TOOL

Cheat Sheet: Linux and IPython

Help and Autocompletion

Command	Description
`obj?`	print documentation about obj
`obj??`	print documentation about obj and any available source code defining it
`obj.`	list the attributes of obj (accessible via . typed after obj)
`sometext<AB>`	show list of possible completions after any initial characters

IPython Magic Functions (prefaced with %)

Information on magic functions

Command	Description
`%magic`	print in-depth information about set of available magic functions
`%lsmagic`	print names of all available magic functions
`%timeit?, %lsmagic?, %who?`	print help for any magic function by appending ? to the end, e.g.,
`%timeit?`	to get information about %timeit

TOOL

ML Python Packages

Python Standard Library

Where	https://docs.python.org/3/library/index.html
What	A diverse set of modules included in all Python distributions providing a wide range of functions and the ability to interact with the operating system.
When	When you need to manipulate files on your computer, perform some mathematical operations, read files in some standard formats, process text strings that you have read in from a file, figure out where your program is spending most of its time when it runs... the list goes on.

IPython

Where	ipython.org
What	An extension of the default Python interpreter that provides powerful interactive capabilities to enhance your productivity.
When	When you need to interactively explore the data with which you are working or develop new data science pipelines by figuring out the tools you need and how they can best work with each other.

Jupyter

Where	jupyter.org
What	A system that enables you to create and share documents integrating code, documentation, commentary, results, and data visualizations.
When	When you want to put together a coherent and reproducible analysis documenting a problem on which you are working, its solution, and the results of your analyses.

Matplotlib

Where	matplotlib.org
What	A 2D plotting package that produces publication-quality figures in a variety of formats, with a high degree of customization.
When	When you need to make plots of your data, including line plots, scatter plots, bar charts, heatmaps, histograms, etc.

NumPy

Where	numpy.org
What	The fundamental package for scientific computing in Python, providing multidimensional arrays and a variety of functions for acting on the data stored in arrays.
When	When you want powerful number-crunching capabilities to work with array data, or as part of the larger Python data science ecosystem, which builds new operations on top of the core functionality provided by NumPy.

Pandas

Where	pandas.pydata.org
What	A fast, powerful, flexible, and easy-to-use data analysis and manipulation tool, especially suited to working with tabular data such as is contained in spreadsheets.
When	When you need to read in data from an Excel spreadsheet or CSV file and then process it further by writing your own analysis code.

Scikit-learn

Where	scikit-learn.org
What	A powerful and comprehensive package for doing machine learning in Python, with support for many algorithms as well as testing and cross-validation procedures.
When	When you want to build predictive models from data, such as classification or regression pipelines (supervised learning) or clustering and dimensionality reduction analyses (unsupervised learning)

SciPy

Where	scipy.org
What	An integrated set of packages for computational mathematics, science, and engineering, providing convenient Python interfaces to many well-established algorithms for scientific computing
When	When you want to find the roots of an equation, identify the parameters at the minimum value of a function, integrate a differential equation, fit your data to a model, do some signal or image processing, interpolate between two data points… the list goes on.

Seaborn

Where	seaborn.pydata.org
What	A data visualization package based on Matplotlib that provides a high-level interface for characterizing statistical properties of data.
When	When you need to visualize structure in your data set, overlaying different components of data into informative visual summaries, with the ability for customization through Matplotlib.

- Histogram : . histplot( )

PACKAGES: DEMO

```
In [1]: import pandas as pd
        import matplotlib.pyplot as plt
        import seaborn as sns
```
} IMPORTS

```
In [2]: rootdir = '/Users/briand/Documents/mle/btt_course'
        infile = rootdir + '/data/cell2cell_data.csv'
```
← concatenation

```
In [3]: df = pd.read_csv(infile)
```
← Read files & puts it into a data frame. Takes the parameter of filepath.

```
In [5]: df.head(10)
```
← List 10 records

