Wrap Up

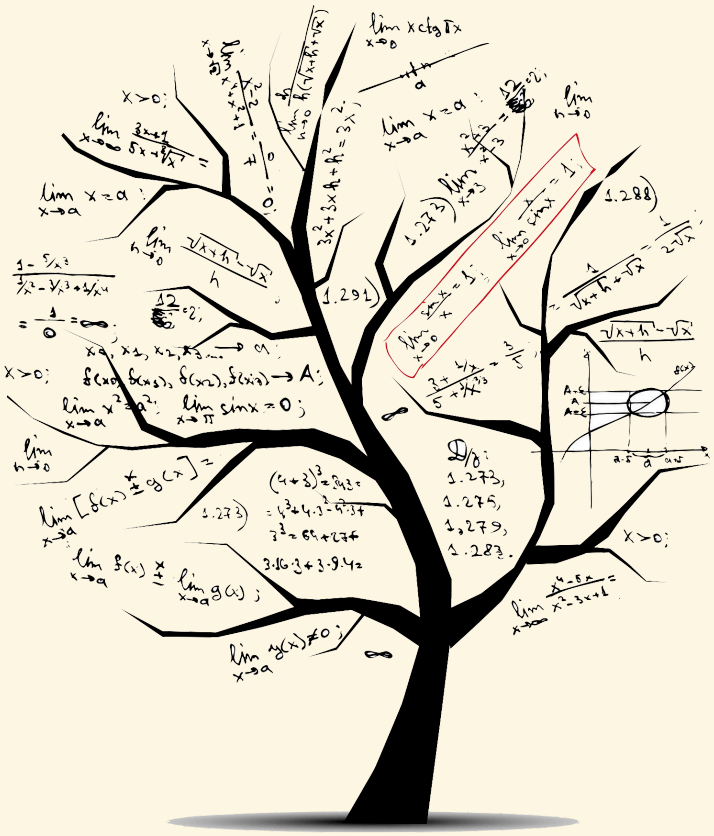
Why Are we Here?

Learning Objectives

1. **Apply** data science skills using real-world data to **create** meaningful, interactive applications addressing authentic problems.

2. **Analyze** and **implement** practical machine learning approaches to **solve** real-world problems, emphasizing pragmatic solutions over deep theory.

