Introduction to Big Data and Machine Learning Preliminaries

Dr. Mihail

August 20, 2019

Big Data Course

The plight of the professor

- I have to develop a structured learning framework for advanced CS topics, where you, the students, learn new things and are competent to apply the knowledge in a “contest”

Big Data Course

The plight of the professor

- I have to develop a structured learning framework for advanced CS topics, where you, the students, learn new things and are competent to apply the knowledge in a "contest"
- Imagine yourself in a "knowledge/skill contest" with a good student from Georgia Tech. This contest is for a high paying job.

Big Data Course

The plight of the professor

- I have to develop a structured learning framework for advanced CS topics, where you, the students, learn new things and are competent to apply the knowledge in a “contest”
- Imagine yourself in a “knowledge/skill contest” with a good student from Georgia Tech. This contest is for a high paying job.
- Realistic?

Big Data Course

The plight of the professor

- I have to develop a structured learning framework for advanced CS topics, where you, the students, learn new things and are competent to apply the knowledge in a "contest"
- Imagine yourself in a "knowledge/skill contest" with a good student from Georgia Tech. This contest is for a high paying job.
- Realistic?
- Yes!

Big Data Course

The plight of the professor

- I have to develop a structured learning framework for advanced CS topics, where you, the students, learn new things and are competent to apply the knowledge in a “contest”
- Imagine yourself in a “knowledge/skill contest” with a good student from Georgia Tech. This contest is for a high paying job.
- Realistic?
- Yes!
- What problems can you identify?

Big Data Course

The plight of the professor

- I have to develop a structured learning framework for advanced CS topics, where you, the students, learn new things and are competent to apply the knowledge in a "contest"
- Imagine yourself in a "knowledge/skill contest" with a good student from Georgia Tech. This contest is for a high paying job.
- Realistic?
- Yes!
- What problems can you identify?

Preparation: math and programming

- I believe background knowledge is highly variable, but potential is not

Big Data Course

Math

- Calculus: understanding functions and how they change

Big Data Course

Math

- Calculus: understanding functions and how they change
- Statistics: understanding of collections of numbers and the stories they tell

Big Data Course

Math

- Calculus: understanding functions and how they change
- Statistics: understanding of collections of numbers and the stories they tell
- Linear algebra: many complex systems can be modeled by linear equations. Linear algebra is central to almost all areas of mathematics. Understanding machine learning algorithms rests fully on linear algebra.

Big Data Course

Math

- Calculus: understanding functions and how they change
- Statistics: understanding of collections of numbers and the stories they tell
- Linear algebra: many complex systems can be modeled by linear equations. Linear algebra is central to almost all areas of mathematics. Understanding machine learning algorithms rests fully on linear algebra.

Programming

- Obviously, you are expected to know how to write code

Big Data Course

Math

- Calculus: understanding functions and how they change
- Statistics: understanding of collections of numbers and the stories they tell
- Linear algebra: many complex systems can be modeled by linear equations. Linear algebra is central to almost all areas of mathematics. Understanding machine learning algorithms rests fully on linear algebra.

Programming

- Obviously, you are expected to know how to write code
- More importantly, at this point, you should be well-rounded enough to be confident that learning any new imperative language (such as Python) or functional language (such as Scala) is a self-study activity, not the responsibility of a upper-level CS course

Big Data Course

Independent learners

- Through this class, when I lecture, prerequisite topics that you have no background on will show up
- Good reactions:
  - Lemme google and learn more about this
  - Lemme ask Dr. Mihail where I can learn more about this
  - Lemme read the textbooks for background
- Bad (not useful) reactions:
  - He didn't teach us this, how does he expect me to pass his exams?
  - This class is too hard, I'll just do bare minimum and prolly get a C.
  - I'm completely lost. I'm not gonna do anything about it until the end of class when I'll ask: "what can I do to get an A in your class?"

