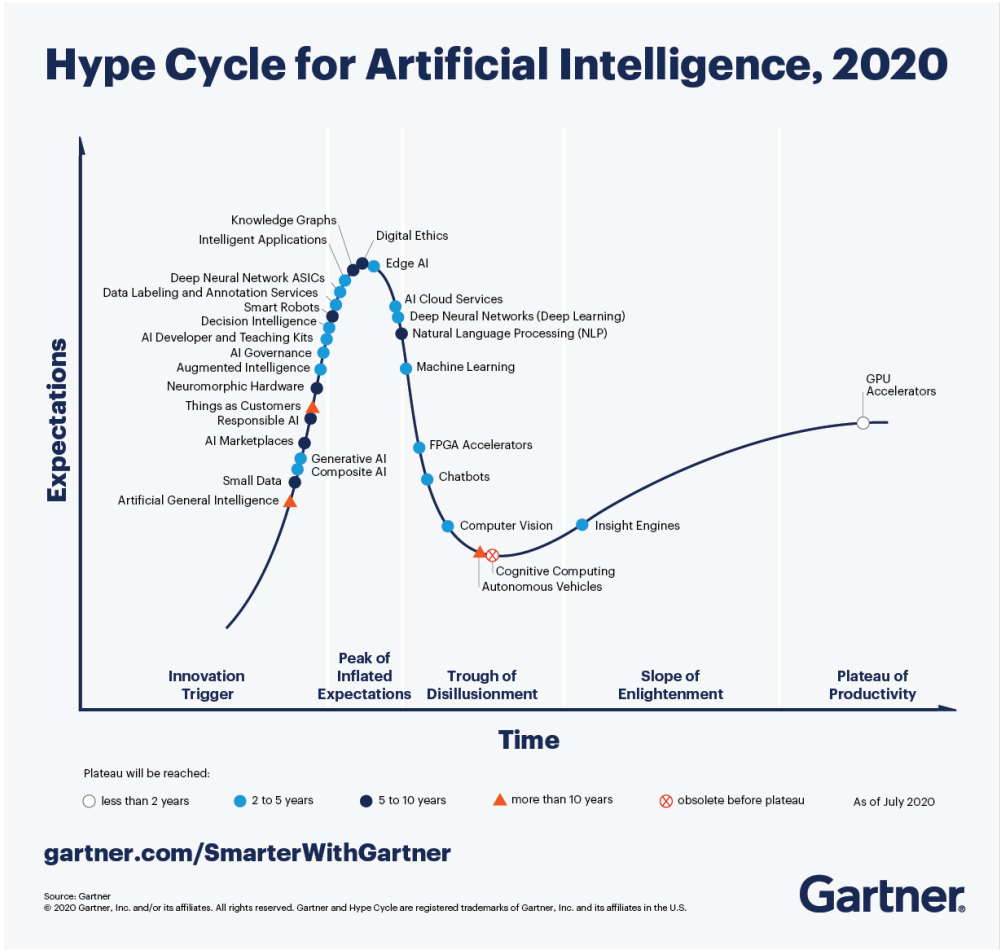
ML, AI, and Data Science (2 of 2)

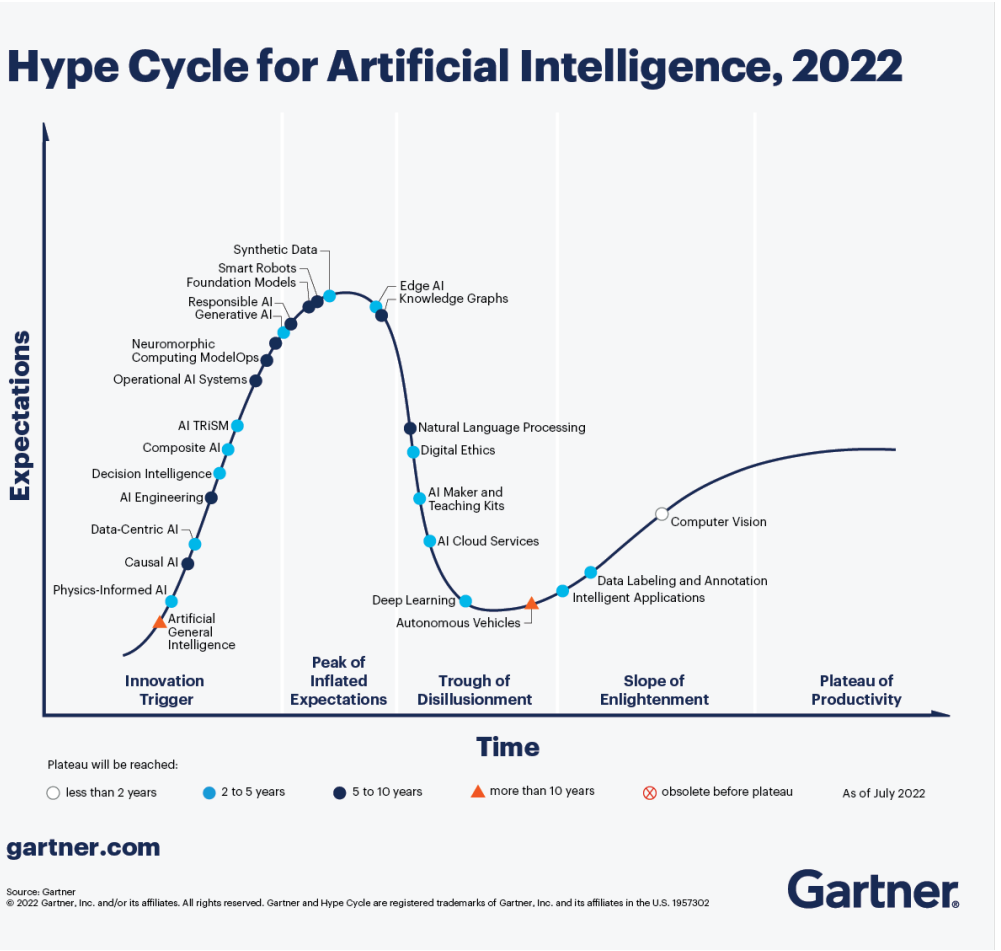
- **Data Science:** Focused on discovering actionable insights from large sets of raw (unstructured) and structured data.

- This course will not be strictly differentiating between Artificial Intelligence (AI), Machine Learning (ML), and Data Science
  - Overlapping algorithms / approaches



https://towardsdatascience.com/artificial-intelligence-vs-machine-learning-vs-data- science-2d5b57cb025b

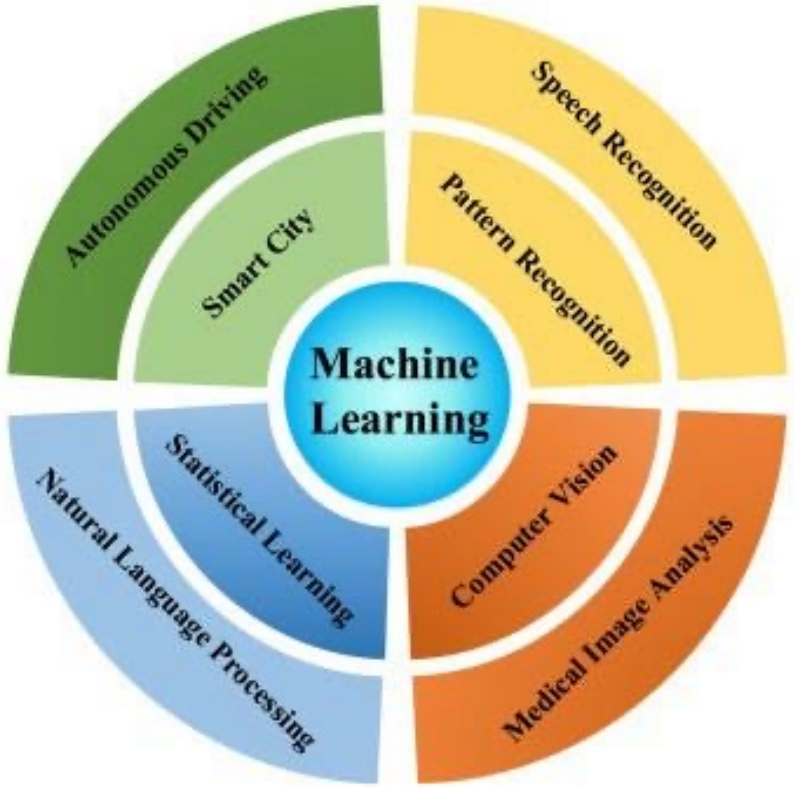




- "Autonomous Vehicles" and "Conversational AI" have moved further along the cycle by 2022, indicating they are progressing through the trough of disillusionment or into the slope of enlightenment.
- "NLP" (Natural Language Processing) in 2020 is at the peak of inflated expectations, but by 2022, it has moved down into the trough of disillusionment.
- What else do you notice?

https://www.gartner.com/smarterwithgartner/2-megatrends-dominate-the-gartner-hype-cycle-for-artificial-intelligence-2020/
https://www.gartner.com/en/articles/what-s-new-in-artificial-intelligence-from-the-2022-gartner-hype-cycle

Examples of ML

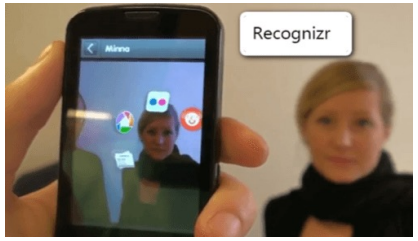


https://www.sciencedirect.com/science/article/pii/S157401372030071X

Successful AI/ML Applications

Language translation services (Google)

> "I won't spend a dime on this kind of effort."➔ "나는 노력의이 종류에 한푼도 지출하지 않습니다."



Speech translation (Google Translate)

News aggregation and summarization (Google)

Song recognition (Shazam)

Face and image recognition (Facebook)

Question answering (Google Home, Amazon Alexa, Apple Siri, IBM Watson)

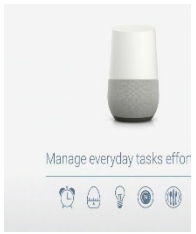
Board games (IBM Deep Blue, Google DeepMind AlphaGo)

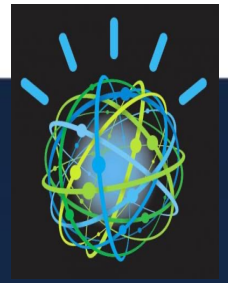
3D scene modeling from images (Microsoft Photosynth)

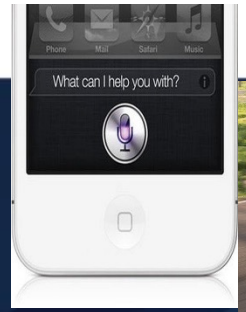
Driverless cars (Tesla, Google)

Traffic Alerts and fastest routes (Google Maps; Waze)

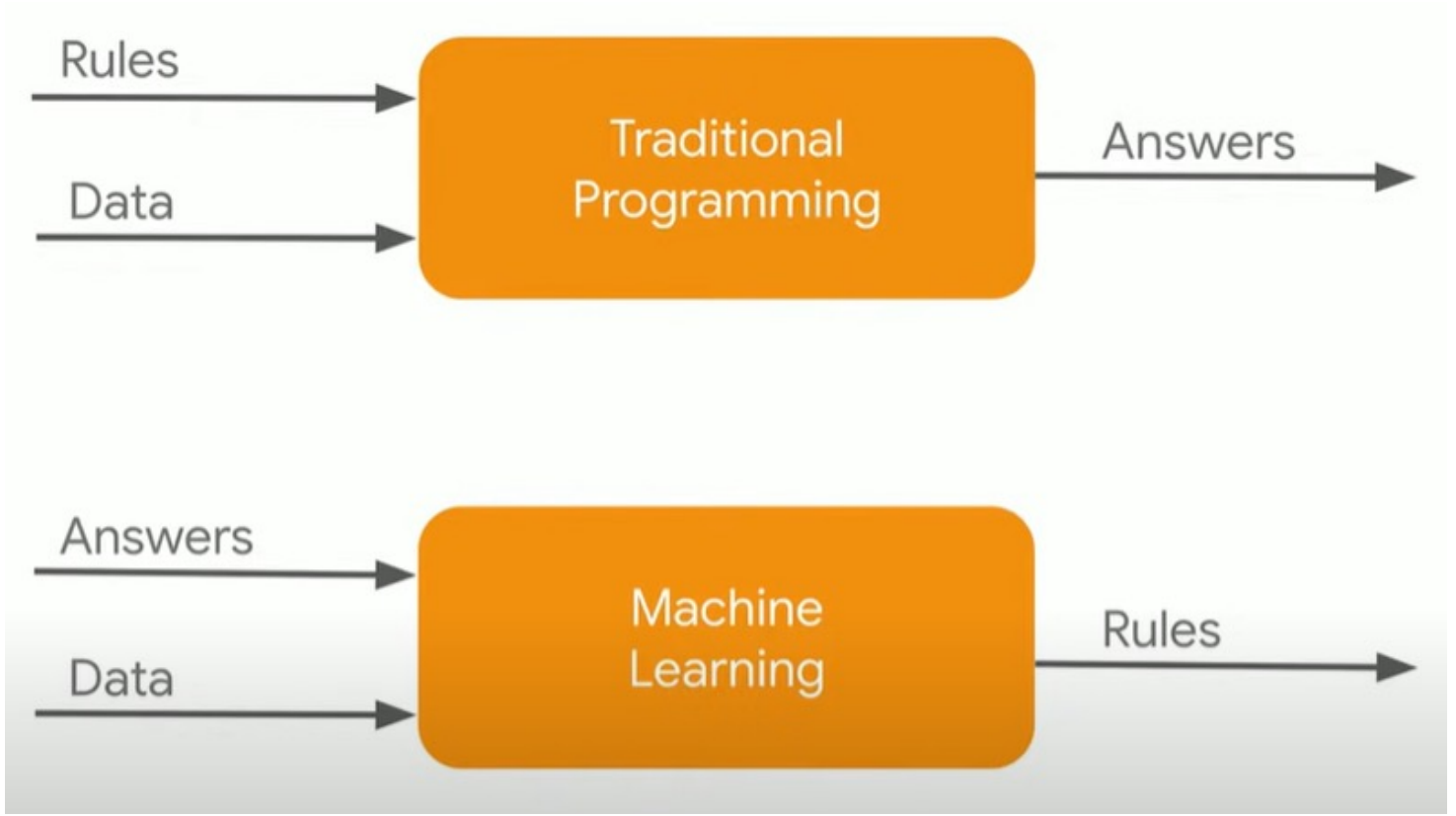
Understanding protein folding (DeepMind AlphaFold)







ML vs Traditional Programming



Machine Learning Zero to Hero (Google I/O'19)
https://www.youtube.com/watch?v=VwVg9jCtqaU

Rules Based Approach

- Can be effective when algorithm known/ simple problem.
- Hard to maintain if problem complex.
- Spam E-Mail Example:
  - If email contains “4U” >> Block
  - What happens when spammer changes to “For U”?

