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 file path.

```
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

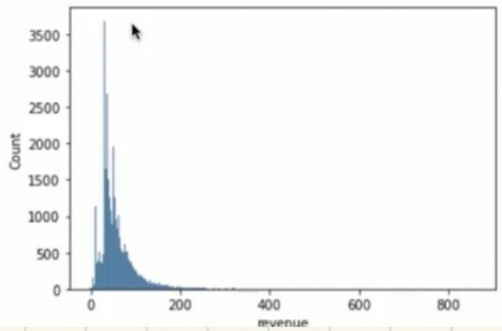
```
In [ ]:
```

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 COMMAINDS

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 interest.

∴ 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'
```

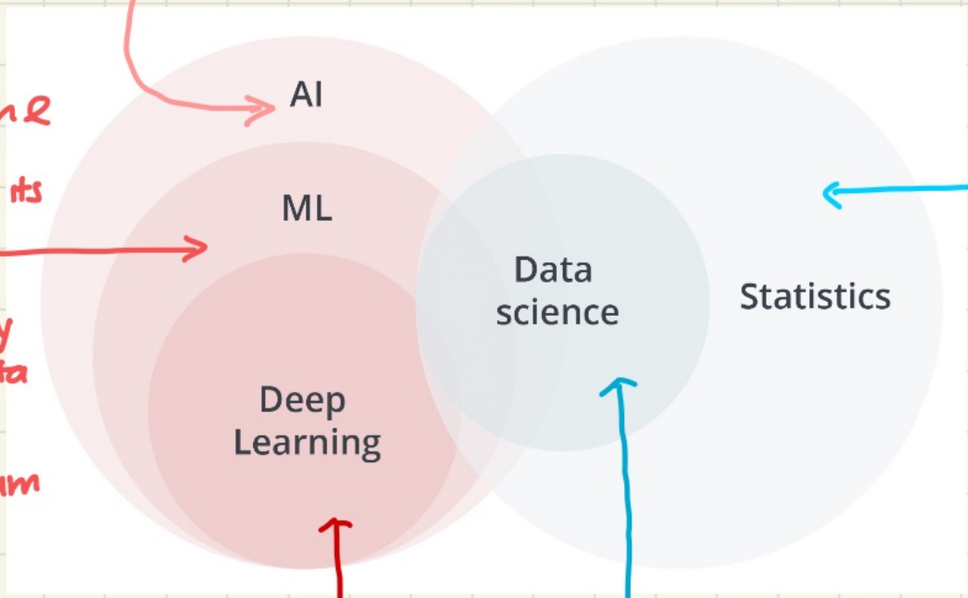
VS.

```
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 statistics
   - Seeks to describe the various properties of a dataset
2. Inferential statistics
   - 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:

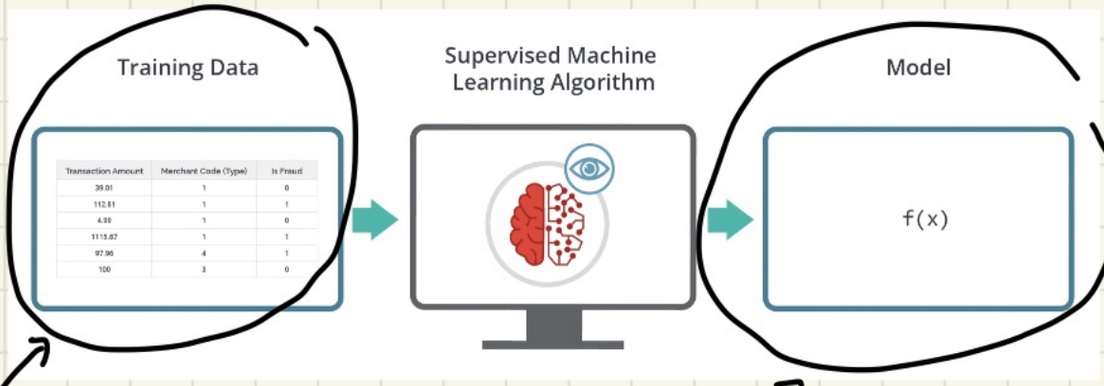
1. Aim to automatically discover general patterns in data.
2. Aim to learn relationships between specific data element in order to make prediction when encountering new, never before seen data.

→ 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.