NumPy and Pandas

Please do a **git pull** to update your repository with today’s lecture materials. We will practice more with NumPy today and then go over the Pandas library.

[The Big Picture]

A large part of data science is asking useful questions and making good decisions.

- “How many undergraduate students does UC Berkeley accept per year?” might be a little useful. “How many transfer students does UC Berkeley accept per year” might be more useful if you are a transfer student. But we don’t stop there:

- “What percentage of transfer students who apply to UC Berkeley get accepted per year?”

- “How much does GPA, extracurricular activities, gender, age, race, and time of year affect your chances of getting into a dream school?”

- “If a lot of people who got accepted to Stanford scored over 2200 on the SAT does that mean I have to beat that score?”

If you have the data, NumPy and Pandas will help answer these questions to help you make better decisions. Companies also ask similar questions and the decisions companies make based on data analysis might mean 10% more revenue next year or 6% more customer satisfaction.

To answer these types of questions, you will first need to be comfortable programming. Python and R are probably the most popular data science programming languages. After you are comfortable with basic programming you typically start to learn statistics and modeling. Knowing more statistics other than “taking the mean” means more questions you can answer!

[For Loop Exercise]

The more hours you spend programming, the better you become!

--- Here I wrote down the hailstone sequence procedure. Look up “Collatz conjecture” on Wikipedia to learn more about these special numbers---

[NumPy Exercise]

Create and multiply the following two matrices 1) element-wise and 2) using standard matrix multiplication:

1, 0, 1 2, 4, 6

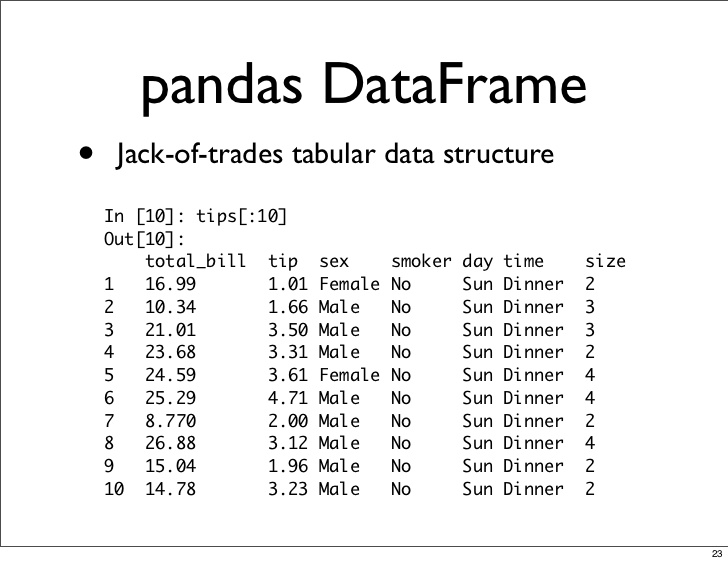
2, 1, 0 -1, 0, 2

3, 2, 1 4, 3, -2

When I say “standard matrix multiplication” I mean the following (--- I wrote an example on board ---):

[The Pandas Library]

Why use another library when we already have NumPy? It turns out Pandas lets you do several things that would take a lot more effort to do in NumPy. **Pandas should the library you use the most.**



Unlike NumPy, Pandas holds its data in something called DataFrames. You can think of them like NumPy arrays *except* DataFrames allow you to store any kind of data type. Recall from last week that the most common data types are floats, integers, strings, and booleans (true and false values or equivalently 1 and 0). DataFrames allow you to store Strings in one column and Integers in another, for example.

To import this library type: **import pandas as pd**

To read in a comma-separated-file type: **data =** **pd.read\_csv(“<filename>”)**

To view the first 5 rows of your data type: **data.head()** or **data[:5]**

Notice that you can use the “**:**” notation for subsetting your data! The following section is how you subset a pandas DataFrame. But before that, let’s play around with some data.

--- Short example of reading in the Titanic data set ---

[Pandas – Subsetting]

Subsetting in Pandas is very similar to subsetting in NumPy. There are four main ways of doing so:

----------------------------------------------

**data[5:10]**

**data[“Age”]**

**data.iloc[1:3, 8:12]**

**data.loc[2:7, “Pclass”: “Age”]**

------------------------------------------------

**[start\_index: end\_index]** lets you select rows.

**[column\_label]** lets you select ONE column.

**iloc[:,:]** lets you index by rows and columns.

**loc[:,:]** lets you select by row labelsand column labels. Column labels are the same thing as column headers (the names for each column). Row labels are by default the integers 0, 1, 2, … <last row number>. You can see these row labels on the very left hand side of the DataFrame output.

--- I demonstrated all of these subsetting examples here ---

[Pandas – Data Statistics]

The following is a list of functions that every DataFrame has. Think of these functions as “living” in each DataFrame, ready to be used using the “**.**” notation.