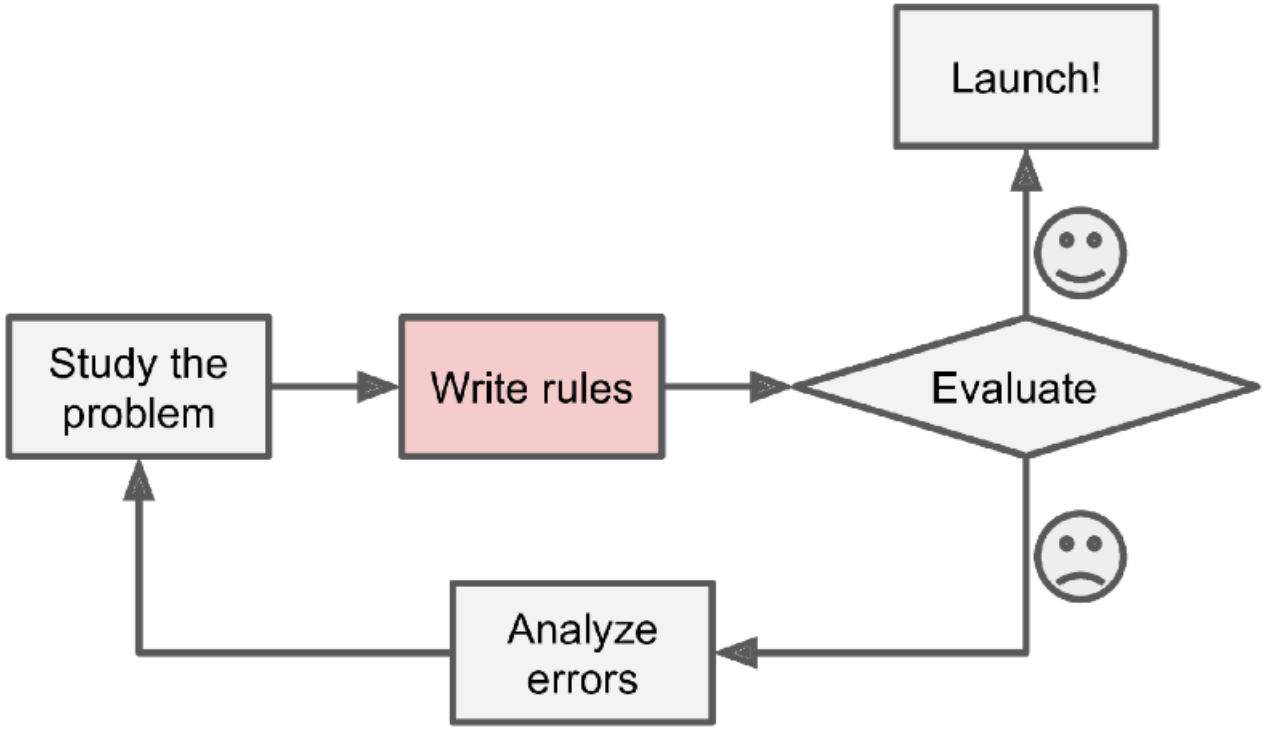


Figure 1-1. The traditional approach

Hands-On Machine Learning, Ch1, p3

Machine Learning Approach

- Automatically learns and can adapt
- Results in program that is shorter, easier to maintain, and most likely more accurate
- Spam E-Mail Example:
  - Learns words and phrases unusually frequent in spam examples

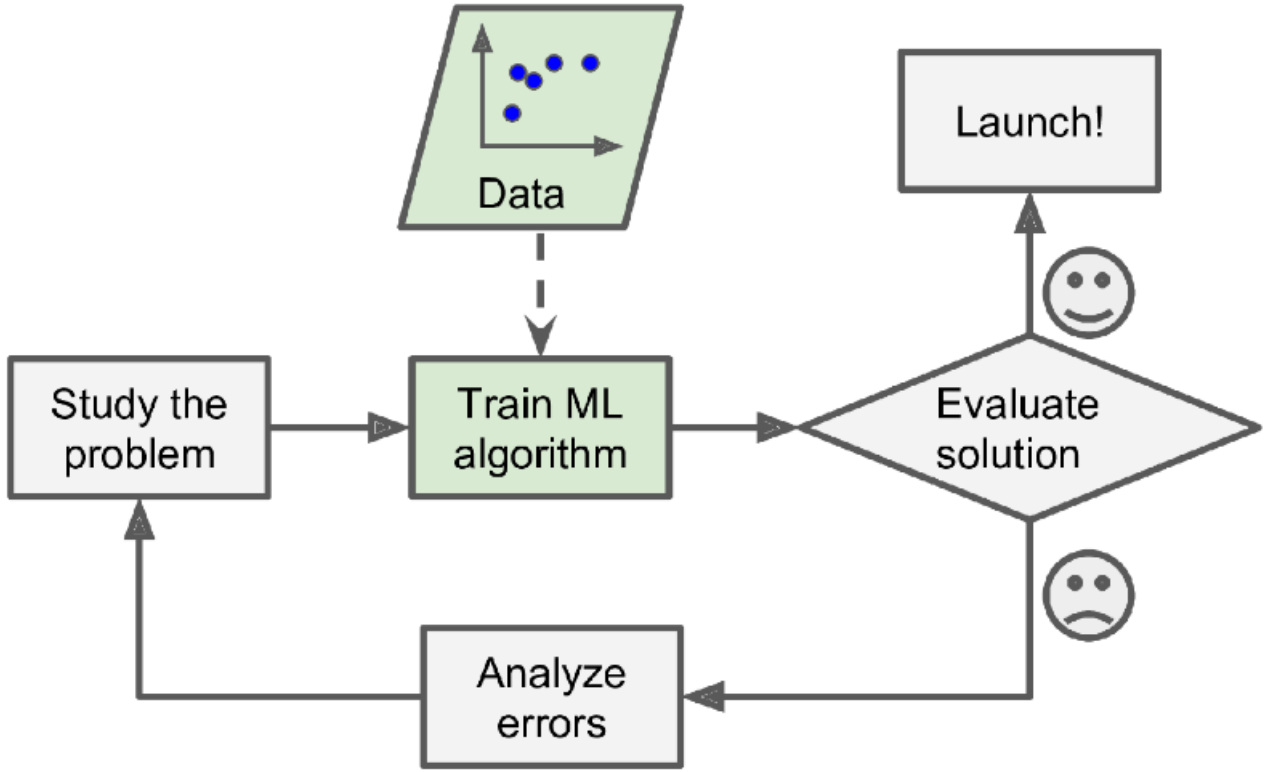


Figure 1-2. The Machine Learning approach

Hands-On Machine Learning, Ch1, p3-4

ML Can Help People Learn

- Inspect ML algorithms to see what they learned
- Apply to large amounts of data to discover patterns (data mining)

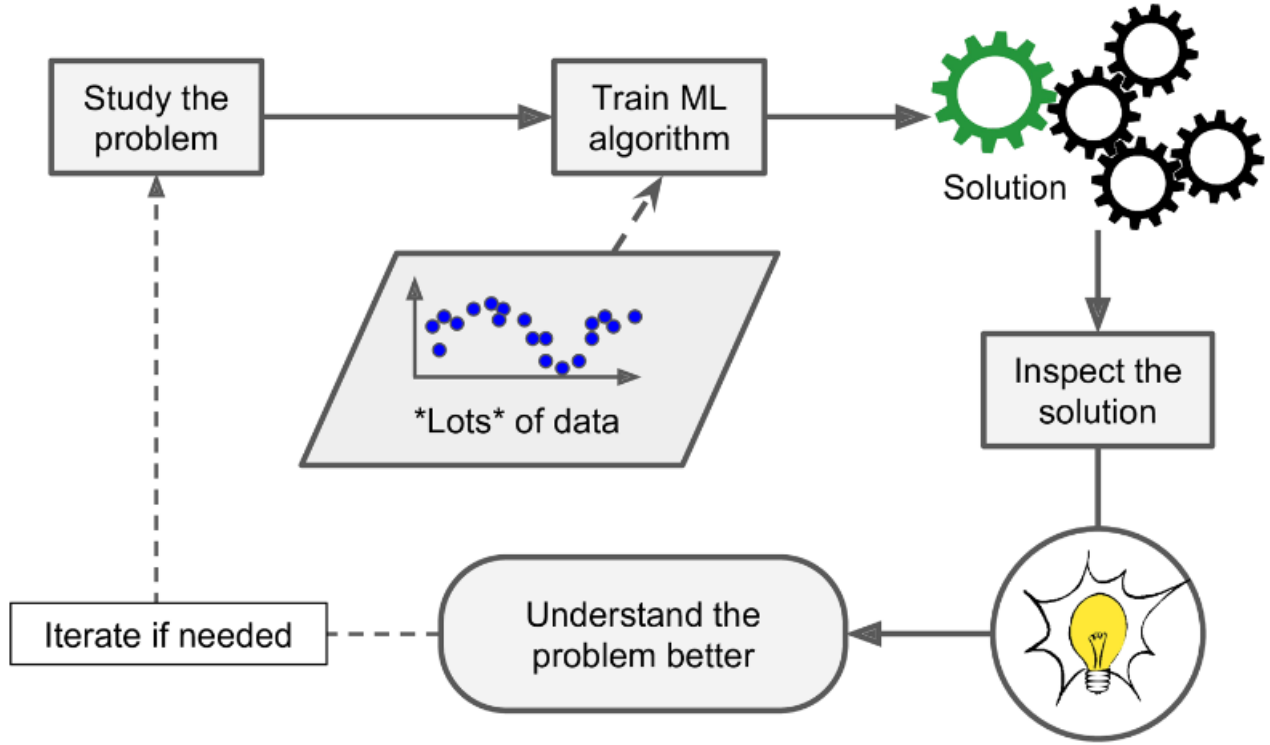


Figure 1-4. Machine Learning can help humans learn

Hands-On Machine Learning, Ch1, p4-5

Why Use Machine Learning?

- Problems too difficult/complex for traditional programming
- Industry 4.0 – the Fourth Industrial Revolution
  - Smart Factories & Connectivity
  - Enhanced Automation
  - Real-time Decision Making
- Data
  - Large quantity
  - Increasing complexity
  - Fluctuating environments
- Help humans learn

Hands-On Machine Learning, Ch1, p5

Machine Learning Has Improved Over Time – Why?

Machine Learning Has Improved Over Time – Why?

- Increasing amounts of data
- Better tools
- Computational power
- Need to improve decision making and overall project and organizational outcomes

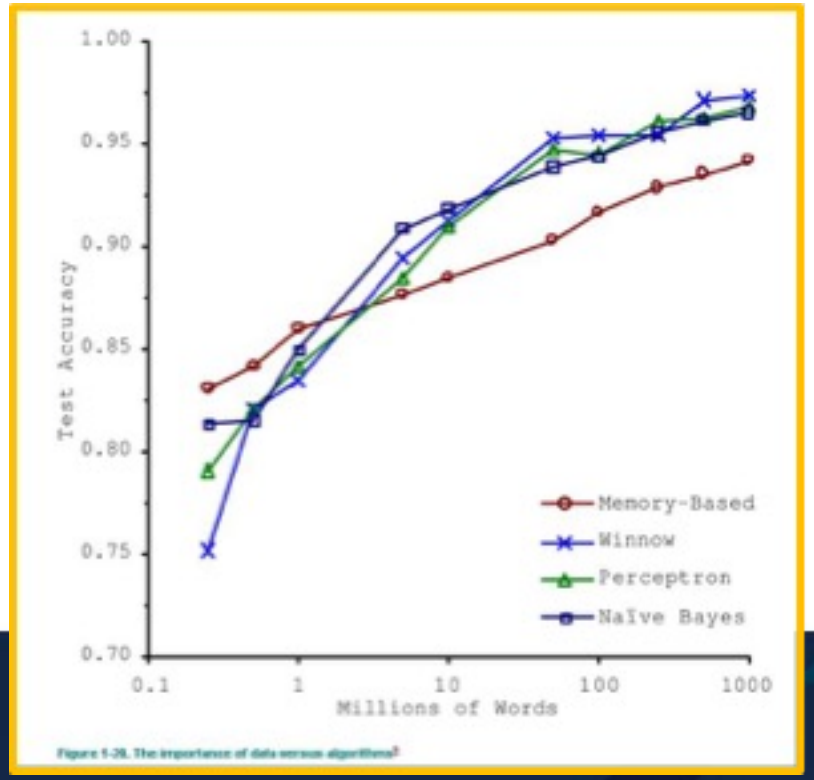
THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

Artificial Intelligence & Machine Learning Challenges

Professor Hamza F. Alsarhan
SEAS 8505

Machine Learning Challenge - Data

- Insufficient Quantity of Training Data
  - ML Algorithms require a lot of data to learn (which can be difficult and expensive to obtain)
  - Data matters more than algorithms – Peter Norvig et. al. paper “The Unreasonable Effectiveness of Data”



Hands-On Machine Learning, Ch1, p25-7

Challenge of Machine Learning - Bad Data

- Nonrepresentative Training Data
  - To generalize well, your training data should be representative of the new cases you want to generalize to
    - Sample Too Small ➔ Sampling Noise
    - Sample Method Flawed ➔ Sampling Bias

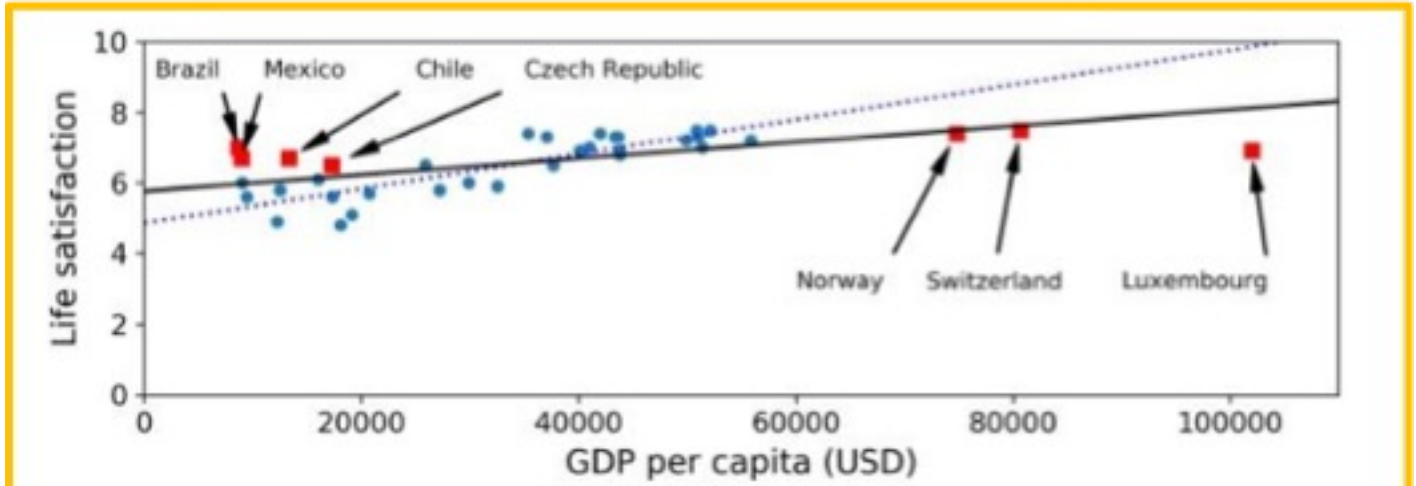


Figure 1-21. A more representative training sample

Hands-On Machine Learning, Ch1, p25-7

Challenge of Machine Learning - Bad Data

- Poor-Quality Data
  - Difficult to detect underlying patterns when data is full of errors, outliers, and noise
- Irrelevant Features
  - Garbage In →  → Garbage Out

Hands-On Machine Learning, Ch1, p25-7

Challenge of Machine Learning - Bad Algorithms

- Overfitting the Training Data
  - The model performs well on the training data but does not generalize well on new data
  - Model too complex relative to amount and noisiness of the data
  - Regularization - constraining a model to make it simpler and reduce the risk of overfitting
  - Hyperparameter – controls the amount of regularization to apply during learning
  - A hyperparameter is a parameter of a learning algorithm (not the model) which is set prior to training and remains constant during training
- Underfitting the Training Data
  - Occurs when the model is too simple (opposite of overfitting)
  - Reality is more complex than the model and predictions are inaccurate, even on training data

Hands-On Machine Learning, Ch1, p28-9

Stepping Back – The Bigger Picture

- Machine Learning is about making machines get better at some task by learning from data, instead of having to explicitly code rules.
- Many types of ML systems:
  - Supervised, Unsupervised, Semisupervised, Reinforcement Learning
  - Batch vs Online
  - Instance vs Model-based
- In an ML project you gather data for training a learning algorithm
  - Model-based algorithm tunes some parameters to fit the model to the training set (i.e., to make good predictions on the training set itself), and then hopefully it will be able to make good predictions on new cases as well.
  - Instance-based algorithm learns the examples by heart and generalizes to new instances by using a similarity measure to compare them to learned instances.
- The system will not perform well if your training set is too small, or if the data is not representative, is noisy, or is polluted with irrelevant features (garbage in, garbage out).
- Lastly, your model needs to be neither too simple (in which case it will underfit) nor too complex (in which case it will overfit).

Hands-On Machine Learning, Ch1, p30

Testing & Validating

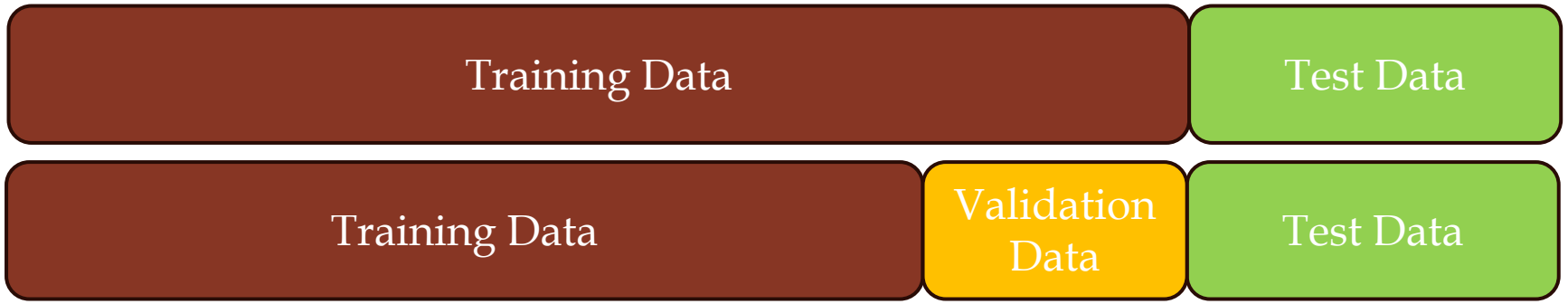
- The only way to know how well a model will generalize to new cases is to try it out on new cases.

- Split your data into two sets: the training set and the test set
  - Train your model using the training set
  - Test using the test set.
    - Generalization error (out-of-sample error) is the error rate on new cases. Tells you how well your model will perform on instances it has never seen before.

