Introduction to Machine Learning

```
! git clone https://www.github.com/DS3001/intro
```

What is Data Science?

- Data science is a multi-disciplinary activity where the gathering, representation, analysis, and use of data are the focus
- Data science is a methodology that can be deployed anywhere: public health, agriculture, neurobiology, public policy, aerospace engineering, economics, etc.
- Analytics/Machine Learning is an important part of this process: Using data to build models that bear sufficient verisimilitude to reality to be used as predictive/explanatory/generative tools
- Most data scientists specialize in a particular field, but data science is inherently multi-disciplinary: A computer scientist, a statistician, and an applied mathematician/engineer/sociologist/physicist/economist/etc. do not make a data scientist.

What is Data Science?

- Design: The philosophy and understanding of the goals of the project and how users will interact with the systems and analysis created (e.g. human-centered design, visualization)
- Systems: The engineering of computers and devices to gather and store data in ways that facilitate understanding them (e.g. setting up a data pipeline into an SQL database)
- Analytics: The algorithms and strategies used for analysis and optimization (e.g. creating a predictive model for which people get sick/purchase a product/etc. and then recommending a treatment/price/course of action)
- Policy/Value: Understanding the consequences and limits of the analysis and ramifications of deployment/publication (e.g. should we be worried about predictive models for medical complications or loan approval or prison sentencing or AI-generated content?)
- Practice: Competence in another field that allows the productive application of Data Science (e.g. engineering, physics, public health, business, genomics, economics, etc.)

The UVA-SDS Model

(This is, itself, an example of "Design'' — seeing it shapes the way you conceptualize and think about what data science is)

This Class: Machine Learning 1

- The goal of the class is to introduce you to foundational concepts and tools in machine learning (models/algorithms, model selection/regularization, cross validation)
- More generally, we want to train you to approach data problems with cautious confidence and get experience working with data
- There will always be a shiny new language, an exciting new algorithm, and a popular new software package: Training your mind to transcend the fads and the noise to think critically and make good decisions about data and modeling is vastly more important than chasing trends

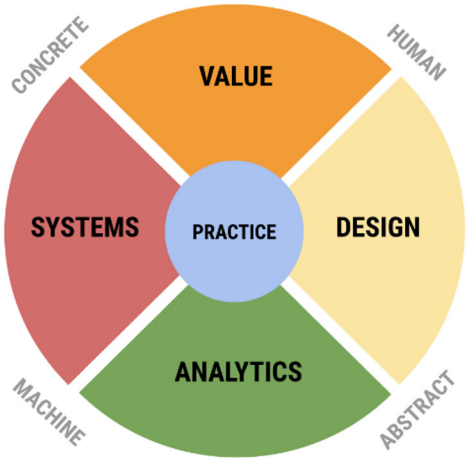


Figure 1: Data Science

Machine Learning

- Machine learning is the use of algorithms implemented on computers (the machine) to analyze data (the learning)
- The machine will be Google Colab and the Rivanna High Performance Computing Cluster at UVa, and we will use a programming language called Python to implement algorithm and estimate models, through an intergrated development environment called Jupyter Lab
- Course content will be hosted on GitHub and pulled onto Colab or Rivanna; when completed, you'll push your work back to your own Github repos where it will be graded
- The goal is for you to get off your laptop and build a portfolio of your work; this transition will be gradual

Outline of the Class

- Unit I: Procedural Programming, Wrangling Data, Exploratory Data Analysis, Data Visualization
- Unit II: Introduction to Algorithms (k-Nearest Neighbor, k-Means Clustering, Linear Models, Decision Trees)
- Unit III: Tuning/Testing/Evaluating Models (Bootstrapping, Cross Validation, Ensemble Methods, Regularization, Gradient Boosting)
- Unit IV: Select topics, time allowing (Generative Modeling, Fairness, Model Explainability and Interpretability)

Grading

- There are assignments and class projects. There are no traditional exams.
- The *assignments* are to give you experience working with computers, data, and models. There is typically a correct answer. Either your code works or it doesn't. The goal is to acquire skills and qualitative understanding.
- The *projects* are "real" work on "real" data, and answers are either better or worse, or more or less useful. Take the time to understand the data and problem, and exercise creativity and independent thinking. Embrace the challenge. Grades depend on results.
- The emphasis in this class is on gaining intuition into how the tools work through assignments and then being creative on projects