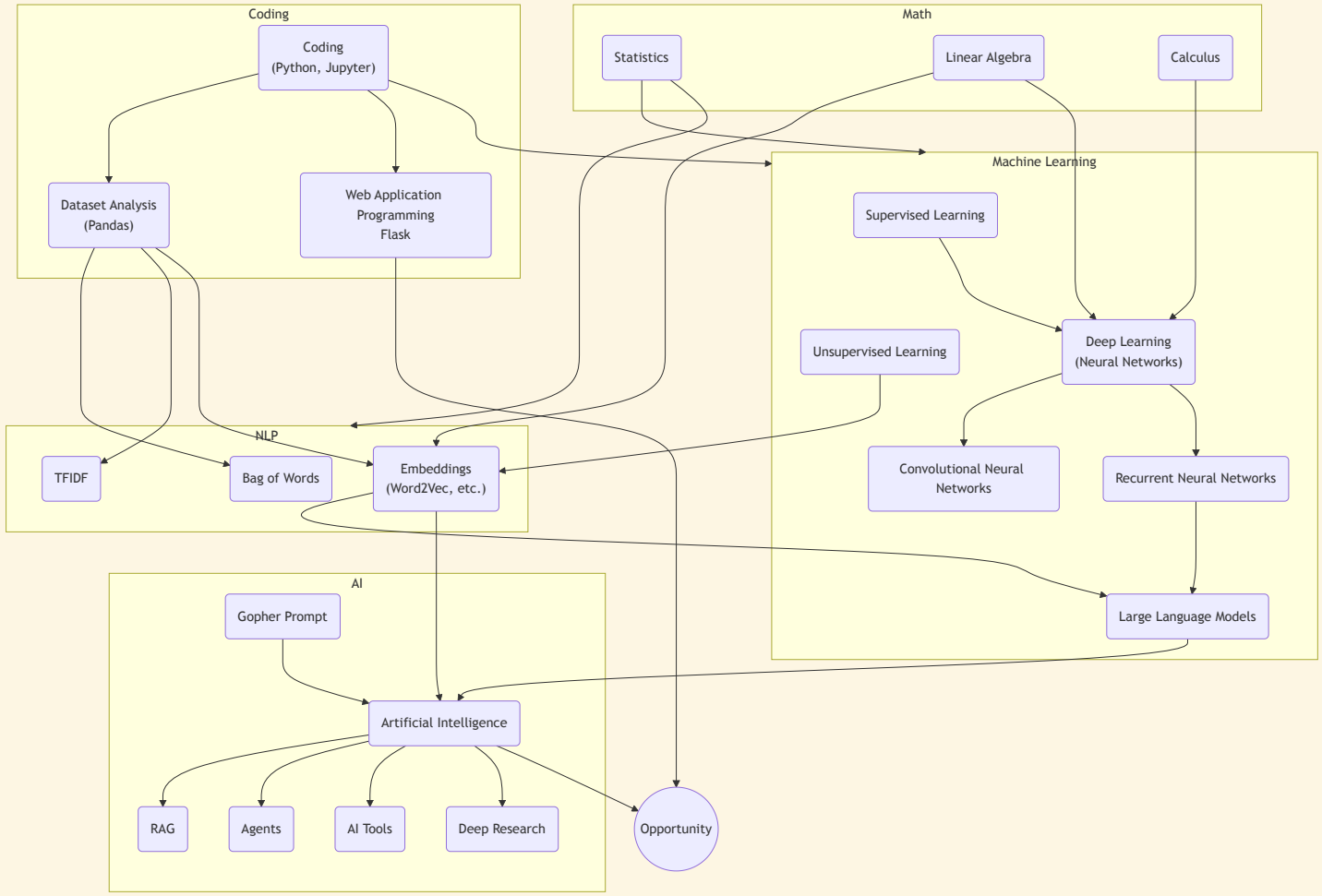
3. **Examine** and **employ** large language models (LLMs) to **enhance** data-driven applications.



Climbing the Tree of Knowledge

- **Python**: Programming language for data science and machine learning.
- **Basic Statistics**: Fundamental concepts for understanding data distributions and relationships.
- **Pandas**: Library for data manipulation and analysis.
- **Scikit-learn**: Library for machine learning algorithms.

My goal for this class was to teach you *just enough* of these tools to show you some of the fruits up in the branches of this tree.


Coding
Coding
(Python, Jupyter)
Dataset Analysis
(Pandas)
Web Application
Programming
Flask
Math
Statistics
Linear Algebra
Calculus
Machine Learning
Supervised Learning
Unsupervised Learning
Deep Learning
(Neural Networks)
Convolutional Neural
Networks
Recurrent Neural Networks
Large Language Models
NLP
TFIDF
Bag of Words
Embeddings
(Word2Vec, etc.)
AI
Gopher Prompt
Artificial Intelligence
RAG
Agents
AI Tools
Deep Research
Opportunity

Career Pathways

Software Engineer

Role: Develops, tests, and maintains software.

Focus: Often specialized (frontend/backend, product/infrastructure).