Hands-On Machine Learning, Ch1, p30

Hyperparameter Tuning and Model Selection

- Holdout validation: hold out part of the training set (i.e. validation set) to evaluate several candidate models and select the best one.



- Train multiple models with various hyperparameters on the reduced training set (i.e., the full training set minus the validation set), and select the model that performs best on the validation set.

- After this holdout validation process, train the best model on the full training set (including the validation set), and this gives you the final model.

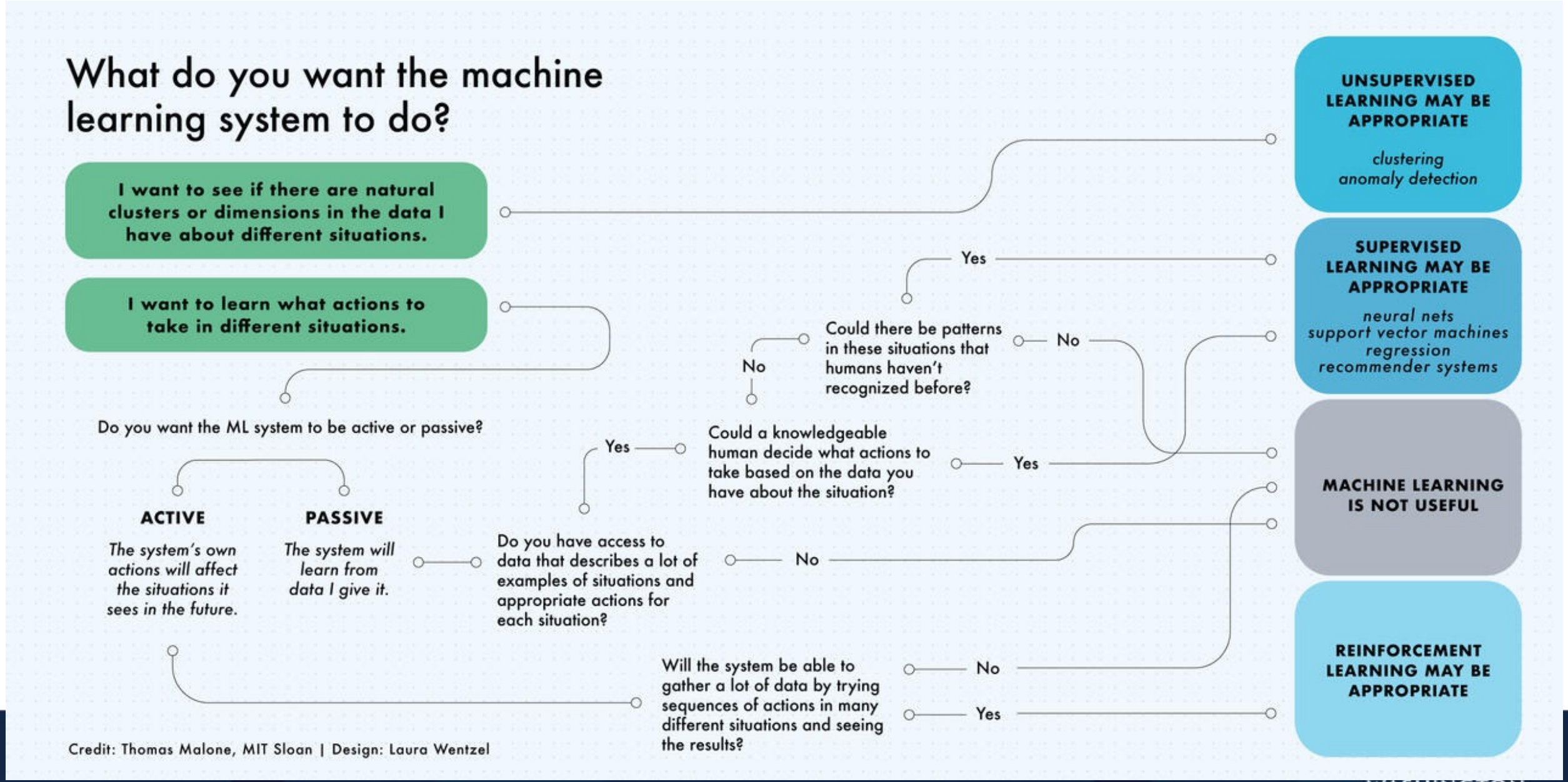
- Lastly, evaluate this final model on the test set to get an estimate of the generalization error.

Hands-On Machine Learning, Ch1, p31

THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

ML Engineering Overview

Professor Hamza F. Alsarhan
SEAS 8505



https://mitsloan.mit.edu/ideas-made-to-matter/machine-learning-explained

Machine Learning Pipeline

- An ML pipeline consists of several components, as the diagram shows. The "Model" (a black box) is a small part of the pipeline infrastructure necessary for production ML.

- In ML, starting with tests is not straightforward. Your tests depend on your data, model, and problem. You need tests for:
  - Input data
  - Feature engineering
  - Model versions
  - Serving infrastructure
  - Pipeline component integration.

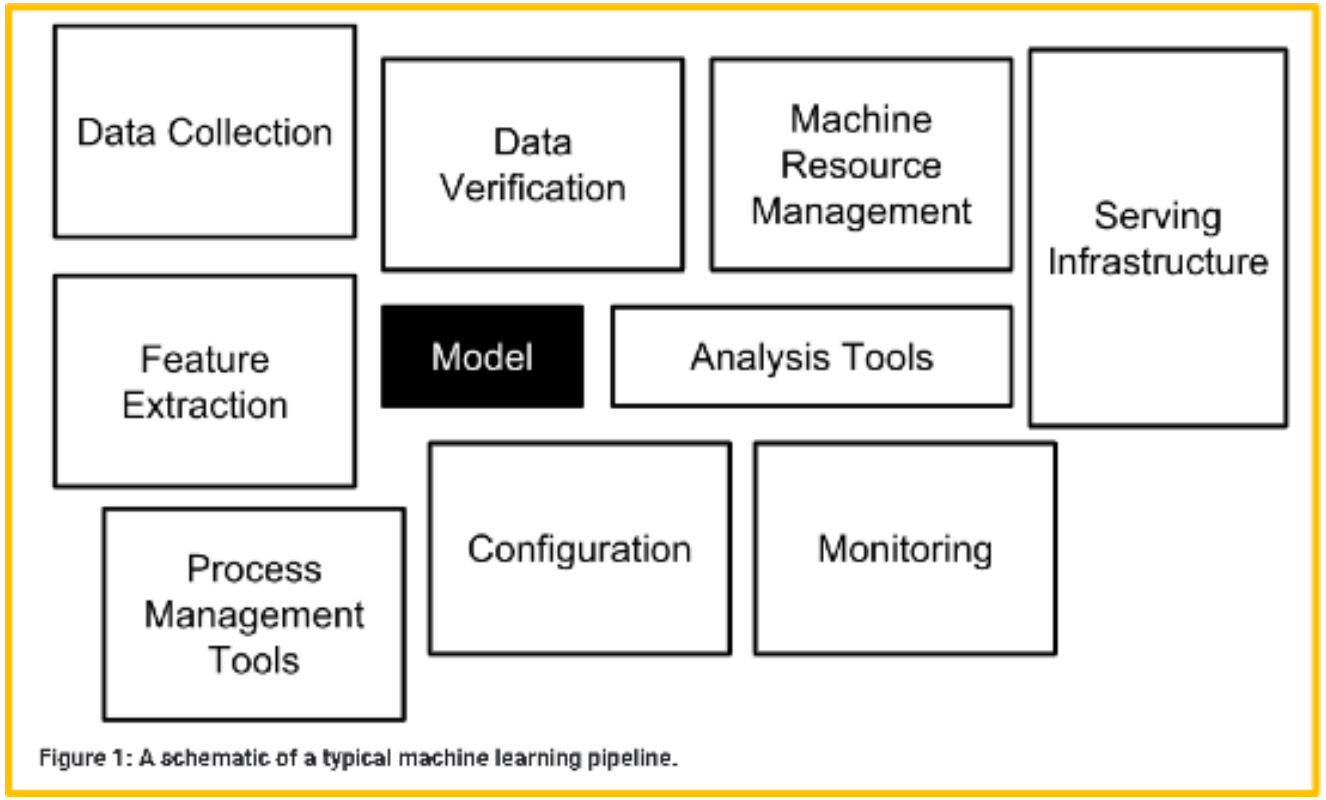
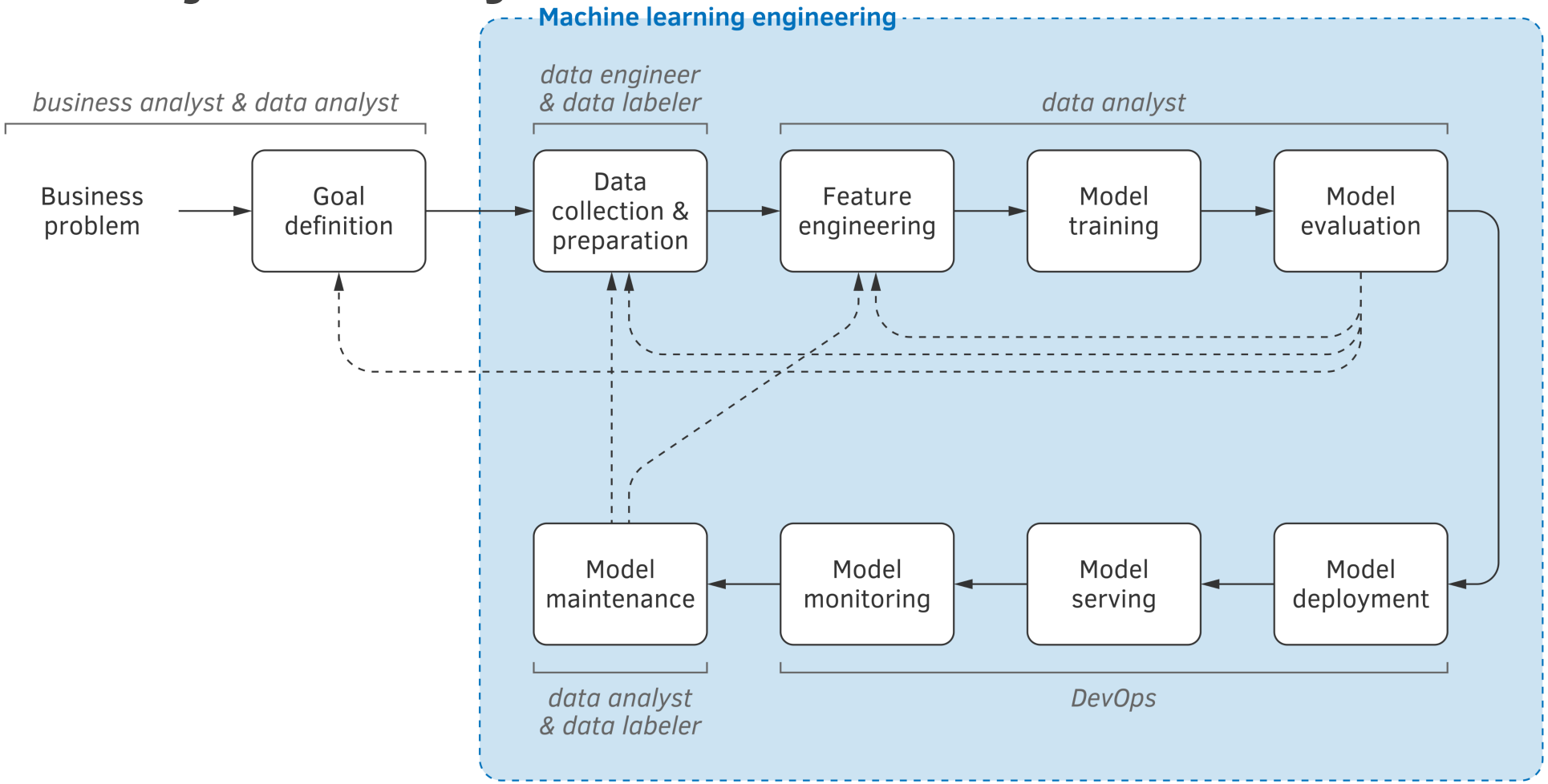


Figure 1: A schematic of a typical machine learning pipeline.

https://developers.google.com/machine-learning/testing-debugging/pipeline/overview

ML Project Lifecycle



http://www.mlebook.com/wiki/doku.php

Cost of Machine Learning

- 3 factors highly influence cost of a machine learning project:
  - Difficulty of the problem
  - Cost of the data
  - Need for accuracy

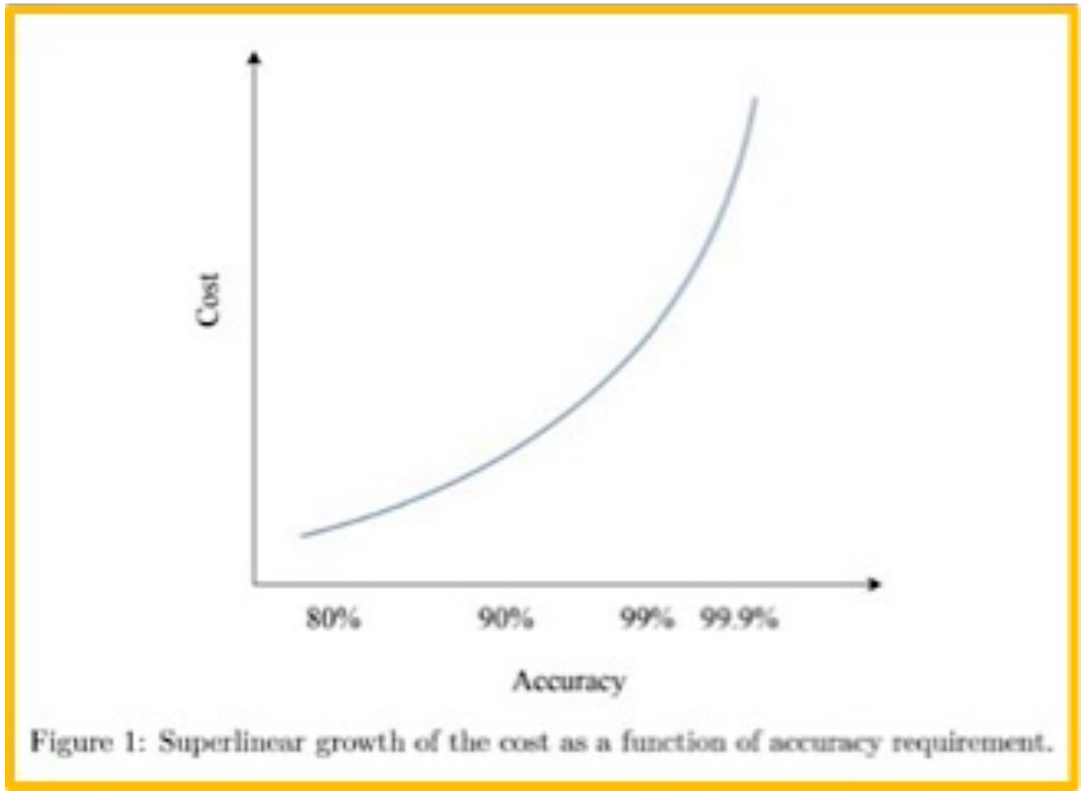


Figure 1: Superlinear growth of the cost as a function of accuracy requirement.