**data.shape**  The dimensions of the DataFrame (rows, columns)

**data.head(n)** Show the first n rows of the DataFrame

**data.describe()** Computes the mean, median, mode, and quantiles of each column

**data[<column\_label>].value\_counts()** Counts how many of each unique value there are.

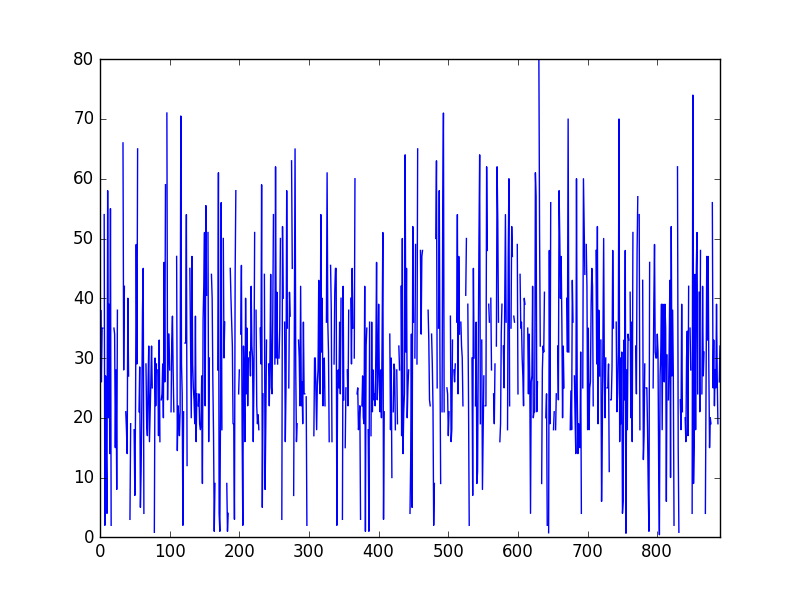
**data.fillna(<any number>)** Fills each missing value of the DataFrame with any number.

--- Here I demonstrated most of these methods with the Titanic data set ---

[Pandas – Plotting]

Plotting in pandas is really easy. You can make line plots and histograms for example. To plot the “Age” column, just type:

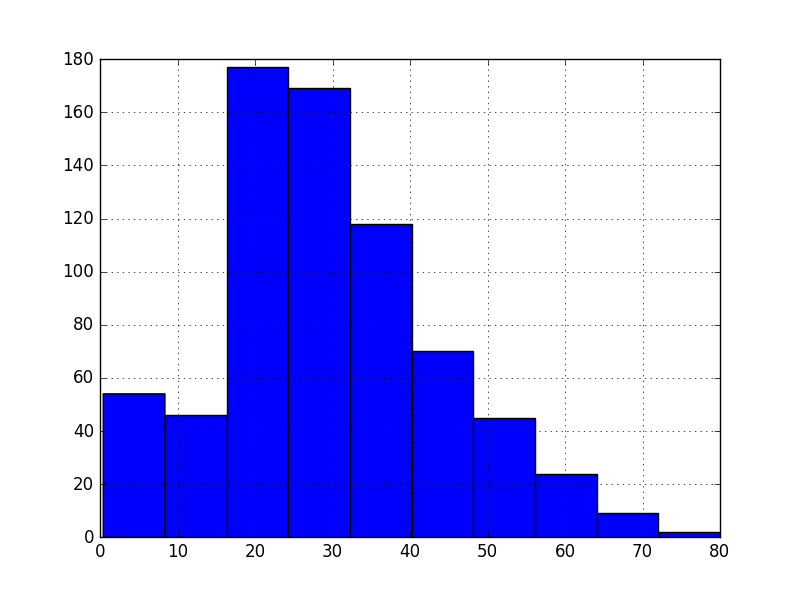
**data[“Age”].plot()**



Ouch! That looks terrible ☹ The default plot is a line plot, which doesn’t make any sense for this type of data. Line plots are used when we are plotting over time, for example, not when trying to plot Age!

Question: What is the age distributionof the Titanic passengers? The distribution will show how many people were young, old, and in-between on the ship.

**data[“Age”].hist()**



This histogram tells us much more information on what the distribution of age was for passengers on the boat. I will talk more about plots (how to add titles, colors, etc.) next week!

[Quiz Instructions]

Quizzes and homework require you to download the appropriate files from my github repository. A repository is a storage space that you can dump all your files and data in, much like your own computer’s file system. In your case, you are going to download my files. To do so, in a separate terminal window (launch another Git Bash terminal in Windows or Command+t on Mac) type:

**git clone** [**https://github.com/jc003/datascience\_yeah.git**](https://github.com/jc003/datascience_yeah.git)

This will copy my repository/folder named “datascience\_yeah” into your computer. Explore the repository. You will find folders for homework, quizzes, and my notes. Your homework that is due next week is already there.

Right now navigate to the quizzes folder and open up the quiz template in a text editor. I will be writing the questions on the board so go ahead and open the file. You will have roughly 20 minutes to work on the quiz. Quizzes are to be done individually with no collaboration. When you are finished, email your python code file to [jerryc@berkeley.edu](mailto:jerryc@berkeley.edu) and wait for others to finish.

--------------------------------

In the future if you want to update your repository (because I made some changes to the quiz or homework file) or to grab new homework assignments type in your terminal:

**git pull**

\*If you get a merge error, that means git does not know how to update your files. This happens for example when you start writing solutions for hw1.py, but then I make changes to the blank hw1.py template to the repository. When you do “git pull”, git (the program) will look at my hw1.py file and your hw1.py file and be confused because it won’t know which lines to add or delete in the correct order. To fix, make a copy of your hw1.py (e.g. hw1\_copy.py), delete your hw1.py, type git pull, then copy your solutions from hw1\_copy.py to the fresh copy of hw1.py.

[Homework Instructions]

You have already downloaded the homework for this week. Navigate to the appropriate folder and open up **hw2.py**. Having done the readings might help for this particular assignment. In general, if you don’t know what a function does, go look it up! Doing google searches on your own should be a natural habit. Homework 2 is due before start of class next week 4/22. Likewise for quizzes, please email your python code file to [jerryc@berkeley.edu](mailto:jerryc@berkeley.edu) once you are done.

Note: Any time you see “FILL ME” that is a problem worth 2 points. I give 2 points for a correct answer. 1 point for a wrong answer but good progress was made, and 0 points for leaving it blank.

Note: The homework problems are intended to be more challenging than the in-class examples or the quiz problems. I highly recommend working with other students together when doing these problems.