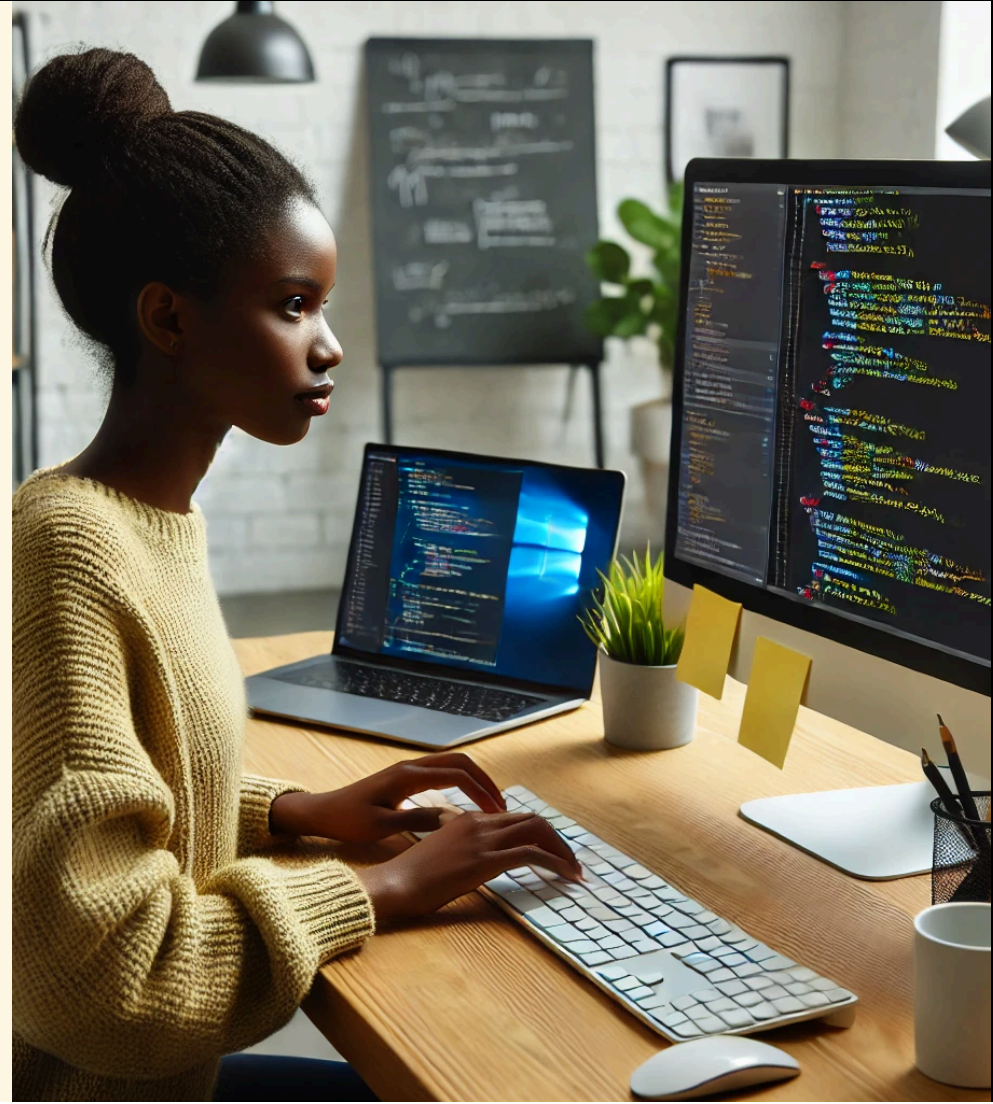
Stakeholders: Internal teams or external customers.

Overlap: Works closely with all other tech roles.

Salary: $128K–$253K
(Levels.fyi, 25th–75th percentile)

Education:

- Typical: Bachelor’s in Computer Science
- Many are also self-taught



Data Engineer

Role: Designs and maintains data pipelines and models.

Focus: Data lifecycle and governance.

Stakeholders: Mostly technical (Data Scientists, Analysts).

Overlap: Strong ties to Software Engineering and MLE.

Salary: $128K–$253K
(Levels.fyi, 25th–75th percentile)

Education:

- Typical: CS or Data Science degree
- Industry certifications are common

Machine Learning Engineer

Role: Builds and deploys ML models in production.

Focus: Performance monitoring and model serving.

Stakeholders: Technical leads or business units with KPIs.

Overlap: Primarily with Data Engineering and Data Science.

Salary: $128K–$253K
(Levels.fyi, 25th–75th percentile)

Education:

- Typical: CS or Data Science background
- Certifications or specialized courses

Data Scientist

Role: Analyzes data to extract insights and drive decisions.

Focus: Exploratory analysis, predictive modeling, and visualizations.

Stakeholders: Can be technical teams or business executives.

Overlap: MLE, Data Engineering, and Business Analysts.

Salary: $125K–$222K
(Levels.fyi, 25th–75th percentile)

Education:

- Common: Bachelor's/Master's in CS, Statistics
- Many have PhDs ("academic converts")

Business Analyst

Role: Uses data to guide business decisions.

Focus: Reporting, dashboards, and stakeholder communication.