When to Use ML (or Not) According to Burkov

When to Use ML

- The problem is too complex for coding
- The problem is constantly changing
- The problem is perceptive (e.g. image, speech, video recognition)
- It's an unstudied phenomenon
- When there is a simple objective (e.g. yes/no)
- When it is cost-effective

When Not to Use ML (helpful hints)

- When system actions, decisions, or changes in behavior must be explainable
- System errors or failures are too costly
- Getting the right data is too hard or impossible
- Simpler methods or heuristics would work reasonably well
- You can manually fill an exhaustive lookup table (list expected output for any input)

http://www.mlebook.com/wiki/doku.php

When to Use ML (or Not) According to Burkov

When to Use ML

- The problem is too complex for coding
- The problem is constantly changing
- The problem is perceptive (e.g. image, speech, video recognition)
- It's an unstudied phenomenon
- When there is a simple objective (e.g. yes/no)
- When it is cost-effective

When Not to Use ML (helpful hints)

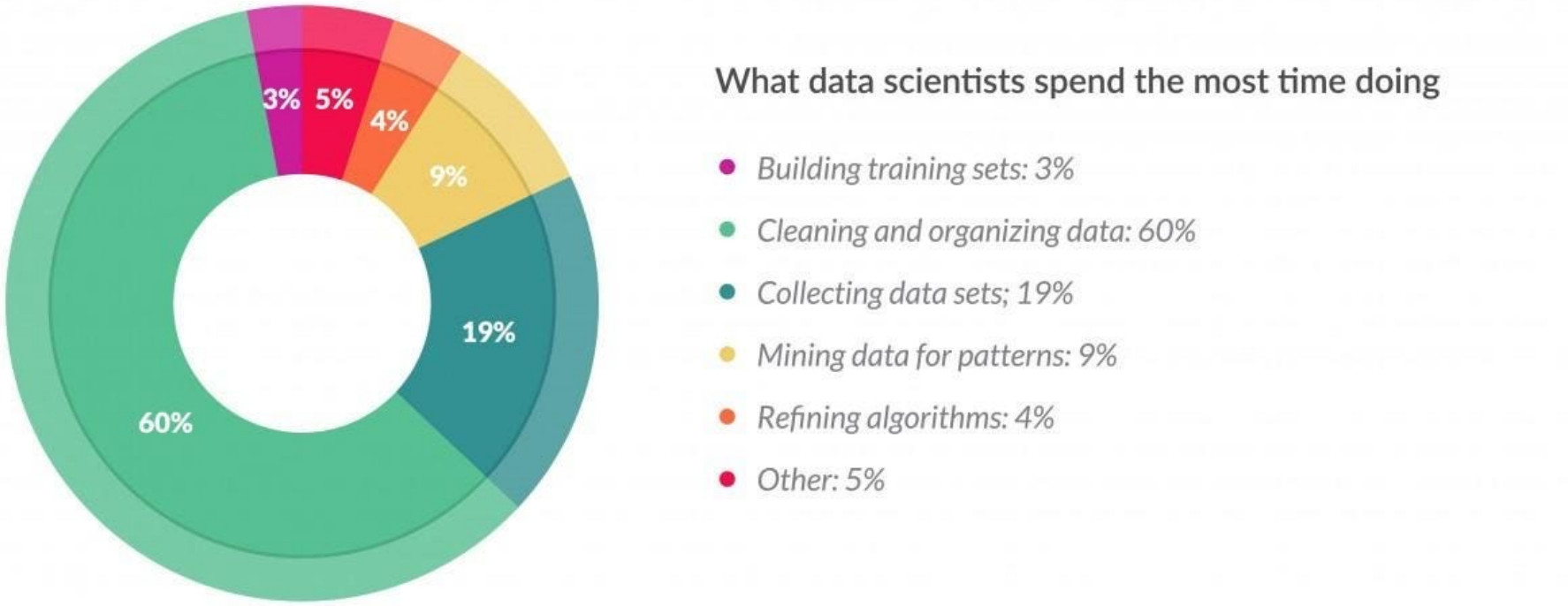
- When system actions, decisions, or changes in behavior must be explainable
- System errors or failures are too costly
- Getting the right data is too hard or impossible
- Simpler methods or heuristics would work reasonably well
- You can manually fill an exhaustive lookup table (list expected output for any input)

http://www.mlebook.com/wiki/doku.php

Steps to Solve a ML Problem

https://www.tensorflow.org/about
https://developers.google.com/machine-learning/guides/text-classification/

What Data Scientists Spend the Most Time Doing



https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/

THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

End-to-End Machine Learning Project

"The first question to ask your boss is what exactly the business objective is. Building a model is probably not the end goal."

-HOML

Professor Hamza F. Alsarhan
EMSE 6769

ML Project Steps

1. Look at the big picture.
2. Get the data.
3. Discover and visualize the data to gain insights.
4. Prepare the data for Machine Learning algorithms.
5. Select a model and train it.
6. Fine-tune your model.
7. Present your solution.
8. Launch, monitor, and maintain your system.

Hands-On Machine Learning, Ch2, p30

1. Look at the Big Picture

- "The first question to ask your boss is what exactly the business objective is. Building a model is probably not the end goal".
  - The objective will determine how you frame the problem, which algorithms you will select, which performance measure you will use to evaluate your model, and how much effort you will spend tweaking it.
- What does the current solution look like (if any)?
  - The current situation will often give you a reference for performance, as well as insights on how to solve the problem.
- Start Designing Your System
  - Is it supervised, unsupervised, or Reinforcement Learning?
  - Is it a classification task, a regression task, or something else?
  - Should you use batch learning or online learning techniques?

Frame the Problem and Look at the Big Picture

1. Define the objective in business terms.
2. How will your solution be used?
3. What are the current solutions/workarounds (if any)?
4. How should you frame this problem (supervised/unsupervised, online/offline, etc.)?
5. How should performance be measured?
6. Is the performance measure aligned with the business objective?
7. What would be the minimum performance needed to reach the business objective?
8. What are comparable problems? Can you reuse experiences or tools?
9. Is human expertise available?
10. How would you solve the problem manually?
11. List the assumptions you (or others) have made so far.
12. Verify assumptions if possible.

Hands-On Machine Learning, Appendix B

2. Get the Data

1. List the data you need and how much you need.
2. Find and document where you can get that data.
3. Check how much space it will take.
4. Check legal obligations, and get authorization if necessary.
5. Get access authorizations.
6. Create a workspace (with enough storage space).
7. Get the data.
8. Convert the data to a format you can easily manipulate (without changing the data itself).
9. Ensure sensitive information is deleted or protected (e.g., anonymized).
10. Check the size and type of data (time series, sample, geographical, etc.).
11. Sample a test set, put it aside, and never look at it (no data snooping!).

Note: automate as much as possible, so you can easily get fresh data.

Hands-On Machine Learning, Appendix B

3. Discover and Visualize the Data to Gain Insights

1. Create a copy of the data for exploration (sampling it down to a manageable size if necessary).
2. Create a Jupyter notebook to keep a record of your data exploration.
3. Study each attribute and its characteristics:
    1. Name
    2. Type (categorical, int/float, bounded/unbounded, text, structured, etc.)
    3. % of missing values
    4. Noisiness and type of noise (stochastic, outliers, rounding errors, etc.)
    5. Usefulness for the task
    6. Type of distribution (Gaussian, uniform, logarithmic, etc.)
4. For supervised learning tasks, identify the target attribute(s).
5. Visualize the data.
6. Study the correlations between attributes.
7. Study how you would solve the problem manually.
8. Identify the promising transformations you may want to apply.
9. Identify extra data that would be useful (go back to “Get the Data”).
10. Document what you have learned.

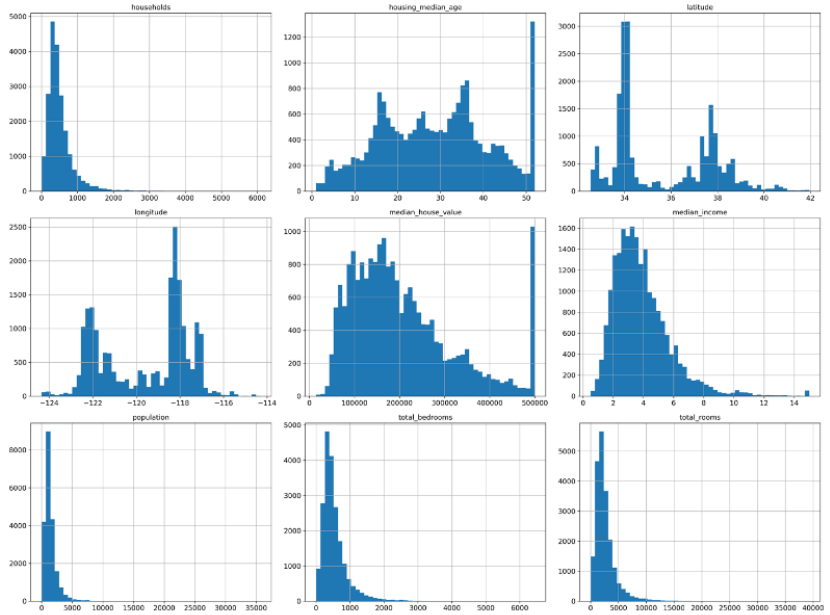


Figure 2-8. A histogram for each numerical attribute

Hands-On Machine Learning, Appendix B

4. Prepare the Data

1. Data cleaning:
    1. Fix or remove outliers (optional).
    2. Fill in missing values (e.g., with zero, mean, median…) or drop their rows (or columns).
2. Feature selection (optional):
    1. Drop the attributes that provide no useful information for the task.
3. Feature engineering, where appropriate:
    1. Discretize continuous features – by creating bins of ranges.
    2. Decompose features (e.g., categorical, date/time – into hours, days, and months, etc.).
    3. Add promising transformations of features (e.g., log(x), sqrt(x), x2, etc.).
    4. Aggregate features into promising new features.
4. Feature scaling:
    1. Standardize or normalize features.

5. Select a Model and Train it

1. Train many quick-and-dirty models from different categories (e.g., linear, naive Bayes, SVM, Random Forest, neural network, etc.) using standard parameters.
2. Measure and compare their performance.
    1. For each model, use N-fold cross-validation and compute the mean and standard deviation of the performance measure on the N folds.
3. Analyze the most significant variables for each algorithm.
4. Analyze the types of errors the models make.
    1. What data would a human have used to avoid these errors?
5. Perform a quick round of feature selection and engineering.
6. Perform one or two more quick iterations of the five previous steps.
7. Shortlist the top three to five most promising models, preferring models that make different types of errors.

Hands-On Machine Learning, Appendix B

6. Fine-Tune your Model

1. Fine-tune the hyperparameters using cross-validation:
    1. Treat your data transformation choices as hyperparameters, especially when you are not sure about them (e.g., if you are not sure whether to replace missing values with zeros or with the median value, or to just drop the rows).
    2. Unless there are very few hyperparameter values to explore or you have prior knowledge of the optimal ranges for a hyperparameter, use random search over grid search.
        - If training is very long, you may prefer a Bayesian optimization approach – exploring unknown regions while exploiting promising regions (e.g., using Gaussian process priors, as described by Jasper Snoek et al.)
2. Try Ensemble methods. Combining your best models will often produce better performance than running them individually.
3. Once you are confident about your final model, measure its performance on the test set to estimate the generalization error.

Hands-On Machine Learning, Appendix B

Jasper Snoek et al., "Practical Bayesian Optimization of Machine Learning Algorithms," *Proceedings of the 25th International Conference on Neural Information Processing Systems* 2 (2012): 2951–2959.

7. Present Your Solution

1. Document what you have done.
2. Create a nice presentation.
    1. Make sure you highlight the big picture first.
3. Explain why your solution achieves the business objective.
4. Don’t forget to present interesting points you noticed along the way.
    1. Describe what worked and what did not.
    2. List your assumptions and your system’s limitations.
5. Ensure your key findings are communicated through beautiful visualizations or easy-to-remember statements (e.g., “the median income is the number-one predictor of housing prices”).

8. Launch, Monitor, and Maintain Your System

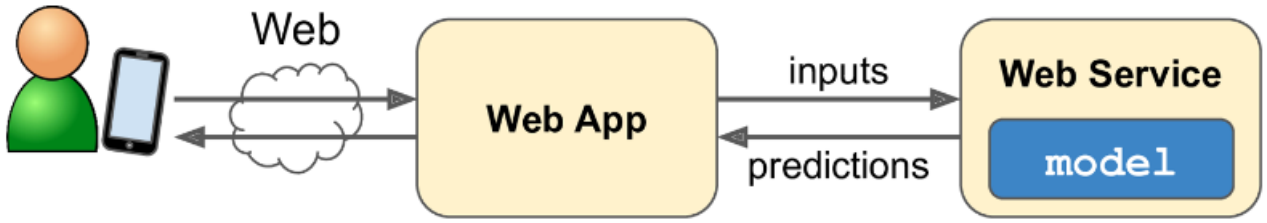


Figure 2-17. A model deployed as a web service and used by a web application

1. Get your solution ready for production (plug into production data inputs, write unit tests, etc.).
2. Write monitoring code to check your system's live performance at regular intervals and trigger alerts when it drops.
    1. Beware of slow degradation: models tend to "rot" as data evolves.
    2. Measuring performance may require a human pipeline (e.g., via a crowdsourcing service).
    3. Also monitor your inputs' quality (e.g., a malfunctioning sensor sending random values, or another team's output becoming stale). This is particularly important for online learning systems.
3. Retrain your models on a regular basis on fresh data (automate as much as possible).

THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

Classification & Machine Learning Metrics

Professor Hamza F. Alsarhan
EMSE 6769

MNIST

- Set of 70,000 small images of digits handwritten by high school students and employees of the US Census Bureau.
- Each image is 28 × 28 pixels
- Each feature simply represents one pixel's intensity, from 0 (white) to 255 (black) (i.e. 784 features)
- Each image is labeled with the digit it represents.
- This set has been studied so much that it is often called the "hello world" of Machine Learning: whenever people come up with a new classification algorithm, they are curious to see how it will perform on MNIST

Hands-On Machine Learning, Ch3

MNIST – Already Processed

- The MNIST dataset is actually already split into a training set (the first 60,000 images) and a test set (the last 10,000 images).

- The training set is already shuffled for us, which is good because this guarantees that all cross-validation folds will be similar (you do not want one fold to be missing some digits).

Cautionary Example – Just look, 93% accuracy!

- Above 93% accuracy (ratio of correct predictions) on all cross-validation folds? This looks amazing, doesn't it? Well, before you get too excited, let's look at a very basic classifier that just classifies every single image in the "not-5" class

- That's right, it has over 90% accuracy! This is simply because only about 10% of the images are 5s, so if you always guess that an image is not a 5, you will be right about 90% of the time.

- This demonstrates why accuracy is generally not the preferred performance measure for classifiers, especially when you are dealing with skewed/imbalanced datasets (i.e., when some classes are much more frequent than others).

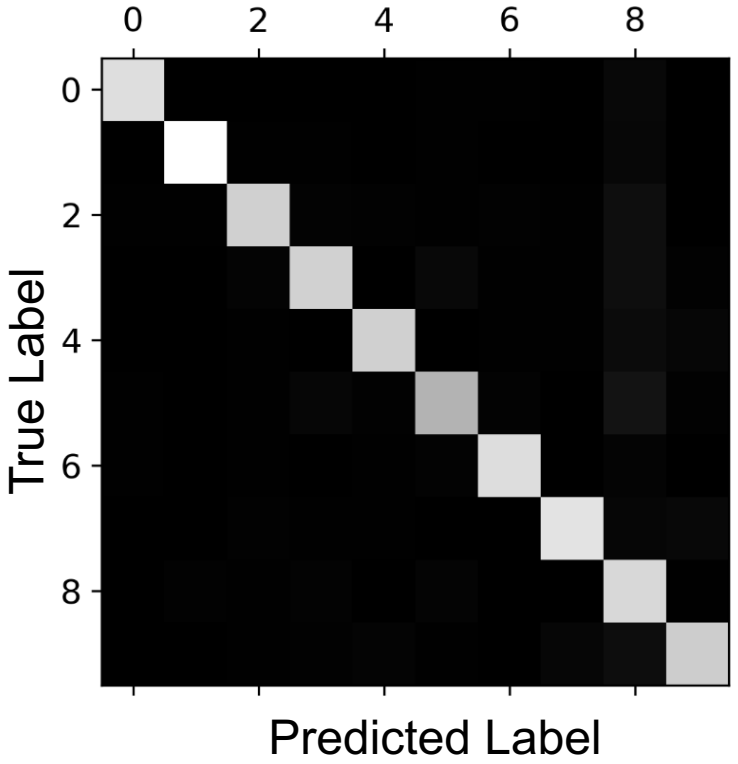
Multiclass Classification

- Whereas binary classifiers distinguish between two classes, multiclass classifiers (also called multinomial classifiers) can distinguish between more than two classes.

- Examples of multi-class classifiers:
  - Intrusion Detection System (IDS) that can determine the type of attack detected
  - Vehicle type recognition – car, truck, motorcycle, etc.

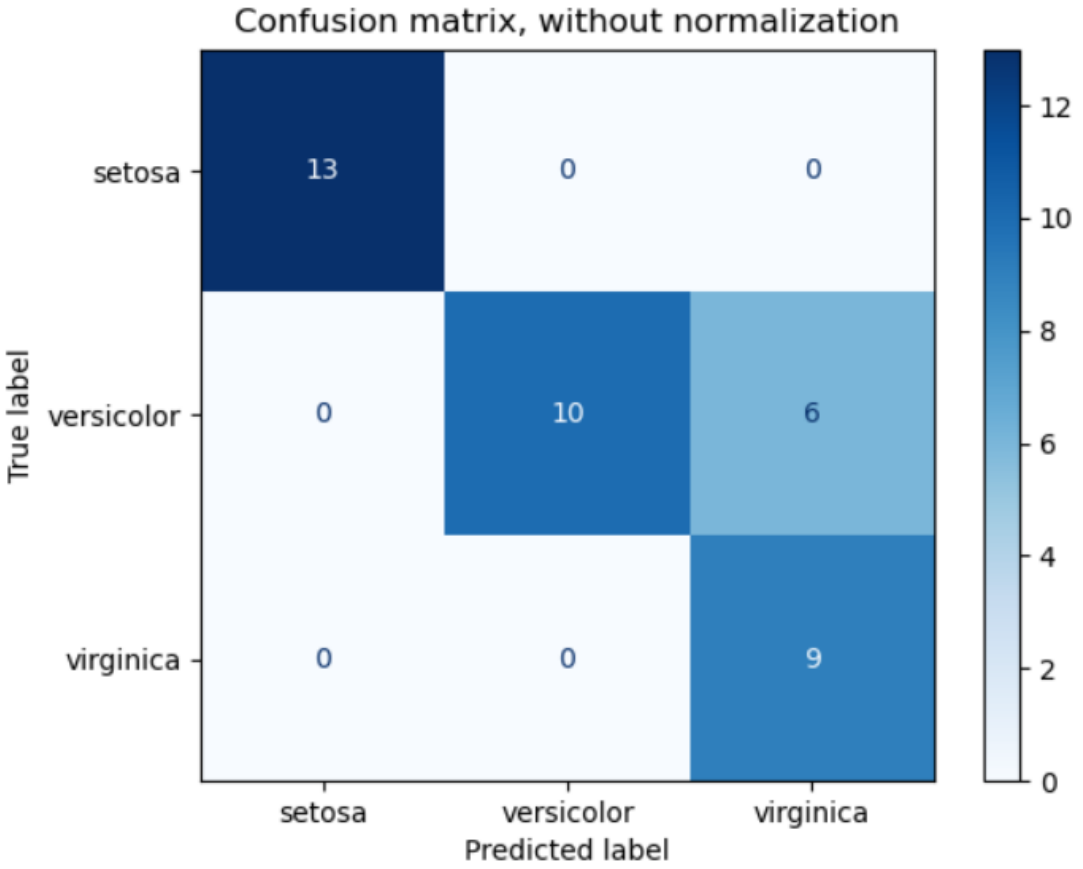
Confusion Matrix

- Better way to evaluate the performance of a classifier.
- Count the number of times instances of class A are classified as class B.
  - For example, to know the number of times the classifier confused images of 5s with 3s, you would look in the row with the 5 label and the column with the 3 label of the confusion matrix.
- Each row in a confusion matrix represents an actual class, while each column represents a predicted class.
- A perfect classifier would have only true positives and true negatives, so its confusion matrix would have nonzero values only on its main diagonal (top left to bottom right).