Big Data Course

Math for ML book - super resource

- https://mml-book.github.io/book/mml-book.pdf

Python references

- https://www.learnpython.org/
- http://cs231n.github.io/python-numpy-tutorial/
- https://scikit-learn.org/dev/_downloads/scikit-learn-docs.pdf
- https://realpython.com/python-matplotlib-guide/

General Python

Prototyping

- In this class and most data sciences, a prototyping language is used to develop (e.g.,: Python)
- Once concept has been shown to work as intended, prototype code is translated to production
- Prototyping typically done incrementally:
  - Interactively, in the shell
  - By running a whole script, similar to compiling then running (less common)

Python Language

General stuff

- No mandatory statement termination characters
- Blocks are specified by indentation
- Statements that expect an indentation level end in a colon (:)
- Comments start with the pound (#) sign and are single-line
- Docstrings start and end with three single quotes ”’
- Values are assigned (in fact, objects are bound to names) with the equals sign (=), and equality testing is done using two equals signs (==)

Python data types

Data structures

- Lists
- Tuples
- Dictionaries (aka hash tables)

```
>>> sample = [1, ["another", "list"], ("a", "tuple")]
>>> mylist = ["List item 1", 2, 3.14]
>>> mylist[0] = "List item 1 again" # We're changing the item.
>>> mylist[-1] = 3.21 # Here, we refer to the last item.
>>> mydict = {"Key 1": "Value 1", 2: 3, "pi": 3.14}
>>> mydict["pi"] = 3.15 # This is how you change dictionary values.
>>> mytuple = (1, 2, 3)
>>> myfunction = len
>>> print(myfunction(mylist))
```

Python comprehensions

List comprehension

Old way:

```
new_list = []
for i in old_list:
    if filter(i):
        new_list.append(expr(i))
```

New way:

```
new_list = [expr(i) for i in old_list if filter(i)]
```

Python slices

Slicing

- You can access array ranges using a colon (:)
- Leaving the start index empty assumes the first item, leaving the end index assumes the last item
- Indexing is inclusive-exclusive, so specifying [2:10] will return items [2] (the third item, because of 0-indexing) to [9] (the tenth item), inclusive (8 items).
- Negative indexes count from the last item backwards (thus -1 is the last item)

Python slices

Code

```
>>> mylist = ["List item 1", 2, 3.14]
>>> print(mylist[:])
['List item 1', 2, 3.1400000000000001]
>>> print(mylist[0:2])
['List item 1', 2]
>>> print(mylist[-3:-1])
['List item 1', 2]
>>> print(mylist[1:])
[2, 3.14]
>>> print(mylist[::2])
['List item 1', 3.14]
```

Python functions

Functions

- Functions are declared with the def keyword
- Optional arguments are set in the function declaration after the mandatory arguments by being assigned a default value
- For named arguments, the name of the argument is assigned a value
- Functions can return a tuple (and using tuple unpacking you can effectively return multiple values)
- Lambda functions are ad hoc functions that are comprised of a single statement
- Parameters are passed by reference, but immutable types (tuples, ints, strings, etc) cannot be changed in the caller by the callee
- This is because only the memory location of the item is passed, and binding another object to a variable discards the old one, so immutable types are replaced

Example code

Code

```
# Same as def funcvar(x): return x + 1
funcvar = lambda x: x + 1
>>> print(funcvar(1))
2

# an_int and a_string are optional, they have default values
# if one is not passed (2 and "A default string", respectively).
def passing_example(a_list, an_int=2, a_string="A default string"):
    a_list.append("A new item")
    an_int = 4
    return a_list, an_int, a_string

>>> my_list = [1, 2, 3]
>>> my_int = 10
>>> print(passing_example(my_list, my_int))
([1, 2, 3, 'A new item'], 4, "A default string")
>>> my_list
[1, 2, 3, 'A new item']
>>> my_int
10
```

Exceptions

```
def some_function():
    try:
        # Division by zero raises an exception
        10 / 0
    except ZeroDivisionError:
        print("Oops, invalid.")
    else:
        # Exception didn't occur, we're good.
        pass
    finally:
        # This is executed after the code block is run
        # and all exceptions have been handled, even
        # if a new exception is raised while handling.
        print("We're done with that.")

>>> some_function()
Oops, invalid.
We're done with that.
```

Serializing

Converting data structures to strings

```
import pickle
mylist = ["This", "is", 4, 13327]
# Open the file C:\\binary.dat for writing. The letter r before the
# filename string is used to prevent backslash escaping.
myfile = open(r"C:\\binary.dat", "wb")
pickle.dump(mylist, myfile)
myfile.close()

myfile = open(r"C:\\text.txt", "w")
myfile.write("This is a sample string")
myfile.close()

myfile = open(r"C:\\text.txt")
>>> print(myfile.read())
'This is a sample string'
myfile.close()

# Open the file for reading.
myfile = open(r"C:\\binary.dat", "rb")
loadedlist = pickle.load(myfile)
myfile.close()
>>> print(loadedlist)
['This', 'is', 4, 13327]
```

Next

Python libraries

- Numpy
- Matplotlib
- scikit-learn