Out[5]:

	revenue	outcalls	incalls	months	eqpdays	webcap	marryyes	travel	pcown	creditcd	retcalls	churndep
0	48.82	10.00	3.00	26	780	1	0	0	0	1	4	1
1	83.53	20.00	1.00	31	745	1	0	0	0	0	4	1
2	29.99	0.00	0.00	52	1441	0	0	0	1	1	3	1
3	51.42	0.00	0.00	36	59	1	0	0	0	0	4	1
4	37.75	2.67	0.00	25	572	0	0	0	1	1	3	1
5	5.00	0.00	0.00	26	785	1	0	0	0	1	3	1
6	5.25	0.00	0.00	45	1354	0	0	0	0	0	2	1
7	26.84	4.00	1.33	49	1471	0	1	0	1	1	2	1
8	42.71	8.67	0.00	27	224	1	0	0	0	0	3	1
9	53.69	15.00	2.33	23	267	1	0	0	0	1	3	1

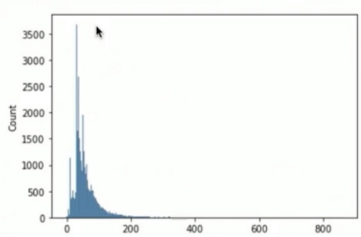
```
In [ ]:
```

Now assume that we've looking to solve & we need to understand what's the distribution of the revenue feature. When we think about DISTRIBUTIONS a good tool is using HISTOGRAM.

☆ SEABORN has a histogram method called .histplot( )

```
In [6]: sns.histplot(x=df.revenue)
```
← Takes the data frame. & in here we'll reference the REVENUE COLUMN

Out[6]: <AxesSubplot:xlabel='revenue', ylabel='Count'>



- We see a histogram with a skewed distribution.

```
In [7]: plt.savefig(rootdir+'/output/revenue_histogram.jpg')
```
← SAVE THIS

<Figure size 432x288 with 0 Axes>

Save into root dir where we have a specific folder called output and we'll name it revenue_histogram.jpg.

```
In [ ]:
```

Running code, and inspecting commands and variables

Command	Description
`%run filename`	execute the python code in filename, and bring everything from that module's namespace into the current interactive namespace
`%hist`	print list of full command history in current session • %hist -f filename : save list of full command history to filename
`%who`	print list of all defined variables
`%whos`	print list of all defined variables and their values
`%debug`	activate the interactive debugger after an error has occurred (requires knowledge of Python pdb debugger operation)
`%pprint`	toggle "pretty printing" on and off (may help with viewing contents of large objects)
`%timeit expression`	run specified Python expression, and return information how long it took to run

Interactions with the Operating System (files, etc.)

The commands below are Linux commands that you can run in iPython.

Command	Description
`%ls`	list files in current directory
`%more filename`	print the contents of filename to the screen
`%cp`	copy one file to another, or to another directory (e.g., %cp file1 file2)
`%mv`	rename one file to another, or to another directory (e.g., %mv file1 file2)
`%mkdir directory_name`	make a new directory called directory_name
`%cd directory_name`	change current directory to specified directory_name • %pwd : print the current directory
`![command]`	execute the specified command in the system shell (e.g., !ls -l)

LINUX COMMANDS

pwd = present working directory

cd = change directory command

- cd OR cd~ = navigates to home directory
- cd/ = navigates to root directory
- cd [directory name] = navigates into the current root directory
- cd [path name/directory name] = navigates into a directory in a specified location

mkdir = make directory command

- mkdir [directory name] = creates a directory in your current working directory
- mkdir [path name/directory name] = creates a directory in a specified location

ls = list files & directories

- ls = list all files in current working directory
- ls [path] = list all files in a specified location
- ls -ltr = -ltr is optional

PRELAB #1

UNIT 1: OVERVIEW

ML is the use and development of computer systems with the ability to learn and discover patterns in a data.

Ex) A computer program can determine if an email is spam or not spam.

Ex) A computer program can also find patterns among shoppers and recommend products tailored toward their needs and interests.

∴ being able to analyze & visualize data in meaningful ways is critical in ML.