Confusion Matrix and Common Outputs

- A confusion matrix is used to evaluate the performance of a classifier
- Diagonals are when the predicted label is equal to the true label
- Off-diagonal elements are those that are mislabeled by the classifier

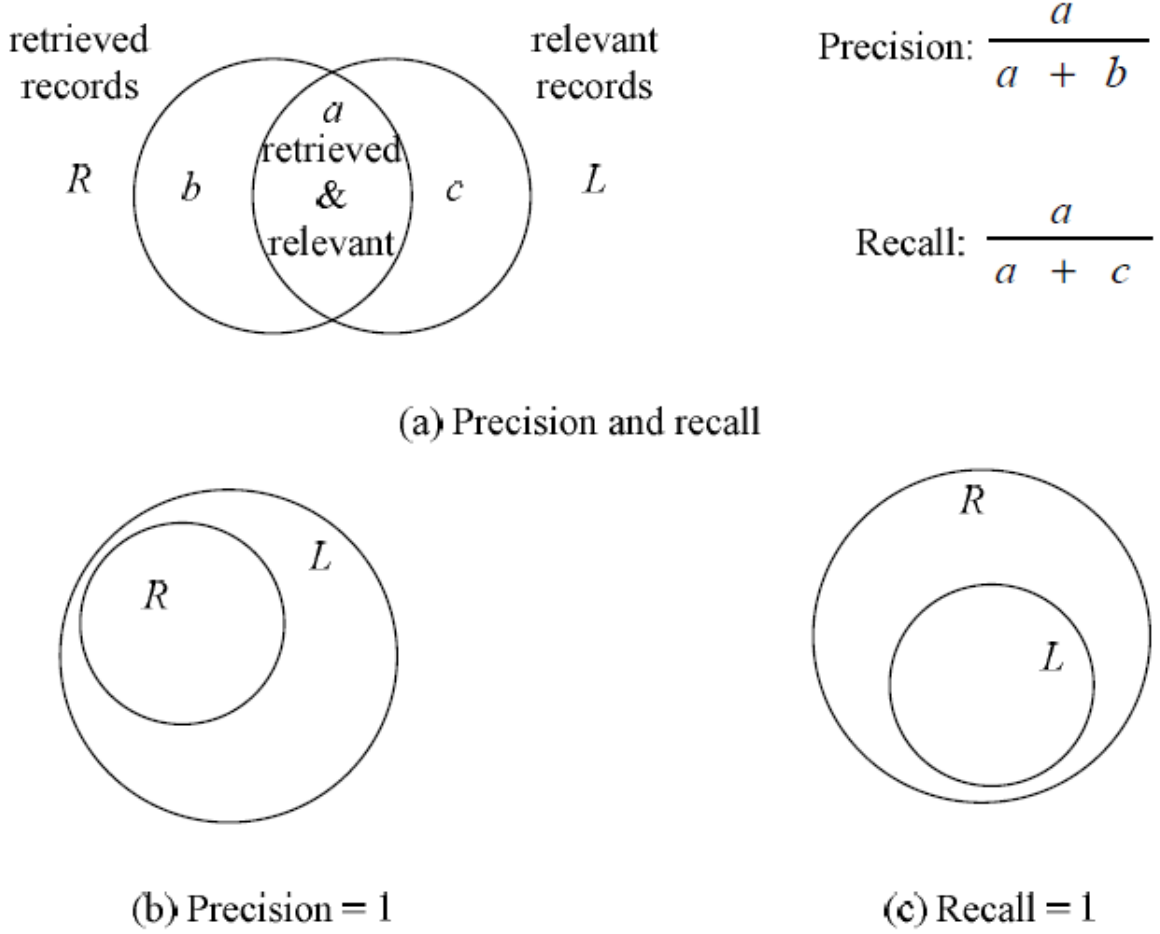


https://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html#sphx-glr-auto-examples-model-selection-plot-confusion-matrix-py

Precision and Recall

- **Precision**: ratio of true positives divided by the total number of positive predictions

- **Recall**: ratio of true positives divided by the total number of positive examples



Precision

$$\text{precision} = \frac{TP}{TP + FP}$$

- Accuracy of the positive predictions; this is called the precision of the classifier
- TP is the number of true positives, and FP (Type I error) is the number of false positives

Recall

$$\text{recall} = \frac{TP}{TP + FN}$$

- Precision is typically used along with another metric named recall, also called sensitivity or the true positive rate (TPR): this is the ratio of positive instances that are correctly detected by the classifier
- FN (Type II error) is the number of false negatives

Hands-On Machine Learning, Ch3

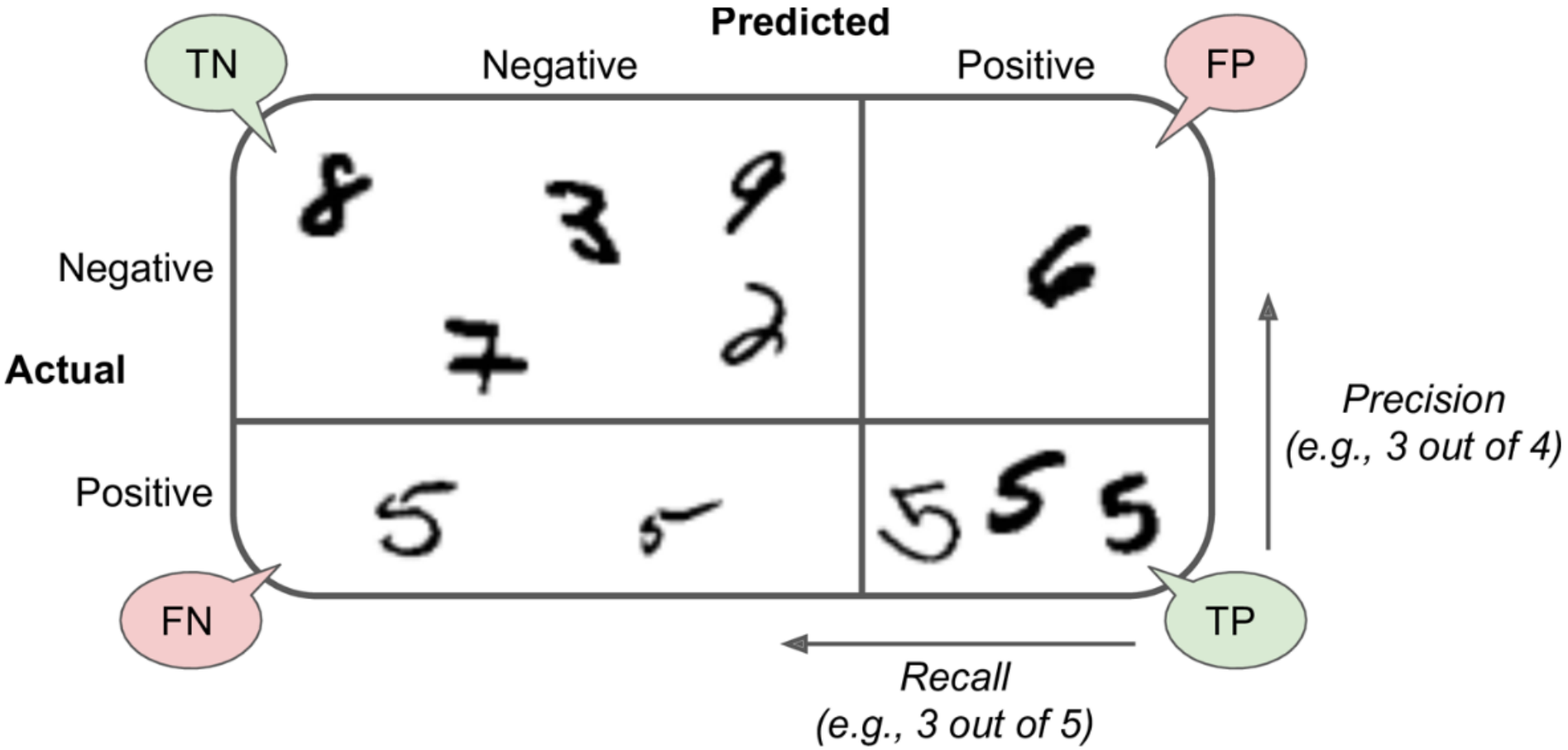


Figure 3-2. An illustrated confusion matrix shows examples of true negatives (top left), false positives (top right), false negatives (lower left), and true positives (lower right)

Hands-On Machine Learning, Ch3

Precision/Recall Trade-off

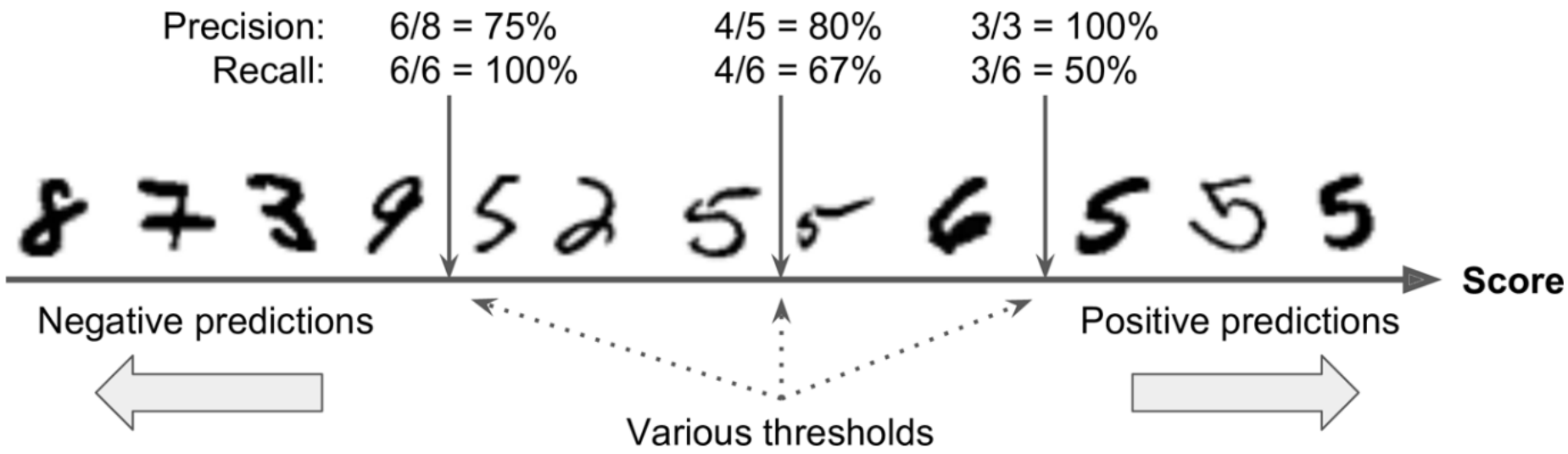


Figure 3-3. In this precision/recall trade-off, images are ranked by their classifier score, and those above the chosen decision threshold are considered positive; the higher the threshold, the lower the recall, but (in general) the higher the precision

- Increasing precision reduces recall, and vice versa.
- Raising the threshold decreases recall.

Hands-On Machine Learning, Ch3

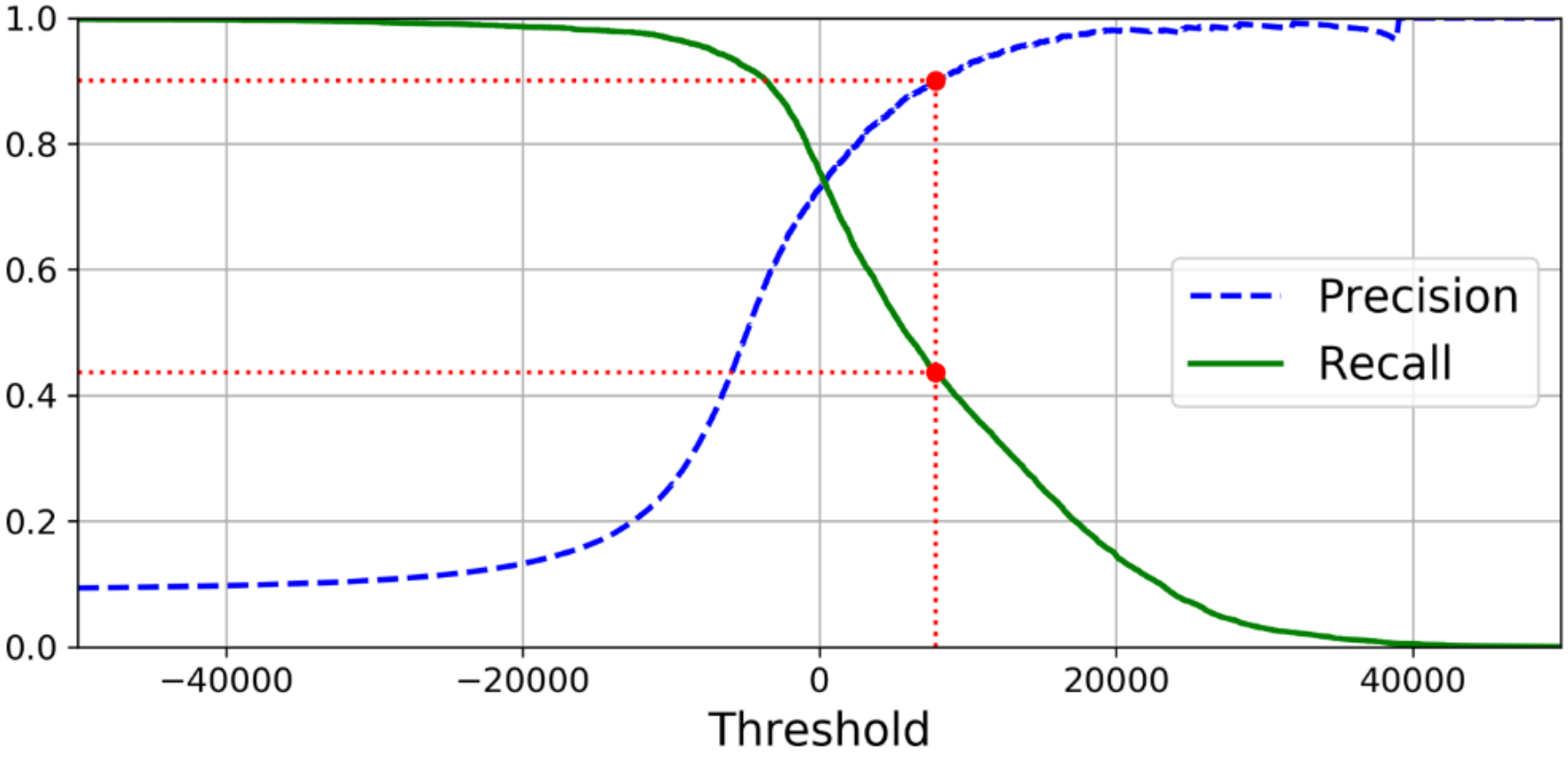


Figure 3-4. Precision and recall versus the decision threshold

Performance Measures

	Predicted class	
True Class	Yes	No
Yes	TP: True Positive	FN: False Negative
No	FP: False Positive	TN: True Negative

- Error rate = # of errors / # of instances = (FN+FP) / N
- Recall = # of found positives / # of positives
- = TP / (TP+FN) = sensitivity = hit rate
- Precision = # of found positives / # of found
- = TP / (TP+FP)
- Specificity = TN / (TN+FP)
- False alarm rate = FP / (FP+TN) = 1 – Specificity
- $$\text{accuracy} \stackrel{\text{def}}{=} \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

