10-606 Mathematical Foundations for Machine Learning

Machine Learning Department
School of Computer Science
Carnegie Mellon University

Data Types & Functions

+

Linear Algebra

Matt Gormley
Lecture 3
September 5, 2018

Q&A

PRELIMINARIES: SETS AND TYPES

Sets

Chalkboard

- Definitions: Set, element of, equality, subset
- Example: Sets of sets
- Set builder notation
- Python list/set comprehentions
- Exercise: Set builder notation
- Definitions: Union, intersection, difference, complement
- Exercise: Set complement
- Tuples and set product
- Exercise: Set product

Exercise: Set Builder Notation (I)

Write a function notdiv(S) that, given an input set S whose elements are positive integers, uses Python's set builder notation to find all the elements of S that are not divisible by 2, 3, or 5. For example: given input

$$\{1, 2, 3, 4, 5, 6, 7\}$$

the function should return

$$\{1, 7\}.$$

A couple of reminders about Python syntax: a Python function definition looks like

```
def foo(x):
    x = x*3
    return x + 2
```

(Note the lack of braces and semicolons: Python uses only whitespace to separate statements and blocks.)

And, Python writes "x mod 3" as **x % 3**.

Exercise: Set Builder Notation (II)

Write a function that, given an input set S whose elements are strings, uses Python's set builder notation to find all the elements of S that use only the letters A–M (in either upper or lower case). For example: given input

{cat, dig, dog, ant, eel}

the function should return

{dig, eel} .

Hint: a good way to test whether a string is OK is to use regular expressions. The following code

```
re.match(r'^[a-mA-M]*$', 'dig')
```

tests whether the string 'dig' is OK: it will return true since 'dig' contains only letters in A–M. (Don't forget to put "import re" at the top of your file, to load the regular expressions library.)

The Game of SET

Rules of the game:

- Each card is characterized by four features:
  1. color (red, green, blue)
  2. shape (rectangle, triangle, oval)
  3. fill (dark, light, outline)
  4. number (1, 2, 3)
- A SET consists of three cards where each feature is either *all the same* or *all different* in the group of cards
- Goal is to find as many SETs as fast as possible
- Player calls "SET", picks up the three cards, and writes down the set in **set builder notation**

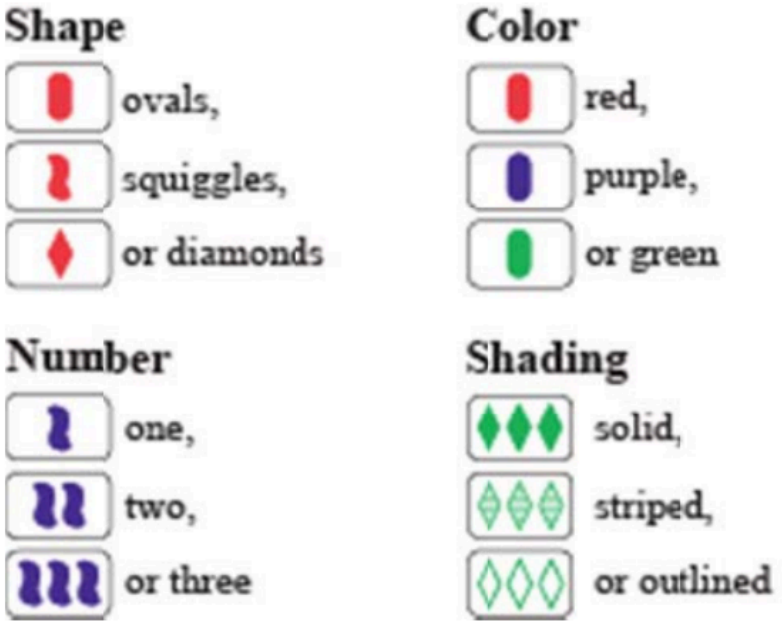


Figure from www.setgame.com

Exercise: Set Complement

Write the set complement operation using set comprehensions in Python. That is, write a function setcomp(S, U) that returns the complement of S under the universe U. Match the set-builder expressions above as closely as possible in your code.

Recall that Python uses "x in S" to test whether the element x is in the set S.

(This is not the most efficient way to implement set complement, but don't worry about that for this exercise.)

Exercise: Set Product

If X is the empty set $\emptyset$ and Y is the set $\{1, 2, 3\}$, then $X \times Y$ is

- ○ the empty set
- ○ $\{\langle \emptyset, 1 \rangle, \langle \emptyset, 2 \rangle, \langle \emptyset, 3 \rangle\}$
- ○ equal to Y

Data Types and Functions

Chalkboard

- Data types, structs, unions
- Tagged unions
- Exercise: Tagged unions
- Functions
- Anonymous functions
- Exercises: Functions

Exercise: Tagged Unions

Explain why $T_1 = \mathbb{Z} \times \{1\} \cup \mathbb{R} \times \{2\}$ is different from $T_2 = \mathbb{Z} \cup \mathbb{R}$.

Explain how T_1 implements a tagged union: that is, given an element of T_1, how do we determine the type of the tagged union? How do we extract the member of the tagged union?

Exercises: Functions in Python

Write a Python function that takes as input a function $f \in \mathbb{R} \to \mathbb{R}$ and returns another function of the same type that computes $f(x) + 1$.

What is the type of this function?

Write a Python anonymous function to represent the predicate

$$(x^4 - x^2) \geq -0.1\,.$$

Use set comprehensions to write a function **filter** that takes a set and a predicate, and returns the elements of the set for which the predicate evaluates to **true**.