Books/References

- I do not really think you need a book specifically for this class, but many people like to have them.
- The most useful places to go are often for specific questions are CrossValidated (link) and StackOverflow (link)
- Here are books that I often referred to in putting the class together, from "easy/simple" to "hard/complex", and all are worth owning:
  - Beginner: *Introduction to Machine Learning*, by Paul Wilmott; *Introduction to Machine Learning with Python*, by Muller and Guido
  - Intermediate: *An Introduction to Statistical Learning*, by James, Witten, Hastie, and Tibshirani; *Machine Learning with Python Cookbook*, by Chris Albon
  - Advanced: *Elements of Statistical Learning*, by Hastie, Tibshirani, and Friedman; *Hands-on Machine Learning with Scikit-Learn, Keras & TensorFlow*, by Aurelien Geron

Artificial Intelligence

- Use generative AI as much as you want to implement your vision: Good programmers already use established code bases and the Internet to economize on their time
- If you use a large chunk of code generated by an AI, cite it in the code comments (there is no penalty for doing this, it is a good thing)
- If you use a large chunk of code generated by an AI, notice that you are not developing as a programmer and will eventually become dependent on the AI

This class is for everyone

- From the syllabus: "Everyone is welcome to participate in this class, of any age, culture, gender, language or geographic heritage, learning and physical abilities, political or social beliefs, race or ethnicity, religious or spiritual beliefs, sex, and social or economic class. Conversely, everyone participating in this class is expected to respect the dignity and humanity of their peers. Please feel free to approach or email your TA or instructor with instructions about how you would like to be addressed, including adjustments to or pronunciation of names or preferred pronouns."
- The foundational of a functional society is charity: Assuming that people who disagree with you are expressing their sincerely held beliefs, and that differences in beliefs are a matter of life experience and not moral superiority

Who am I?

- Terence Johnson, PhD Economics, BS Math/Econ/PoliSci
- Research: Development Economics, Mechanism Design
- Been teaching university courses since 2007 at the University of Maryland, Notre Dame, and Batten/Economics at UVa
- Conducted randomized controlled trials in Ghana, Burkina Faso, Kenya, Uganda, Senegal, Tanzania, Mexico. Member of Innovations for Poverty Action (Yale), Jameel Abdul Latif Poverty Action Lab (MIT). Research funded by the B&M Gates Foundation, PEDL, World Bank. Published in *American Economic Journal: Applied*, *Journal of Economic Theory*, *Journal of Development Economics*, *Economic Theory*, *Handbook of Market Design*
- Use machine learning to design policy interventions and maximize the impact of social/economic programs

Our Systems: Python and Jupyter Lab

- Python is a general purpose computing language with extensive machine learning support
- It is an **interpreted language**, meaning that the interactive Python interpreter waits for you to give it commands, and then it executes them
- To interact with the IPython interpreter, we'll use a browser-based IDE called Jupyter Lab

- Programming in data science is about quickly iterating on an analysis to get the best answer possible, and that's what Jupyter Lab is designed for

Our Systems: Google Colab and Rivanna HPC

- There are a variety of reasons not to install a data science stack on your personal computer (accessibility, variability, equity)
- Mostly, we want to get you off your personal computer and using the tools that make development possible in complex environments
- Our simple/entry-level option is Google Colab, and our more complex and demanding option is Rivanna HPC at UVa
- We will start on Google Golab and transition to Rivanna as enrollment in the class stabilizes

Our Systems: Git and GitHub

- Content for the class will be hosted on GitHub
- The purpose of this is to teach you another set of important tools: How to collaborate on a data science project and get off your personal computer
- In the "real world," people use tools like GitHub instead of Dropbox/Box/Google Drive, and learning about GitHub opens up a new world of software for you to use
- At the end of the class, you'll have a portfolio of your work, and you can come back to it later to get code that you've developed for future projects