Stakeholders: Typically non-technical teams.

Overlap: Collaborates with Data Science and Data Engineering.

Salary: $87K–$145K
(Levels.fyi, 25th–75th percentile)

Education:

- Often: Business, Economics, or Data Science degree

Research Scientist

Role: Conducts specialized research and experiments.

Focus: Developing new theories, publishing in journals/conferences.

Stakeholders: Primarily funded by grants or universities.

Overlap: Data Scientists, ML Engineers (in R&D settings).

Salary: Varies widely (often grant-based or academic scale)

Education:

- Typically a PhD in a relevant field
- Postdoc experience is common

Kinds of Workplaces

Type	Salary	Vibe	Job Security	Scope of Work
Big Tech	High	Competitive	Good	Limited
Startups	Low	Exciting	Poor	Broad
Academia	Low	Intellectual	Good	Deep
Banks	High	Conservative	Good	Limited
Government	Low	Bureaucratic	Good	Broad

Fields That Use Data Science

Manufacturing

Finance

Molecular Biology

Healthcare

Supply Chain

Journalism

Education

Legal Services

Sales

Environmental Science

Marine Biology

Public Policy

> I’m a mediocre cartoonist, a mediocre writer, and a mediocre businessperson. But I’m a combination of all three, and the intersection of mediocrity makes me successful.

~ Scott Adams

What to Study

Math

- Calculus 1, 2
- **Linear Algebra**
- **Graph Theory**
- Abstract Algebra

Statistics

- **Probability Theory**
- **Statistical Inference**
- Regression Analysis
- Elementary Stockastic Processes
- Time Series Analysis

Computer Science

- **Data Structures**
- **Algorithms**
- **Databases**
- Computer Architecture
- Operating Systems
- Networking

Soft Sciences

- Basic Psychology
- Philosophy of Language
- Logic

Teach Yourself Programming in Ten Years

Peter Norvig

Why is everyone in such a rush?

Walk into any bookstore, and you'll see how to *Teach Yourself Java in 24 Hours* alongside endless variations offering to teach C, SQL, Ruby, Algorithms, and so on in a few days or hours. The Amazon advanced search for [title: teach, yourself, hours, since: 2000 and found 512 such books. Of the top ten, nine are programming books (the other is about bookkeeping). Similar results come from replacing "teach yourself" with "learn" or "hours" with "days."

The conclusion is that either people are in a big rush to learn about programming, or that programming is somehow fabulously easier to learn than anything else. Felleisen *et al*. give a nod to this trend in their book *How to Design Programs*, when they say "Bad programming is easy. *Idiots* can learn it in *21 days*, even if they are *dummies*." The Abtruse Goose comic also had their take.

Let's analyze what a title like *Teach Yourself C++ in 24 Hours* could mean:

- **Teach Yourself:** In 24 hours you won't have time to write several significant programs, and learn from your successes and failures with them. You won't have time to work with an experienced programmer and understand what it is like to live in a C++ environment. In short, you won't have time to learn much. So the book can only be talking about a superficial familiarity, not a deep understanding. As Alexander Pope said, a little learning is a dangerous thing.
- **C++:** In 24 hours you might be able to learn some of the syntax of C++ (if you already know another language), but you couldn't learn much about how to use the language. In short, if you were, say, a Basic programmer, you could learn to write programs in the style of Basic using C++ syntax, but you couldn't learn what C++ is actually good (and bad) for. So what's the point? Alan Perlis once said: "A language that doesn't affect the way you think about programming, is not worth knowing". One possible point is that you have to learn a tiny bit of C++ (or more likely, something like JavaScript or Processing) because you need to interface with an existing tool to accomplish a specific task. But then you're not learning how to program; you're learning to accomplish that task.
- **in 24 Hours:** Unfortunately, this is not enough, as the next section shows.

Teach Yourself Programming in Ten Years

Translations

Thanks to the following authors, translations of this page are available in:

Arabic
(Mohamed A. Yahya)
العربية

Bulgarian
(Boyko Bantchev)

Chinese
(Xiaogang Guo)

Croatian
(Tvrtko Bedekovic)

Esperanto
(Federico Gobbo)

French

https://norvig.com/21-days.html

How to Be Successful

My three-step fool-proof plan to be successful in Data Science, Machine Learning, or anything.