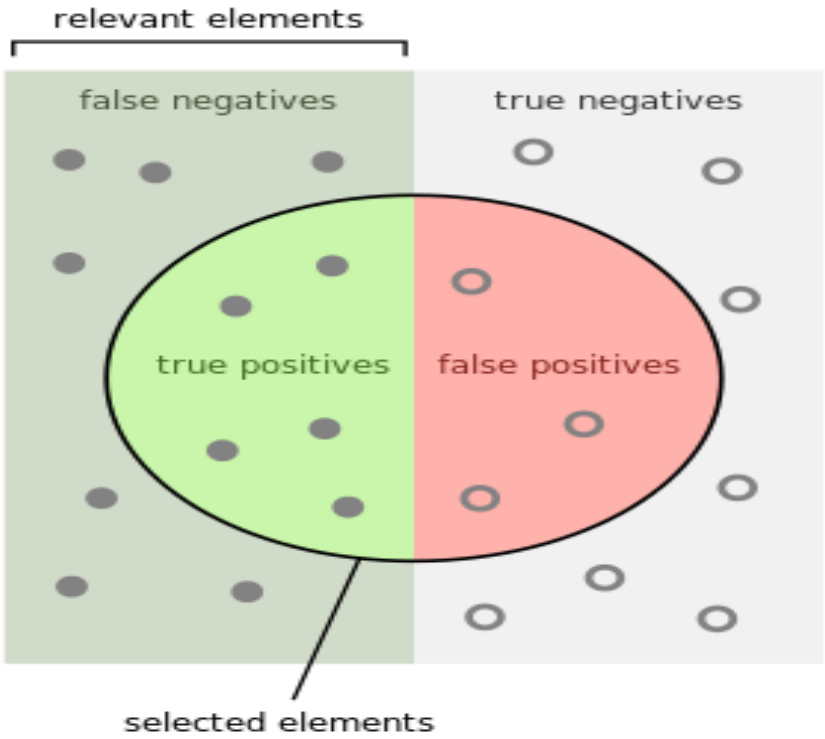
Accuracy, Sensitivity, Specificity

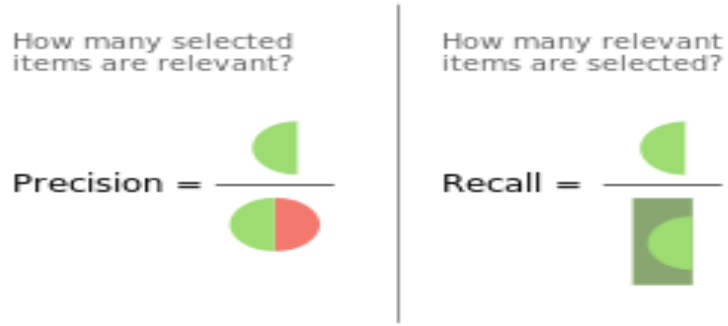
Accuracy: total number of correctly classified examples divided by the total number of classified examples

Name	Formula
error	$(fp + fn)/N$
accuracy	$(tp + tn)/N = 1 - \text{error}$
tp-rate	tp/p
fp-rate	fp/n
sensitivity	$tp/p = \text{tp-rate}$
specificity	$tn/n = 1 - \text{fp-rate}$

$$\text{TPR} \stackrel{\text{def}}{=} \frac{\text{TP}}{\text{TP} + \text{FN}} \quad \text{and} \quad \text{FPR} \stackrel{\text{def}}{=} \frac{\text{FP}}{\text{FP} + \text{TN}}$$

MLE, Chapter 5





MLE, Chapter 5

F-Measure (F1 Score)

- Percentage of retrieved documents that are relevant: **precision**=TP/(TP+FP)
- Percentage of relevant documents that are retrieved: **recall** =TP/(TP+FN)
- F-measure = (2 × recall × precision)/(recall + precision)
- Sometimes sensitivity × specificity is used as a measure:
  - Sensitivity x Specificity = TP-rate x (1 – FP-rate) = (TP / (TP + FN)) × (TN / (FP + TN))

F1 Score

$$F_1 = \frac{2}{\frac{1}{\text{precision}} + \frac{1}{\text{recall}}} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} = \frac{TP}{TP + \frac{FN+FP}{2}}$$

- Simple way to compare two classifiers
- Combines precision and recall into a single metric
- The classifier will only get a high F1 score if both recall and precision are high
- The F1 score favors classifiers that have similar precision and recall

The ROC Curve

- The receiver operating characteristic (ROC) curve is another common tool used with binary classifiers.
- Very similar to the precision/recall curve however the ROC curve plots the true positive rate (another name for recall) against the false positive rate (FPR).
- The FPR is the ratio of negative instances that are incorrectly classified as positive.
- The TNR is also called specificity. Hence, the ROC curve plots sensitivity (recall) versus 1 – specificity.
- Can compare classifiers by measuring the area under the curve (AUC).
  - A perfect classifier will have a ROC AUC equal to 1,
  - A purely random classifier will have a ROC AUC equal to 0.5.

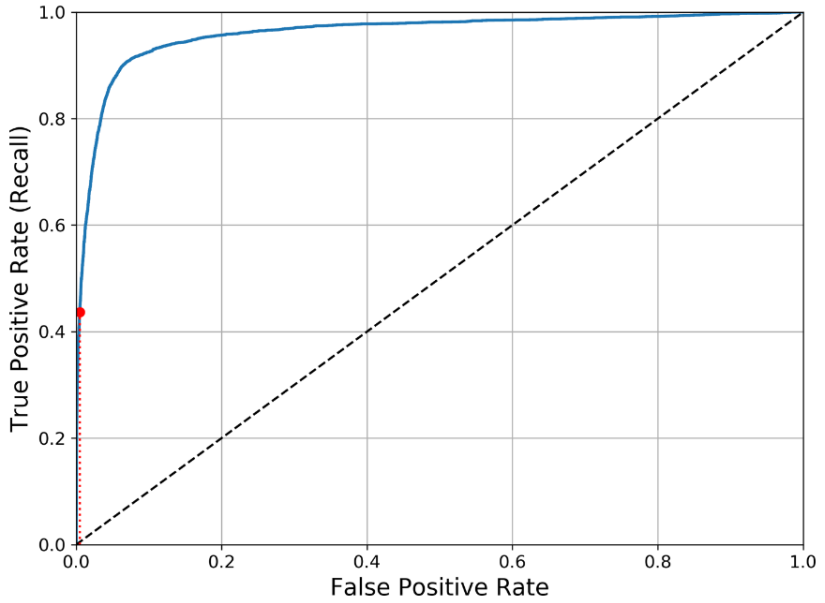
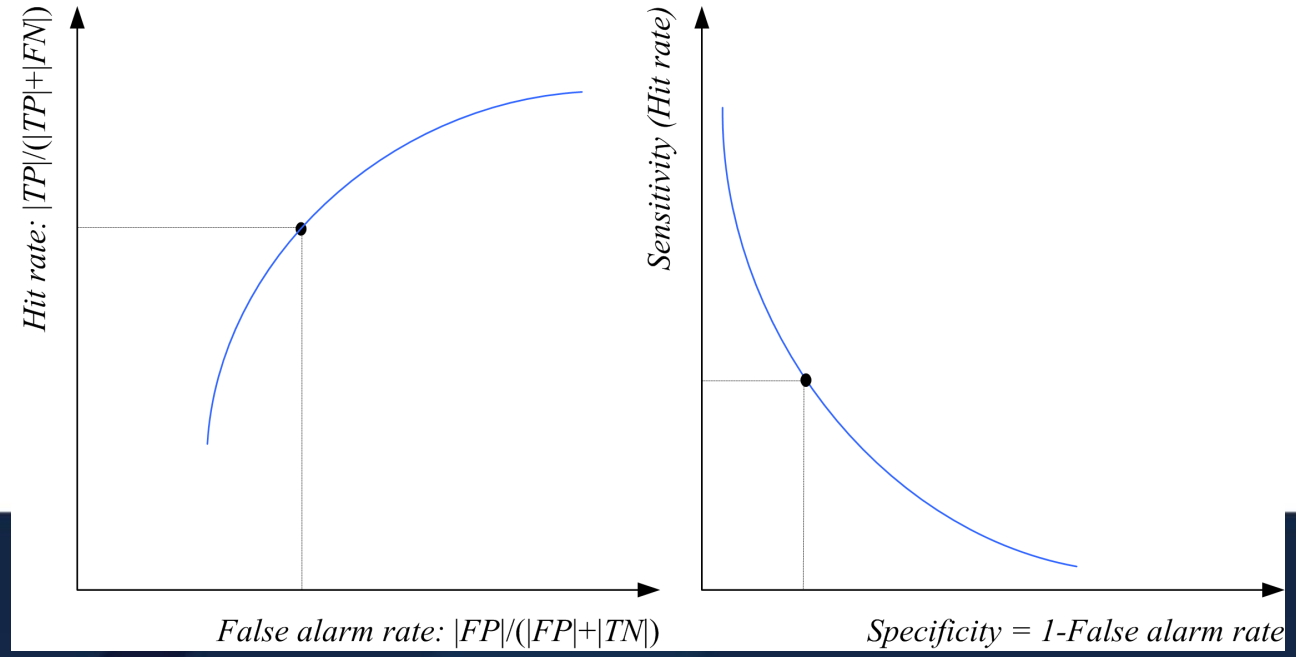


Figure 3-6. This ROC curve plots the false positive rate against the true positive rate for all possible thresholds; the red circle highlights the chosen ratio (at 43.68% recall)

ROC Curve

- Area under the ROC curve (AUC):
  Is the probability that a randomly chosen positive instance is ranked above a randomly chosen negative one



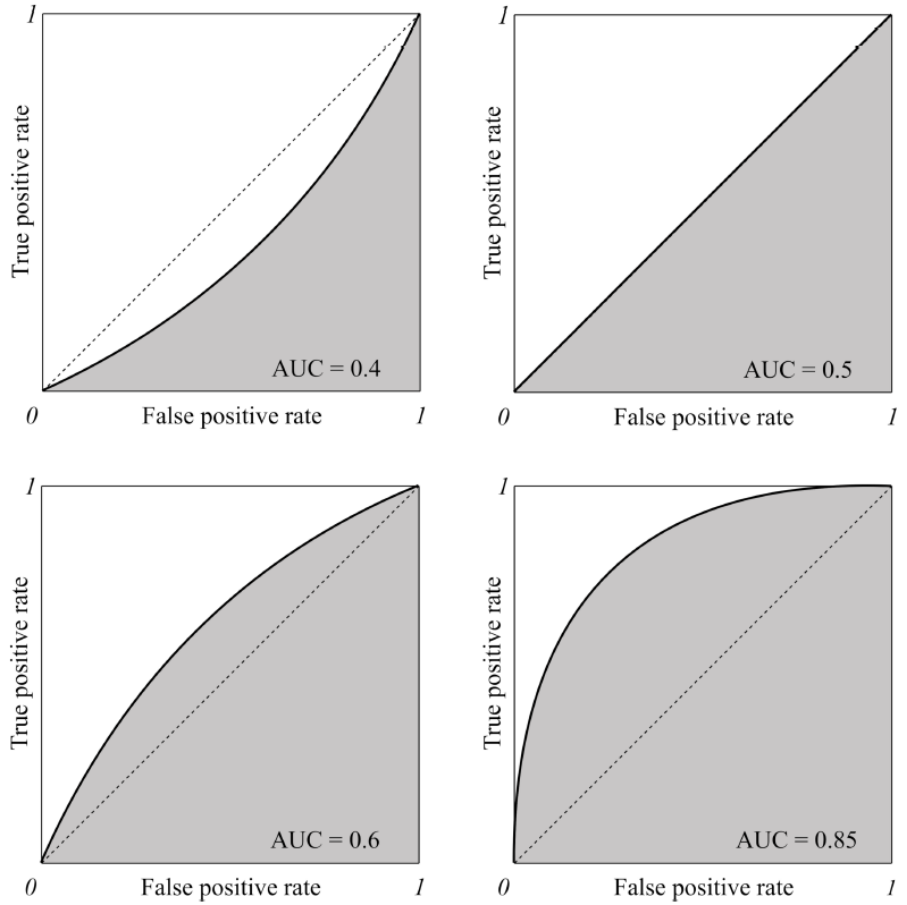
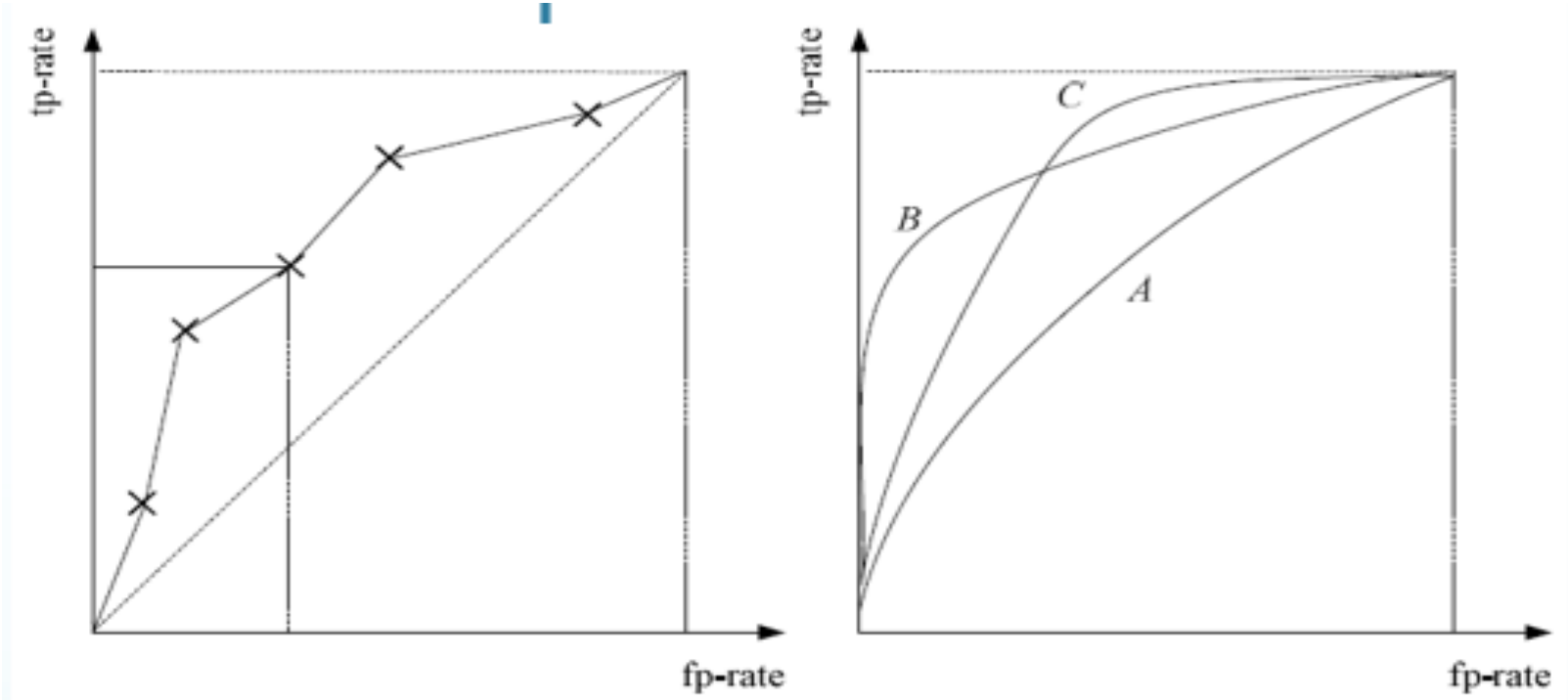


Figure 5: The area under the ROC curve (shown in grey).

MLE, p145

Example ROC Curves



(a) Typical ROC curve. Each classifier has a threshold that allows us to move over this curve, and we decide on a point, based on the relative importance of hits versus false alarms, namely, true positives and false positives. The area below the ROC curve is called AUC.

(b) A classifier is preferred if its ROC curve is closer to the upper-left corner (larger AUC). B and C are preferred over A; B and C are preferred under different loss matrices.

- Given two ROC curves with the same AUC:
  - You would prioritize a model with higher sensitivity (TPR) even is that leads to a higher FPR in a medical diagnostics scenario.
  - You would prioritize a slightly lower TPR with a lower FPR in scenarios where the cost of a FP prediction is high.