Language/IDE/Software Preferences

- I mostly use R, and have done projects in C++, FORTRAN, MATLAB, Stata, and Python
- Every language is roughly the same to me at this point (except Stata), and I don't see much point in debating which is better: What matters is using the best tool for the job
- Likewise, debates about IDE's are not something I find useful: Jupyter Notebooks have grown on me, but I understand if you are used to a more full-featured IDE like VS Code or RStudio or PyCharm
- As long as you are programming in Python and submitting your results in .ipynb/.html files that the grader can easily read, I don't care how you do it: The Colab-Rivanna/Jupyter Lab solution will, however, work for everyone who takes the class
- CS students have been *loud* about this stuff in the past

What is the goal of the class?

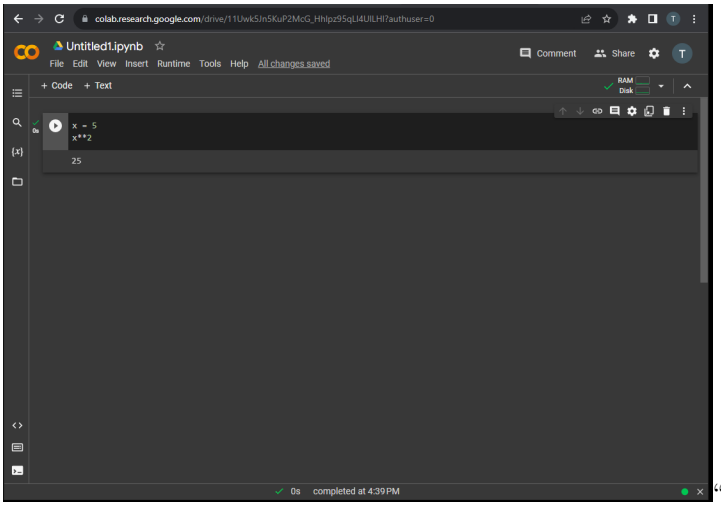
- The goal of the class is to learn ideas that transcend Python/Jupyter and whatever the next set of trends are: How do you create real insight and value?
- If all you learn is how to manipulate code and the trendiest toys and how to code some models, you are extremely replaceable. The future belongs to people who can think clearly and have applied experience that informs their analysis and decisions.
- That said, the beginning focuses a lot on the tasks that students find deeply unexciting but are absolutely necessary for data analysis to succeed.
- "Choices have consequences" is the core theme of the class – there are rarely hard and certain correct answers, and what defines our work is the different choices we make and why

Getting Started with Google Colab and GitHub

- Go to Google Colab (link) and open a `New Notebook`
- This is a Jupyter Lab session, hosted by Google
- Each "box" in the notebook window is a *code chunk*, and when you type `CTRL+Enter`, it executes the code in that chunk (`ALT+Enter` executes the chunk and creates another chunk after it)
- This makes *learning* coding, debugging, and how to use particular packages easy

- We're introducing Google Colab as a backstop technology: If there is some reason you need instant access to a Jupyter Notebook and Rivanna is unavailable for some reason, you can always use Google Colab as a last resort

Getting Started with Google Colab and GitHub



Getting Started with Google Colab and GitHub

- Git is version control software that allows teams of people to work on a project all at the same time, track their changes, and merge their work back together
- GitHub is a web-based service that hosts directories that use Git, called **repositories** or repos
- Over the course, we'll introduce more features of Git and GitHub (commit, branch, push, merge)
- The material for this course is hosted at this repository: Course Content (link)

Getting Started with Google Colab and GitHub

- To get this lecture, click `File -> Open Notebook -> GitHub` and type `DS3001` into the search box. The file `intro.ipynb` should appear, which generates these slides.
- To import the entire repo into your Colab workspace, type the following in a code chunk and execute it by pressing `CTRL+Enter`:
  - `! git clone https://www.github.com/DS3001/intro`
- There will be a GitHub clone command at the top of every lecture, and the html/handouts/slides version will be on canvas
- So wherever you are and whatever machine you're working on, you'll get able to get the class content from GitHub
- In general, files and work are not persistent on Colab unless you have Google Drive, so make sure to download any files you edit and want to keep before you exit Colab (Rivanna will be persistent)