ML: A NEW PARADIGM OF PROGRAMMING

↳ To apply ML in practice it's best to approach it as first a new paradigm of programming.

↳ **MODEL:** the core output of ML process. It's technically a data object that represents pattern we've observed in data. (AKA. ML program)

↳ EX. Like a function where some inputs are mapped to an output

$y = f(x)$

To achieve some decision making purpose.

We'll shift from expressing ML as pure math and instead treat it as just another programming function.

$y = f(x)$ ⟹

```
def hello_ml(x):
    #my ML logic
    return y
```

*** BENEFITS OF FUNCTIONS: Lets us reuse the encapsulated logic in our applications.

With ML, the logic is learned by finding patterns in the data.

↳ ML engineers control how the logic is learned.

```
def not_ml(x):
    if x > 100:
        return 'safe'
    else:
        return 'risky'
```

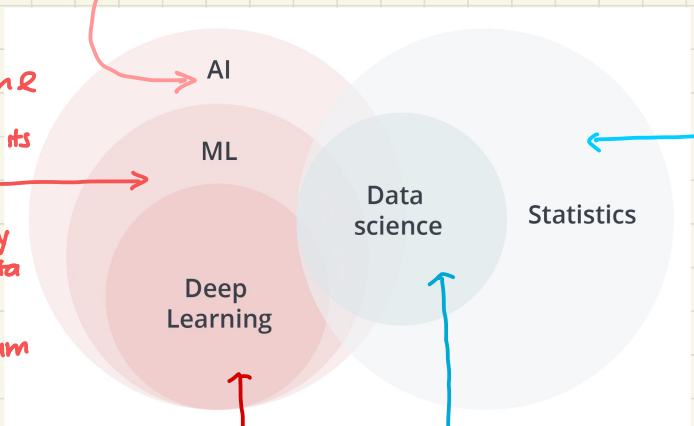
VS.

```
def hello_ml(x):
    y = model.predict(x)
    return y
```

We apply the same logic here BUT with the return method.

1) ML vs. RELATED TOPICS

Captures all the research and implementation of systems that are capable of performing tasks intelligently in a given environment.
AI should operate in an environment w/ previously unseen stimuli & is able to make appropriate decisions w/ human intervention



Deals with research & implementation of systems that shape its behavior based on experience.

★ ML solves problem by learning from past data to make decisions

★ We instruct a program HOW TO LEARN.

↳ A model needs historical data to perform well.

⇓

- It goes through a training phase to consume data (in high volume) in order to update its inner structure.

Utilizes special types of ML model (neural networks) as an underlying algorithm.

These are used to solve complex real world problems.

(Ex. image recognition)
(Ex. language processing)
(Ex. Speech recognition)

Collecting & analyzing numerical data.
USES:
① Descriptive statistic
↳ Seeks to describe the various properties of a data set.
② Inferential statistic
↳ makes inferences on a larger group based on a subset of sampled data.

Finding patterns & providing insights from data. ML aims to automate these processes. ML is capable of learning on its own from enormous data means it can pick up on relationships within data that were missed.

MACHINE LEARNING:

- In ML an algorithm learns from training data & builds a model or prediction algorithm that can be used to make accurate predictions when encountering new data.
- GOAL OF ML: To generalize well to new, previously unseen data.

EX) If a button was pressed (INPUT), soap dispenser (OUTPUT), else do nothing
↳ HOWEVER when performing tasks involving uncertainty S.A. making a prediction OR recognizing patterns, traditional programs are limited to the experiences & bias of their developers.

∴ This is where machine learning comes in.

In ML developers DON'T explicitly tell a program what to do in a given situation.

INSTEAD we tell the program to learn from DATA (historical).

The result is a PREDICTIVE PROGRAM (AKA. machine learning model)
It's used like a FUNCTION where some inputs are mapped to an output to achieve a decision making purpose.