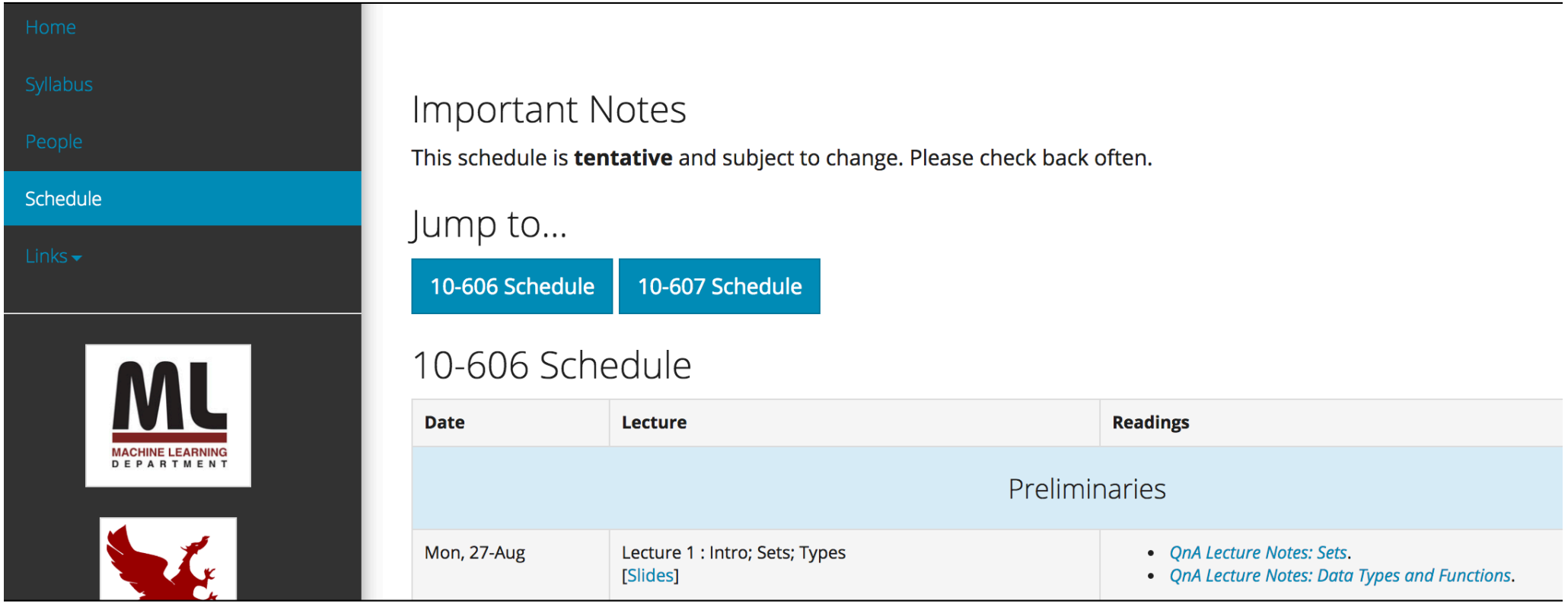
Test **filter** by applying it to the set $\{-1, -0.75, -0.5, \ldots, 0.75, 1\}$ and the predicate you defined above.

Write a Python function of the type

$$((\mathbb{R} \to \mathbb{R}) \times \mathbb{R}) \to \mathbb{R}$$

and call it.

(Note: Python doesn't easily allow you to restrict the type of the arguments to a function. So, for the exercises in this block: if we ask for a function $\mathbb{R} \to \mathbb{R}$, we mean that you are allowed to assume that the argument's type is $\mathbb{R}$. Given such an argument, the function should be guaranteed to return a value of type $\mathbb{R}$.)



See the course schedule for detailed lecture notes on Sets, Data Types, Functions.

MOTIVATION: LINEAR ALGEBRA

LINEAR ALGEBRA

Vector Spaces

Chalkboard

- Real-valued vectors of length n, R^n
- Properties of a vector space
- Inner product space
- Complete inner product space
- Other vector spaces

Linear Algebra: Notation

Chalkboard

- Real-valued matrices
- Real-valued column/row vectors
- Elements of matrices/vectors
- Dot notation for columns/rows

Linear Algebra: Operations

Chalkboard

- Vector operations
  - dot product
  - outer product
- Matrix multiplication:
  vector-vector, matrix-vector, matrix-matrix
- Transpose

Linear Algebra: Operations

Chalkboard

- Vector operations
  - dot product
  - outer product
- Matrix multiplication:
  vector-vector, matrix-vector, matrix-matrix
- Transpose
- Vector Norms
  - Euclidean norm
  - ℓ_p norms

Linear Algebra: Vector Properties

Chalkboard

- Orthogonal
- Normalized

HEBIAN LEARNING & MATRIX MEMORIES

Storing a Pattern Pair in a Matrix

In-Class Exercise

1. Given vectors:
   - stimulus: $x = [1, 0, 1, -2, 2]$
   - response: $y = [2, 1, -1]$
2. Compute their outer product: $W = xy^T$
3. Compute the predicted response: $r = Wx$
4. Does the relationship you observe here between r and y always hold?

Linear Algebra: Vector Properties

Chalkboard

- Compactly representing the average of several outer products
- Storing several pattern pairs in a single weight matrix

NEXT TIME: LINEAR ALGEBRA (PART II)