Bringing the Pieces Together

- You now know how to:
  - Choose the appropriate metrics for your task
  - Evaluate your classifiers using cross-validation
  - Select the precision/recall trade-off that fits your needs
  - Use ROC curves and ROC AUC scores to compare various models

THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

Getting Started with Google Colab

Professor Hamza F. Alsarhan
EMSE 6769

Google’s Impact on AI and Machine Learning

- **Leader in AI Research:** Google has emerged as a leader and innovator in AI research
- **TensorFlow AI Framework:** The creation of TensorFlow, an advanced AI framework, showcases Google's commitment to pushing the boundaries of machine learning.
- **Open-Sourcing TensorFlow:** TensorFlow has been open-sourced, allowing broader access and collaboration within the AI community.
- **Colaboratory Development Tool:** Colaboratory, a development tool by Google, complements TensorFlow and facilitates collaborative work.
- **Google Colab:** Formerly known as Colaboratory, Google Colab is now freely accessible to the public since 2017.
- **Free GPU Support in Colab:** Google's enticing offer includes free GPU support in Colab, enhancing the capabilities for developers.
- **Academic Standardization:** The public accessibility of Colab may contribute to its adoption as a standard tool in academic settings for teaching machine learning and data science.
- **Streamlined Learning Processes:** The introduction of Colab has significantly streamlined the learning and development processes for machine learning applications

Benefits of Google Colab

As a programmer, you can perform the following using Google Colab:

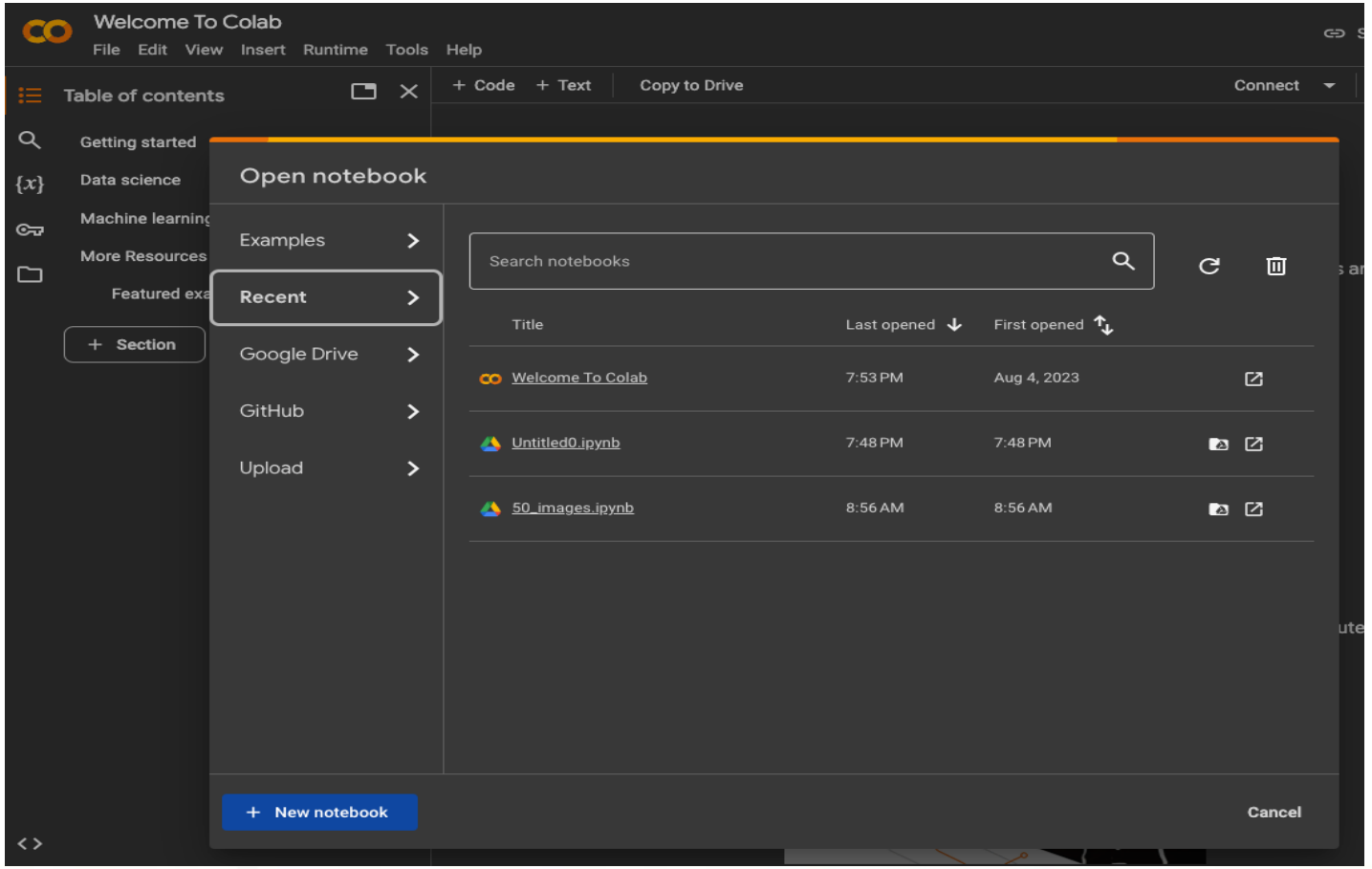
- Write and execute code in Python
- Document your code that supports mathematical equations
- Create/Upload/Share notebooks
- Import/Save notebooks from/to GoogleDrive
- Import/Publish notebooks from GitHub
- Import external datasets e.g. from Kaggle
- Integrate PyTorch, TensorFlow, Keras, OpenCV
- Free Cloud service with free GPU

Colab allows you to code within your browser

- Introduction to Colab https://www.youtube.com/watch?v=inN8seMm7UI

Google Colab Tutorial

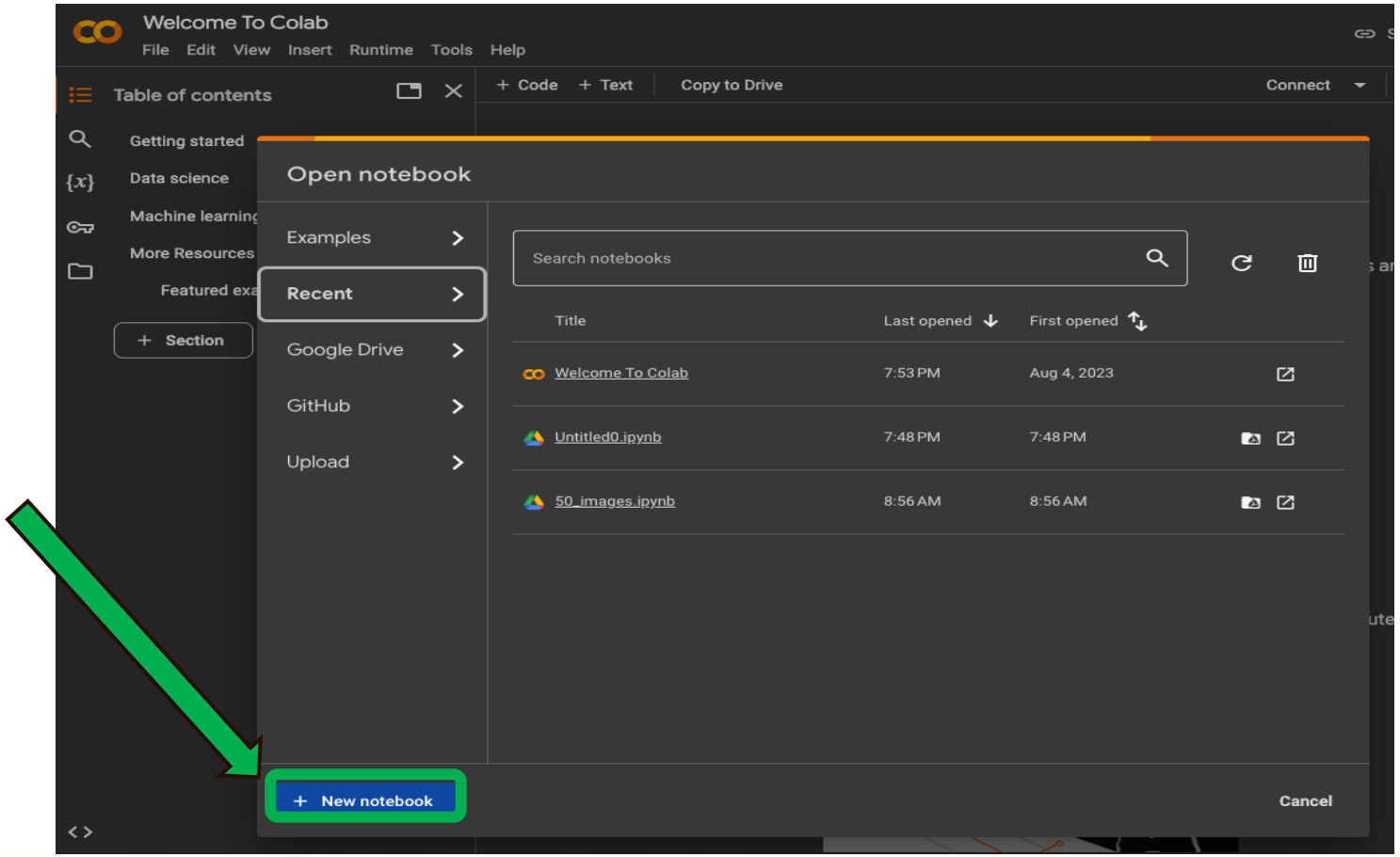
Open the following URL in your browser:
https://colab.research.google.com



https://www.tutorialspoint.com/google_colab/index.htm

Google Colab Tutorial

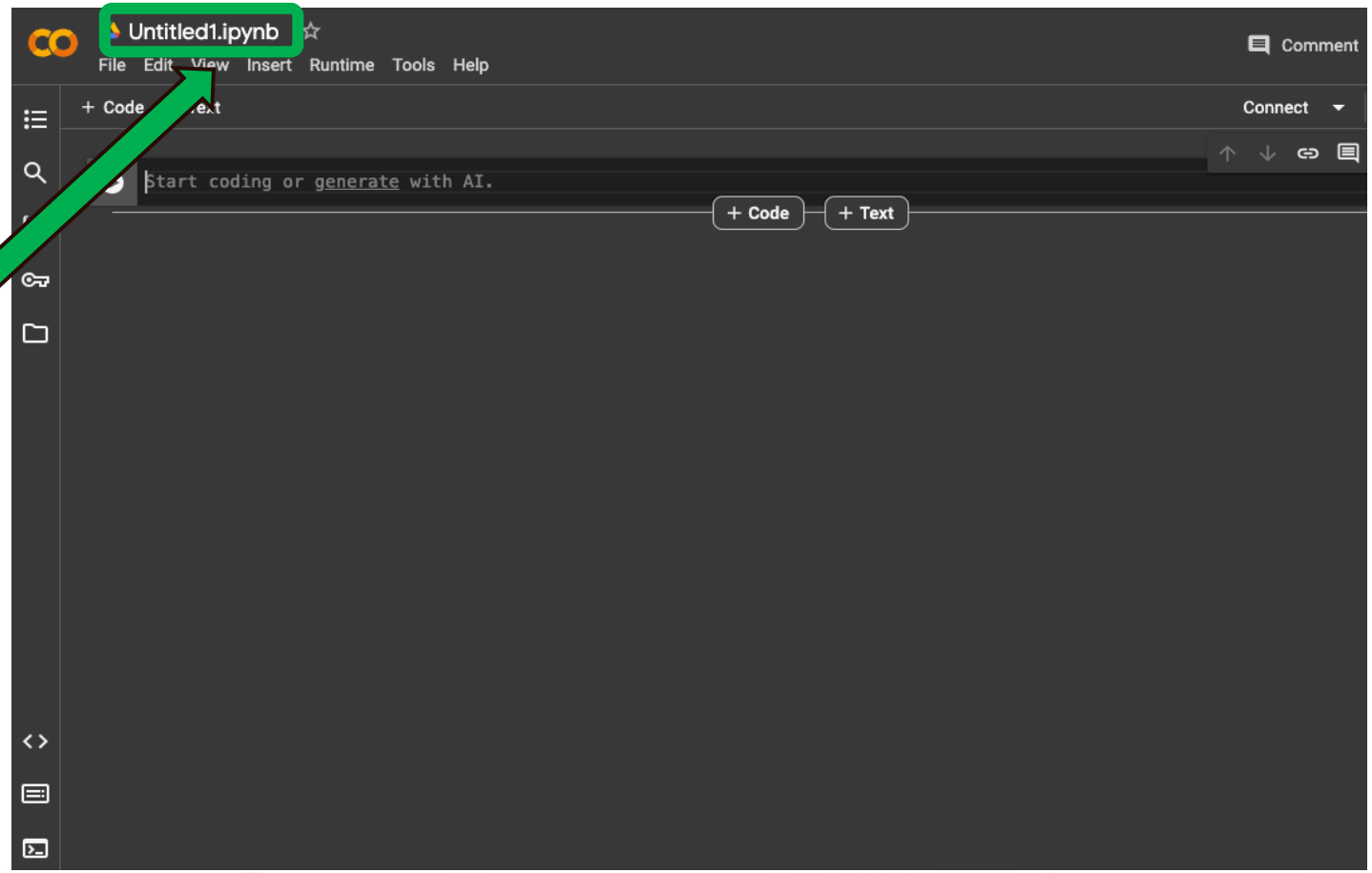
Click on the NEW NOTEBOOK link at the bottom of the screen. A new notebook will open as shown in the screen



https://www.tutorialspoint.com/google_colab/index.htm

Google Colab Tutorial

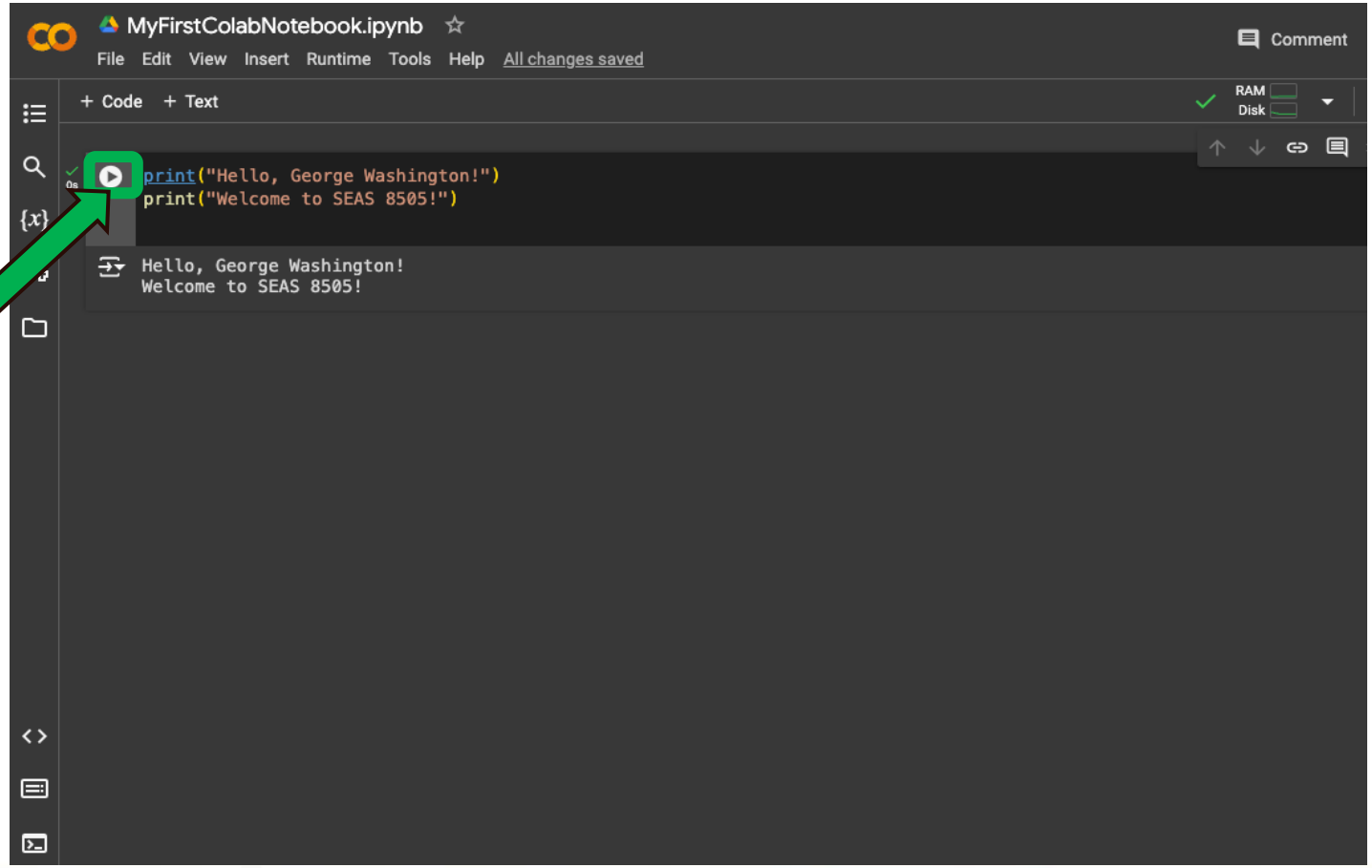
We will call this notebook MyFirstColabNotebook, so type in this name in the edit box and hit ENTER. The notebook will acquire the name that you have given now.



https://www.tutorialspoint.com/google_colab/index.htm

Google Colab Tutorial

Entering Code: You will now enter a simple piece of Python code in the code window and execute it.