You can use ML approaches to:

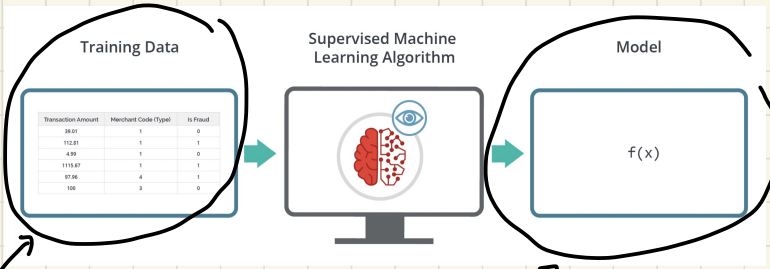
1. Aim to automatically discover general patterns in data.
2. Aim to learn relationships between specific data element in order to make predictions when encountering new, never before seen data.

→ ML is NOT just one method or algorithm. BUT a broad of different approaches, methods, algorithms, and techniques that addresses various kinds of problems.

→ You can solve most problems using one of two types of ML methods:

1. UNSUPERVISED LEARNING
2. SUPERVISED LEARNING [MORE COMMONLY USED]

→ TRAINING PHASE



NOTE: Model will/can be referred as functions

The INPUT DATA. The standard ML algorithm is run on input data.
(EX. historical housing data)
(EX. images)

⇒

The algorithm learns from that training data and produces a program known as the ML model. Which can be used to make future predictions on brand new data.

- **NOTE:** Think of supervised ML model as a function that takes in new data & outputs prediction.
- This process is an iterative one.
- The training phase can input millions of data points & improve the model's inner-wiring.
- ML models can learn from these data to inform its decision making ∴ it can pick up subtle relations & patterns that are beyond the natural capabilities of the human mind.

NOTE: It's not possible for the ML model to accurately be 100% correct.
∴ It's best to GENERALIZE from past data to produce a best guess.

Develop a model that has the ability to GENERALIZE well to new data is the focus.

- AI is everywhere thanks to ML algorithms.

EXERCISE: ML OR NOT ML?

1. An online language service matches users with tutors. All users see the same ranking of tutors, which is computed using a pre-specified formula that aggregates and weighs past student reviews of each tutor.
   ↳ ML b/c the rule that governs the ranking of tutors is pre-defined as opposed to being learned from the data.
2. A linguist is building a part speech tagged for sentences. For every input, the output of the algorithm is a sequence of tags. For example, in English, an input "I need help" would result in the output: <pronoun>, <verb>, <noun>. The linguist writes code that directly implements a series of known grammatical rules to complete the task. → NO. b/c he's hard coding & no rule learning from data occurred.

EXAMPLE: Using ML to detect if certain cc transaction are fraud.

① Assume we have historical data that represents past data of this problem.

Each column represents independent attributes of the transactions.

Location of transaction vs. account

Merchant Distance	Transaction Amount	Merchant Code (Type)	Is Fraud
5	39.01	1	0 ← FALSE
2	112.81	1	1 ← TRUE
8	4.99	1	0
6	1115.67	1	1
0.5	97.96	4	1
10.11	100	3	0
4	1.15	1	0
29.79	5.87	1	0
45.47	15.93	2	1
4	2500	1	0
0.95	25.66	2	0
21.33	2	5	0

② True goal is to access riskiness of new transaction.

↳ EX:

Merchant Distance	Transaction Amount	Merchant Code (Type)
2.44	98.88	2

} This data will be inputs into your FRAUD MODEL

ML will help us find the most relevant subset (similar users/transactions) and then make guesses on that group. This is basically GENERALIZATION & ACTION

CAUTIONARY: THINGS TO CONSIDER WHEN USING ML

Let the application drive the solution, NOT the other way around.

↳ ML is just a tool. Use it if it's the right tool for the job.

To leverage ML, you need the right data

↳ People trained in ML can tell if the company has the right data

↳ No single answer for how much data is enough.

Never underestimate the value of a good heuristic.

With data comes great responsibility.

CASE STUDY: RECOMMENDATION SYSTEMS

There are no single way that these are built ∴ many core algorithms can be used here.