Step 1: Be Interested

- You can do Data Science and Machine Learning without fixating on only that!
- Programming and Math are both enormous fields littered with interesting detours.
- Detours should not only be expected but embraced.

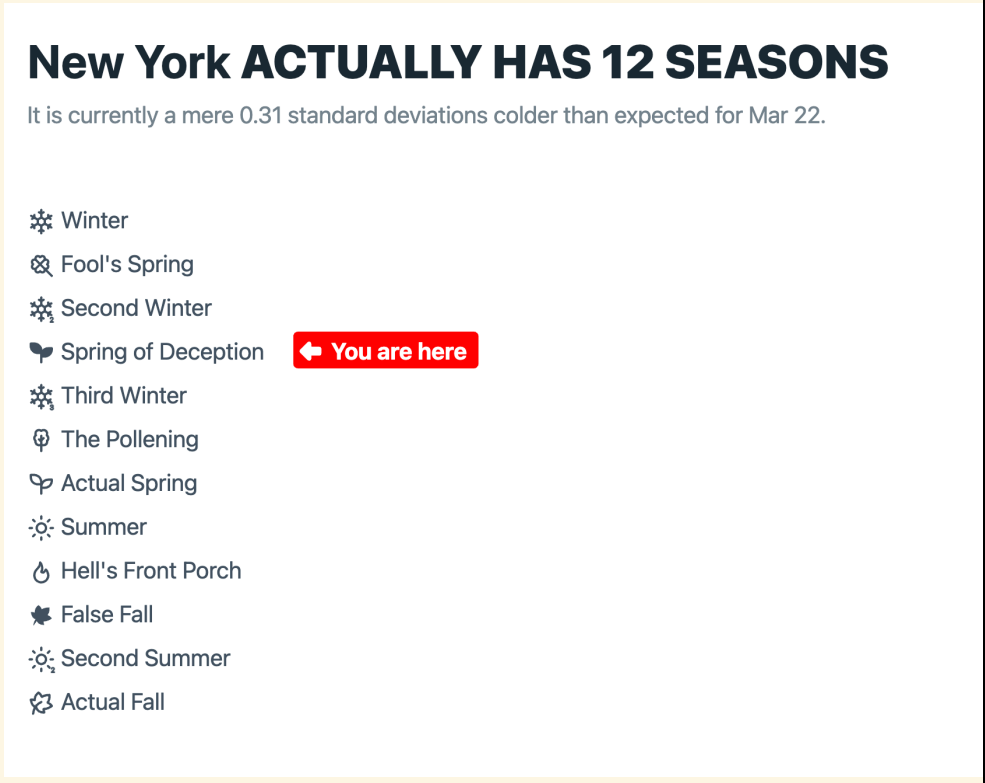
Examples

- My years as a Ruby on Rails developer taught me how the internet works.
- My hobbies as a game developer reinforced my math skills and taught me how to think about state.

Step 2: Make Stuff

- If you want to be a good coder, you need to write a lot of code.
- The same is true for Data Science and Machine Learning.
- Nothing will teach you more than doing the thing you want to be good at.
- Nothing will be more satisfying than having a stupid project idea and being about to run out an build it.

Learn some basic **Web Frameworks** (Flask, FastAPI, HTMX), **Game Frameworks**, (PyGame, Pico-8, PhaserJS), **Art Generation** frameworks (Processing, P5.js, three.js, d3.js), and **anything that interests you**.



Step 3: Find Community

Building things is a life-long journey that's more fun with friends.

Surround yourself with people who are smart, hard-working, and share your passion.

Examples

- Join your school Computer Science club.
- Take part in a hackathon.
- Attend a meetup.
- Join a Discord server.
- **Connect with your fellow students in this class.**

Thank You

- I will be writing **Evaluation Letters** for all of you. These will include details about the course and the assignments you completed and will be delivered by Columbia.
- I will email you all of the slides at the end of this class and the source code will be available at at github.com/x/columbia-bigd103-summer-2025.

Contact

Personal Email: devon@peticol.as

Personal Github: github.com/x