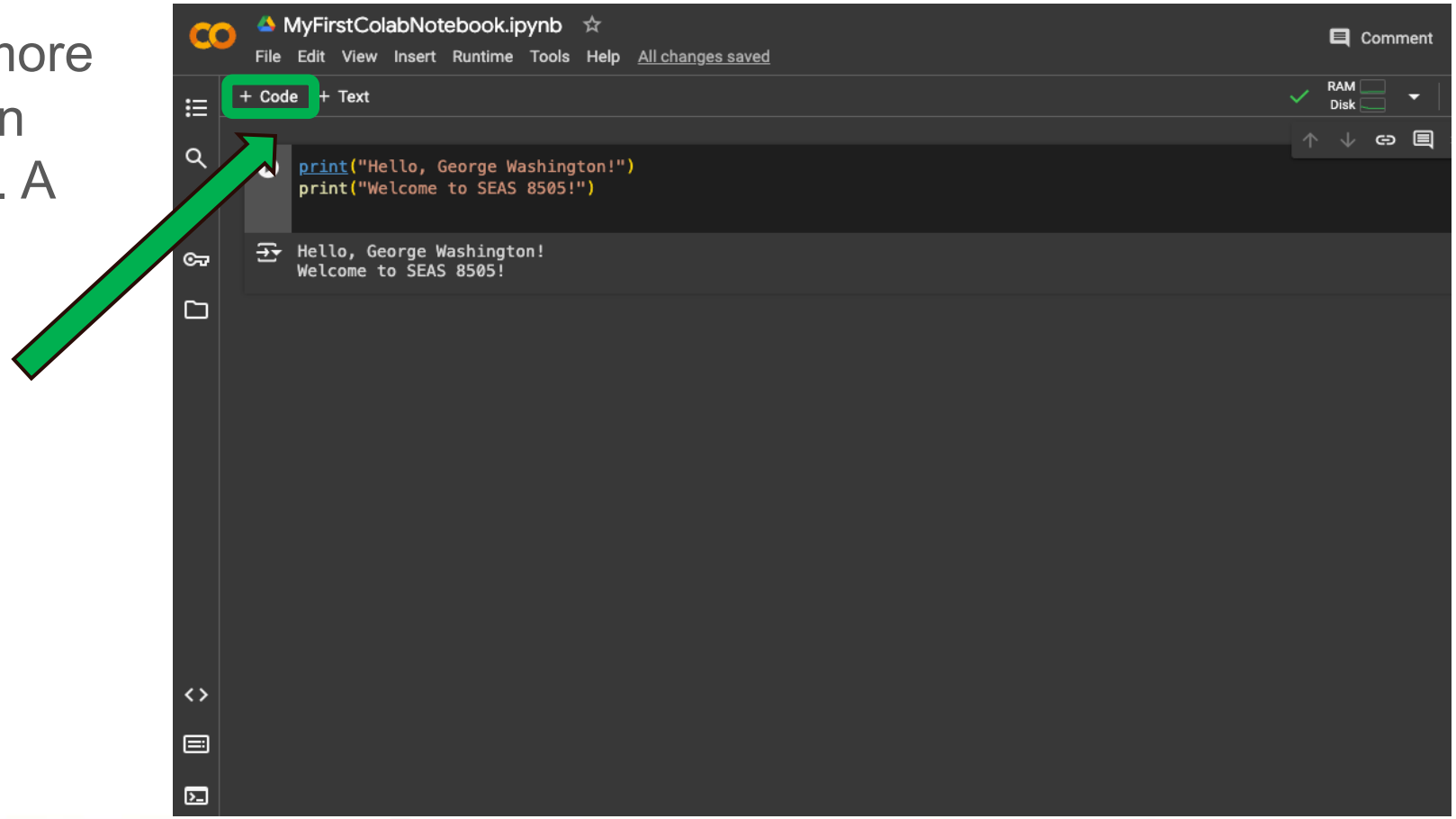


Executing the Code: To execute the code, click on the arrow on the left side of the code window. You should see the output underneath the code window.

https://www.tutorialspoint.com/google_colab/index.htm

Google Colab Tutorial

Adding Code Cells: To add more code to your notebook, click on the + CODE to add a new cell. A new code cell will be added underneath the current cell.

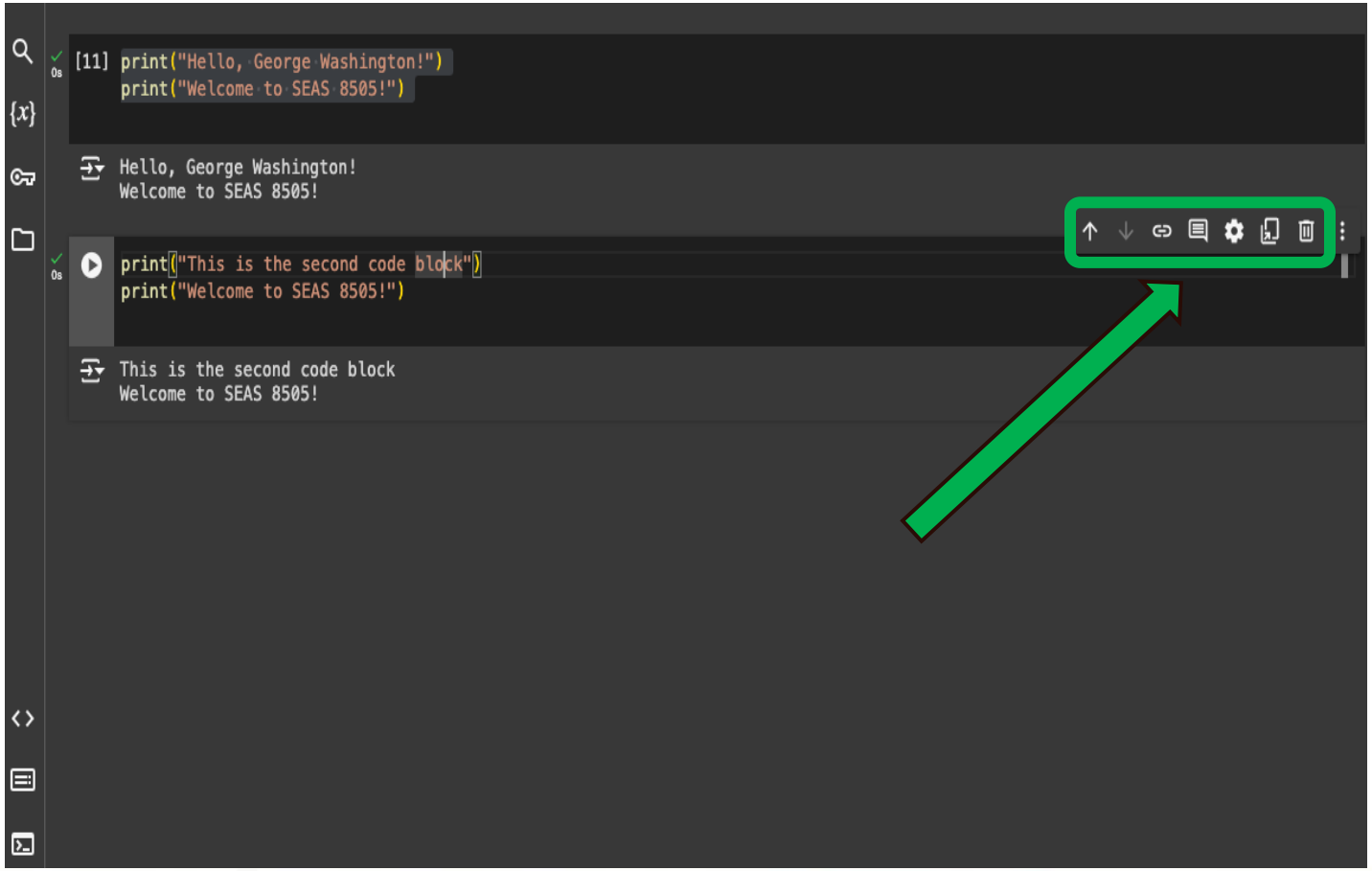


https://www.tutorialspoint.com/google_colab/index.htm

Google Colab Tutorial

Moving Code Blocks: If you click on a cell you will see a tool where you can click the **UP CELL** or **DOWN CELL** buttons.

Deleting Code Blocks: You can remove such cells from your project easily with a single click. Click on the tool **delete icon** at the top right corner of your code cell.

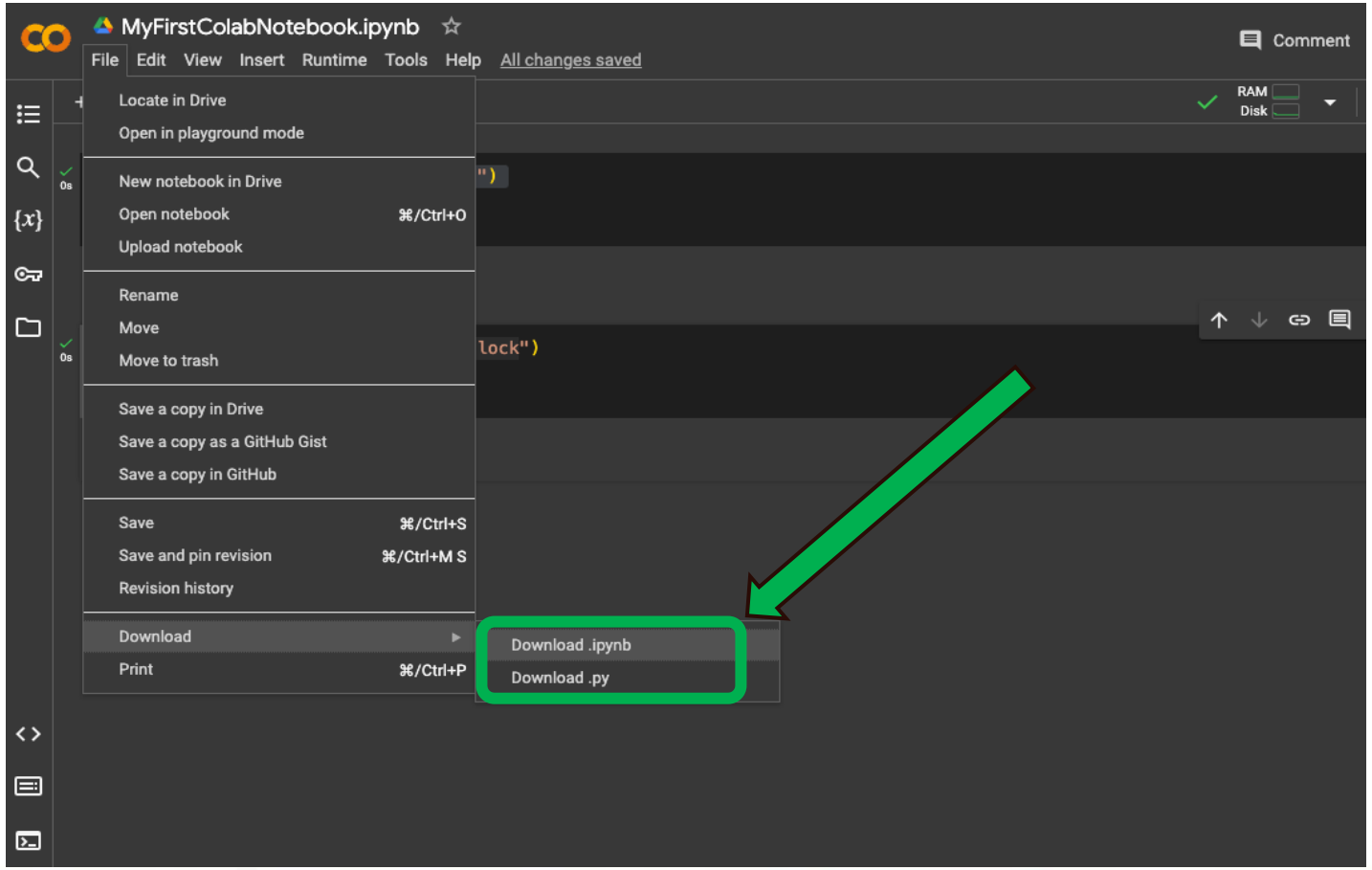


https://www.tutorialspoint.com/google_colab/index.htm

Google Colab Tutorial

Saving and Retrieving Your Work:

- There are many options for saving your work including Google Drive, GitHub, and to your local machine
- You can also share your Colab Notebook with others
- We will cover saving to and uploading from your local machine
  - From the “File” tab, you can download or upload a notebook from your local drive



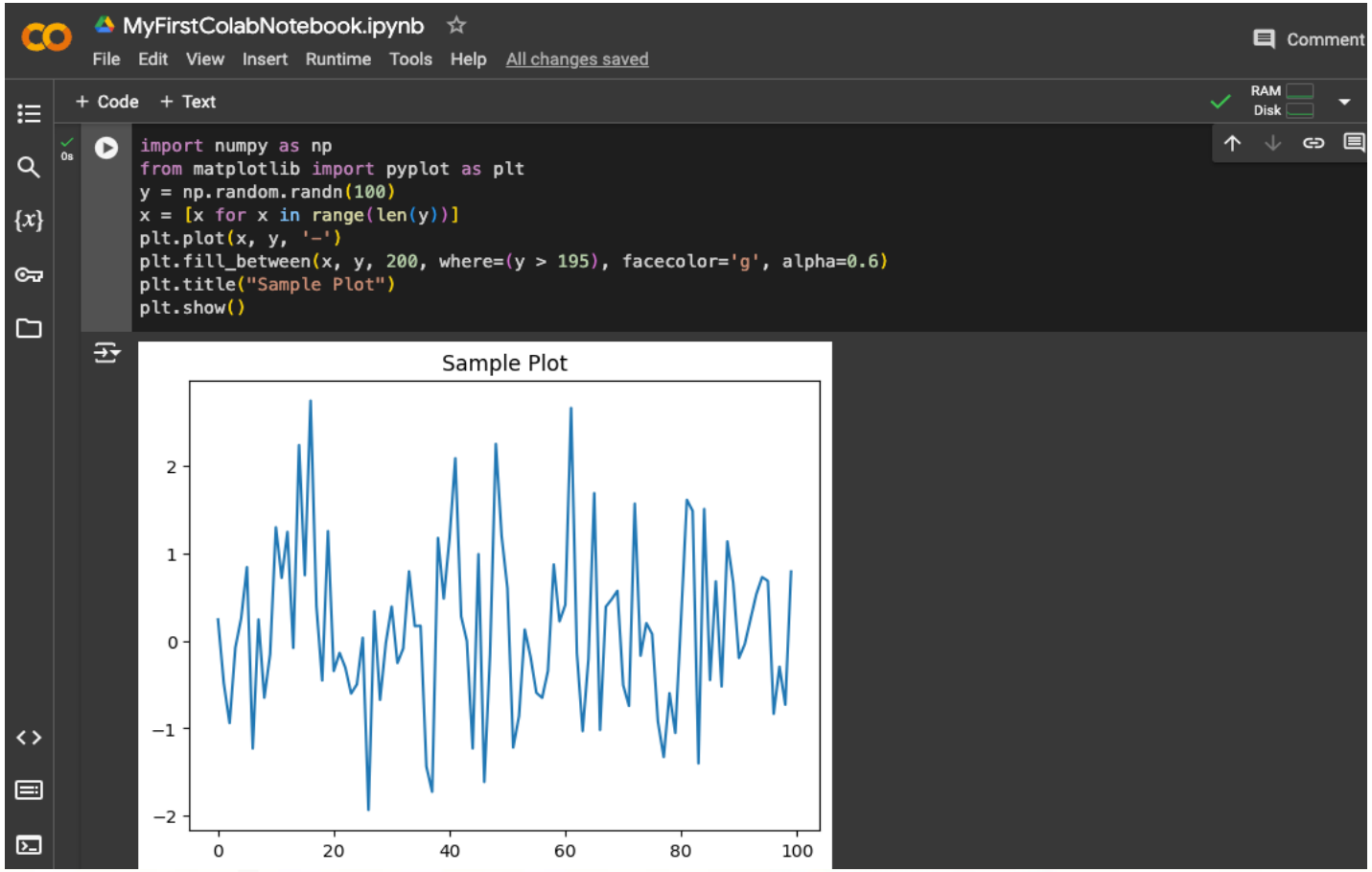
https://www.tutorialspoint.com/google_colab/index.htm

Google Colab Tutorial

Run and Example:

Type the following into a Code Cell:

```
import numpy as np
from matplotlib import pyplot as
plt
y = np.random.randn(100)
x = [x for x in range(len(y))]
plt.plot(x, y, '-')
plt.fill_between(x, y, 200,
where=(y > 195), facecolor='g',
alpha=0.6)
plt.title("Sample Plot")
plt.show()
```



https://www.tutorialspoint.com/google_colab/index.htm

THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

Homework Overview

This Week

- Reading:
  - Chapters 1, 2, and 3 in the textbook
- HW #1 (Run Python Script in Google Colab first, and then answer the homework questions)
- Discussion #1
- Start getting familiar with TensorFlow and Google Colab

- Reminder: No extensions provided. Start assignments early!

Next Steps

- Come to office hours with any questions you may have.
- Work on your HW #1 and Discussion #1 and submit them by 9:00 am ET on Saturday.
- See you next class!

Thank you!