Most companies that use recommander systems have the same user problem to solve.

EX) Youtube recommendation system

↳ PRODUCT GOAL: Ensure every user has access to relevant, informative, and/or entertaining content w/ little friction as possible.

↳ Recommendation engine is just one part of the bigger system.

↳ It acts as another function: takes in data & returning recommendation.

↳ GENERALIZATION: recommending new relevant videos based on videos that similar users have historically found to be relevant.

THE ML TAXONOMY

DATA MATRIX: A structured table consisting of rows & columns. A collection of examples that are either labeled/unlabeled.

Features | Label

Merchant Distance	Transaction Amount	Merchant Code (Type)	Is Fraud
5	39.01	1	0
2	112.81	1	1
8	4.99	1	0
6	1115.67	1	1
0.5	97.96	4	1
10.11	100	3	0
4	1.15	1	0
29.79	5.87	1	0
45.47	15.93	2	1
4	2500	1	0
0.95	25.66	2	0
21.33	2	5	0

Examples

FEATURES: Input variables. Each individual feature is an attribute that describes the data. (AKA. dimensions)

↳ Features are grouped together as a vector (FEATURE VECTOR) represented by X.

↳ Each individual feature in the vector is represented by: $x_1, x_2, x_3, \ldots, x_n$

LABELS: The attribute we want to predict (AKA. target variable OR output variable)

↳ Represented by y

EXAMPLES: instance of variable. It corresponds to a row in the data matrix. (AKA. datapoint)

↳ LABELED EXAMPLES: contains features [SUPERVISED]

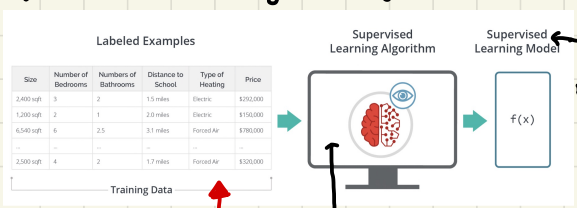
↳ UNLABELED EXAMPLES: contains only features NO labels. [UNSUPERVISED]

SUPERVISED LEARNING

Attempts to discover relationship between features & an associated label for the purpose of future predictions.

- Creates a program [model] that learns from past data to make predictions on similar new data.
- Learns to predict a label

TRAINING: Creating + learning the model.



← A predictive model is built & iteratively improved by using the label as means to check its correctness.

run on training data comprimised of labeled examples

enabling the algorithm to gradually learn the relationship between features & labels.

Once training is complete, the result is an optimal model that can be used to make future predictions

MODEL:

A supervised learning model is the learned program that is able to make predictions on new, previously unseen data.

In supervised learning has 2 phases:

1. Training phase = model is built
2. Prediction phase / Inference phase = model is used to make predictions.

UNSUPERVISED LEARNING

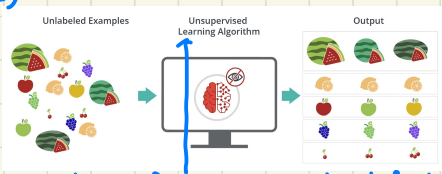
Discovering patterns in data containing labeled examples. It DOES NOT provide the answers (the label).

→ Discovers patterns on its own.
→ Uses unlabeled examples (feature values).
→ Algorithm discovers patterns in data typically by finding examples that are similar to each other & mapping them into different segments / clusters.

EX) Provided diff images of diff animals. The algorithm determines pictures of humans from 2 legs, 2 arms, walks upright, etc.

CLUSTERING: unsupervised learning technique that groups subsets of data that are collectively similar to one another based on the similarity of their feature values.

EX)



Learns from images that fruits come in different sizes & shapes & colors. After learning their characteristics it can categorize the fruits

GENERALIZATION: A model's ability to adapt to new, unseen data that's ≅ to the data that has been trained on.

↳ NOTE: While it's important to make accurate predictions on training data, the ultimate goal is to create a model that can do well on future data

↳ MAKE SURE that our model GENERALIZES to NEW EXAMPLES!

SUPERVISED LEARNING ALGORITHMS:

① **CLASSIFICATION:** The process of recognizing, understanding, and grouping ideas & objects into preset categories or "sub-populations". Use input training data to predict the likelihood of future data that'll fall into predetermined categories.

↳ Labels are discrete or categorical values.

↳ With classifications there are multiple classifications:

↳ **BINARY CLASSIFICATION:** problems where the Q itself is yes/no, T/F, reject/approve.

- We can map each class to "+" or "−" OR 1/0
- EX) Spam mapped to "+" & non spam mapped to "−"

↳ **MULTICLASS CLASSIFICATION:** label we're predicting belongs to 3 or more distinct label values.

- EX) What animal is in given image — cat, dog, rabbit, or frogs

EX) Whether an email is spam or not [BINARY CLASSIFICATION]

EX) What is the topic of given article [MULTICLASS CLASSIFICATION]

EX) Whether a CC transaction is fraudulent or not [BINARY CLASSIFICATION]

NOTE: When dealing w/ classification problems a label = class label

② **REGRESSION:**

The labels are continuous or any REAL NUMBER.

EX) What will the stock price be tmmorow?

EX) Housing price of new home in NYC.

EX) Minimum & maximum temperature on friday

EXAMPLE: SUPERVISED LEARNING PROBLEM

BUSINESS PROBLEM: We're a new credit lending start up. We know credit lending is a risky business ∴ we want to build a credit scoring system to make better lending decisions for people applying for credit.

⇓ [Translate into machine learning problem]

Build a default prediction model where default is defined as the customer missing 3 consecutive payments in the first 6 months.

⇓ ↳ **NOTE:** We're precise in defining our default. This will help us when we code it out.

Definition of default can be translated into T/F Q.

∴ with this label we have a binary classification problem.

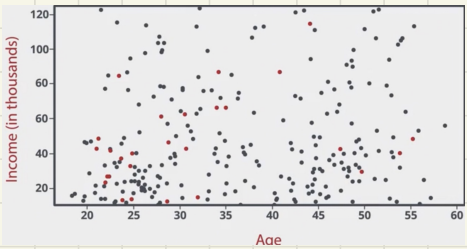
↳ To build our model, we'll need data

↳ If you have no data this is called **COLD START PROBLEM**. To combat this issue we start the experiment where we give credit to everyone who applies. When they do, we collect their AGE, INCOME as our main predictors.

We then observe for the first 6 months to determine if they've default or not.

Age	Income (in thousands)	y
25	21	0
26	32	0
24	23	0
30	55	0
27	11	0
20	46	0
24	13	1
26	29	0
26	19	0
20	17	0

When we plot the data we can observe the relationship between age, income, & defaults:

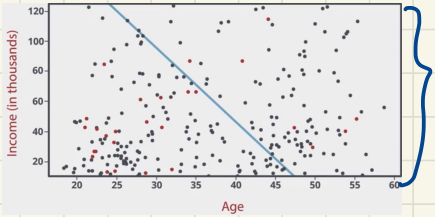


NOTE: We plotted the default customers as RED vs. BLACK.

With ML our goal is to GENERALIZE a pattern from observed examples from the data. This means looking beyond the individual data points & trying to find regions on the graph that have higher density of defaulters.

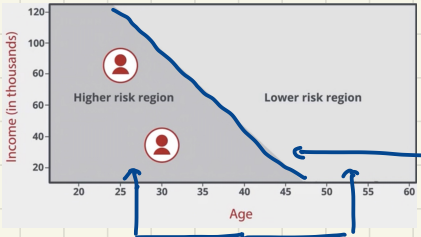
AS AN EXERCISE: Draw a single line that'll best separate the RED from BLACK points.

ML helps us improve from lenders who would draw the lines using heuristic. However there are multiple problems: leaves a good value of people & discriminatory practices.



OPTIMAL line using ML which helps us ultimately in reducing our error in predicting default incorrectly.

Once we have a model, we can ignore the original training data:



This line represents our best generalization of the data. We can use this to solve our business problem.

There is still a continuum between them.

- We can use the distance from the line to set appropriate credit limit.

EXAMPLE: UNSUPERVISED LEARNING PROBLEM

BUSINESS PROBLEM: Assume we have our credit model and are doing well using it to assess lending risk of our single product. But as our company evolves, we realize we're not serving the needs of all our potential customers with just one product.

⇓

NEW BUISNESS PROBLEM: Can we build different lending products that are tailored to specific needs of our different customers & target them at the time of application?

⇓ [TRANSLATE TO ML PROBLEM]

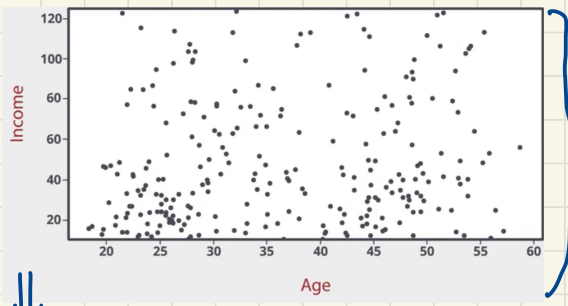
MACHINE LEARNING PROBLEM: Can we identify clusters of similar customers based on their observed attributes at application time.

> **NOTE:** No mention of specific outcome } Indicates
> Keywords: "cluster" & "similar" } UNSUPERVISED LEARNING

GOAL: Match an applicant to the right product at time of application, we'll need to limit the data we use to what's available at application time

⇓

Start with some basic visualization:



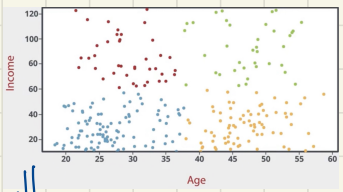
Scatterplot between AGE & INCOME.

We want to segment these points into discrete clusters where members of each cluster are ≅ to each other & ≇ to members outside of the cluster.

⇓

Draw 2 lines to split into 4 segments

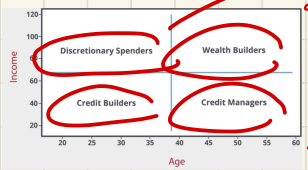
∴ using our unsupervised learning clustering algorithms:



Now it's up to us to add meaning & utility to these clusters.

NOTE: In unsupervised we need to add subjective interpretation to derive full benefit.

↳ EX)



↗ Helps humanize segments + inform our product development process.

In many companies will conduct research/surveys in each segments to build strong human connection to them.

⇓

Design product to each specific needs.

THE ML LIFECYCLE

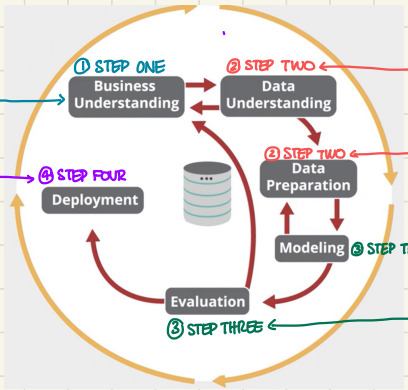
The core of machine learning is a set of computational algorithms driven by a mixture of mathematical techniques from calculus + linear algebra + information theory.

MACHINE LEARNING PROCESS : CRISP-DM

Start by understanding the problem in a business + domain + technical perspective.

When building models in practice we start w/ a written project brief where you document:
- ↳ motivation
- ↳ key constraints
- ↳ Planned approach + timeline

- Differentiates ML engineers from data scientists.
  - ↳ Deployments are left for ML engineers.



Determine if you have the right data.

- When we learn model collections we'll see how intertwined they are

---

SYSTEM RISKS: Applies to all software systems. It's the likelihood for complex & dynamic systems to have failure points.

↳ Up to ML engineers to implement the appropriate tests and monitors to be able to detect & mitigate such problems.

ETHICAL RISKS: The likelihood for large scale & automated decision systems to cause unintended harms.

↳ It's been shown that models that perform well on average can hurt non-majority & historically marginalized groups.

ML PROBLEM FORMULATION

First stage in ML development cycle is: **PROBLEM FORMULATION.**

Questions to consider when handling practice problems:

1. What problem is the model solving & why is it better than a non-ML based solution?
2. What kind of data would you need?
3. How would you solve this without ML?
4. What are the potential risks we should anticipate?

CASE STUDY: RECOMMENDATION SYSTEMS revisited

Youtube recommendation system:

Recommendation systems for any type of product typically start with a common data pattern. We call this **USER ITEM MATRIX**

	Item 1	Item 2	Item 3	Item 4	...	Item k
User 1	2	1			...	
User 2		2	4		...	2
User 3	3				...	
User 4	1	2	5	3	...	
User 5	3	2			...	
User 6					...	1
User 7		4	1		...	4
User 8		4	2		...	5
User 9	1				...	
User 10			3	4	...	1
...	...	...	...	...	...	
User N				1	...	4

ROWS = individual users/customers

COLUMNS = items

For our Youtube example:

ROWS = individual videos

← We have entries 1, 2, 3, 4, 5 that can tell us the ratings for how much it's consumed.

The values in the matrix can vary. But they should represent some level of consumption or engagement with specific items.

EX) Binary entries indicating if they viewed the video

EX) Numeric entries indicating ratings or level of consumption.

To make recommendations we can either use supervised/unsupervised approach.

SUPERVISED APPROACH

KEY: identify a good label

Label = entries of the matrix.

- If labels are binary ⇒ classification problem
- Labels are 1-5 ratings ∴ we can use regression.

- Features of the model will be whatever you have on both the user & the videos.
- For video recommendations we might use prior videos watched, descriptions, genre, video author as features.

UNSUPERVISED APPROACH

Use clustering algorithm & the key is to treat members of the same cluster as the group that'll help select & rank videos from a given user.

EX) When shopping online we have items tailored to show ≅ items to what we've been looking at. This type of recommendation is driven by ITEM BASED CLUSTERING

HYBRID APPROACH

There are often too many items & it would be too $ to run in a brute force way. ∴ we have the hybrid approach

We can use the resulting clusters to filter the billions of video ↓ to just a few thousand. Then we can use supervised model to score the smaller set.

NOTE: Hybrid is used to gain better accuracy of the supervised learning approach w/ the scalability of the unsupervised

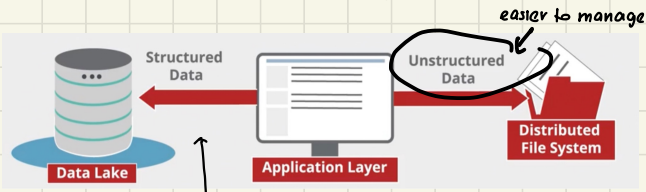
THE ML TECH STACK

Before building data we need to ask ourselves the following questions:

1. Where do I find my data?
2. Where do I actually build the model?
3. How do I access these different systems?

As a ML engineer you'll most commonly be accessing different data sets in DATA LAKE. Data is sent to data lake through logging processes & SQL updates to databases from application layer.

A general software controlling the entire system.



Data sent to a database is typically structured meaning it has a well defined schema or format that's the same for every record

Once you know where your data lives we have to study its format. You'll need to access it for analysis, exploration, & modeling.

↳ Usually done in IDE ⟸ main point for accessing data ⟹ Jupyter Notebook

---

ML WORKFLOW APPLICATIONS

- Data is often stored in Linux-based server.
- To access the data directory, ya have to use the terminal, navigate the data using typical command line tools.

IMPORTANT: Paths are very important!

Ex) Assume you have a folder in your desktop named myFolder and creak a file in the folder named myFile.txt. The path is:

⇓

"/Users/myUserName/Desktop/myFolder/myFile.txt"