Machine Learning

Introduction to the Course

Computer & Health Informatics Department

Course Description

- Based on fundamental knowledge of computer science principles and skills, probability and statistics theory, and the theory and application of linear algebra. This course provides a broad introduction to machine learning and statistical pattern recognition.

- Machine learning uses interdisciplinary techniques such as statistics, linear algebra, optimization, and computer science to create automated systems that can sift hrough large volumes of data at high speed to make predictions or decisions without human intervention

Course Objectives

- To introduce students to the basic concepts and techniques of Machine Learning.
- To develop skills of using recent machine learning software for solving practical problems.
- To gain experience of doing independent study and research.
- Recognize the characteristics of machine learning that make it useful to real-world problems.
- Characterize machine learning algorithms as supervised, semi-supervised, and unsupervised.
- Effectively use machine learning toolboxes.
- Be able to use support vector machines.

Course Outlines

- Introduction to the course
- Basics of Machine learning
- Introduction to Python
- Types and Methods of ML
- Python - NumPy
- Understanding Data with Statistics
- Understanding Data with Visualization
- Preparing Data (Pre-processing)

Course Outlines

- Feature Selection Techniques
- ML Algorithms – Classification
- Support Vector Machine(SVM)
- Decision Tree
- Naïve Bayes Algorithm
- Random Forest
- ML Algorithms – Regression

Course Assessment

- Mid Exam 20%
- Practical Exam 20%
- Assignment / projects 10%
- Final Exam 50%

Teaching Resources

- A Course in Machine Learning, Hal Daumé III
- Artificial Intelligence: A Modern Approach (3rd Edition), Stuart Russell and Peter Norvig. Prentice Hall, 2009

Python Resources

- Python/numpy Tutorial.
- scikit-learn: Machine learning in Python

Introduction to ML

- Machine learning is about extracting knowledge from data. It is a research field at the intersection of statistics, artificial intelligence, and computer science and is also known as predictive analytics or statistical learning.

- The application of machine learning methods has in recent years become ubiquitous in everyday life.

Introduction to Python

- Python has become the lingua franca for many data science applications. It combines the power of general-purpose programming languages with the ease of use of domain-specific scripting languages like MATLAB or R. Python has libraries for data loading, visualization, statistics, natural language processing, image processing, and more. This vast toolbox provides data scientists with a large array of general- and special-purpose functionality.

What is Python?

- Python is a cross-platform programming language, meaning, it runs on multiple platforms like Windows, Mac OS X, Linux, Unix and has even been ported to the Java and .NET virtual machines. It is free and open source.

- To install Python download the latest version of Python from https://www.python.org/downloads/ and install it

Starting The Interpreter

After installation, the python interpreter lives in the installed directory.

in Windows: C:\PythonXX, where the 'X' denotes the version number there are various ways to start Python:

1. **Immediate mode:** Typing python in the command line will invoke the interpreter in immediate mode.
2. **Script mode:** This mode is used to execute Python program written in a file. Python scripts have the extension .py.

   For example: helloWorld.py

   To execute this file in script mode we simply write python helloWorld.py at the command prompt.
3. **Integrated Development Environment (IDE):** IDEL is a graphical user interface (GUI) that can be installed along with the Python programming language and is available from the official website.

Hello World Example

To write first Python program.
Type the following code in any text editor or an IDE

```
print("Hello world!")
```

Python Keywords

In Python, keywords are case sensitive, All the keywords except True, False and None are in lowercase and they must be written as it is. The list of all the keywords are given below.

Keywords in Python programming language

False	class	finally	is	return
None	continue	for	lambda	try
True	def	from	nonlocal	while
and	del	global	not	with
as	elif	if	or	yield
assert	else	import	pass	
break	except	in	raise	

```
>>> import keyword
>>> print(keyword.kwlist)
```

Python Identifiers

- **Rules for writing identifiers**

1. Identifiers can be a combination of letters in lowercase (a to z) or uppercase (A to Z) or digits (0 to 9) or an underscore (_). Names like myClass, var_1 and print_this_to_screen, all are valid example.
2. An identifier cannot start with a digit. 1variable is invalid, but variable1 is perfectly fine.
3. Keywords cannot be used as identifiers.
4. We cannot use special symbols like !, @, #, $, % etc. in our identifier.
5. Identifier can be of any length.

Python Statement

Instructions that a Python interpreter can execute are called statements.

Multi-line statement:

In Python, end of a statement is marked by a newline character. But we can make a statement extend over multiple lines with the line continuation character (\). For example:

```
a = 1 + 2 + 3 + \
    4 + 5 + 6 + \
    7 + 8 + 9
```

This is explicit line continuation. In Python, line continuation is implied inside parentheses ( ), brackets [ ] and braces { }.

```
a = (1 + 2 + 3 +
     4 + 5 + 6 +
     7 + 8 + 9)
colors = ['red',
          'blue',
          'green’]
```

We could also put multiple statements in a single line using semicolons, as follows

```
a = 1; b = 2; c = 3
```

Python Indentation

Most of the programming languages like C, C++, Java use braces { } to define a block of code. Python uses indentation.

A code block (body of a function, loop etc.) starts with indentation and ends with the first unindented line. The amount of indentation is up to you, but it must be consistent throughout that block.

e.g.
```
for i in range(1,11):
    print(i)
    if i == 5:
        break
```

Python Comments

In Python, we use the hash (#) symbol to start writing a comment. For example

#This is a comment

#print out Hello

print('Hello')

Multi-line comments

use hash (#) in the beginning of each line.

Another way of doing this is to use triple quotes, either … or “ “ “ .

E.g.

"""This is also a
perfect example of
multi-line comments"""

Docstring in Python

Docstring is short for documentation string.

It is a string that occurs as the first statement in a module, function, class, or method definition. We must write what a function/class does in the docstring.

Triple quotes are used while writing docstrings.

For example:

```
def double(num):
    """Function to double the value"""
    return 2*num
```

Python Variables

- A variable is a location in memory used to store some data (value).
- They are given unique names to differentiate between different memory locations.
- The rules for writing a variable name is same as the rules for writing identifiers in Python.
- We don't need to declare a variable before using it. In Python, we simply assign a value to a variable and it will exist.
- We don't even have to declare the type of the variable. This is handled internally according to the type of value we assign to the variable.

Variable assignment

- We use the assignment operator (=) to assign values to a variable. Any type of value can be assigned to any valid variable.

```
a = 5
b = 3.2
c = "Hello"
```

- Here, we have three assignment statements. 5 is an integer assigned to the variable a.

- Similarly, 3.2 is a floating point number and "Hello" is a string (sequence of characters) assigned to the variables b and c respectively.

Multiple assignments

- In Python, multiple assignments can be made in a single statement as follows:

```
a, b, c = 5, 3.2, "Hello"
```

- If we want to assign the same value to multiple variables at once, we can do this as

```
x = y = z = "same"
```

- This assigns the "same" string to all the three variables.

Data types in Python

- Every value in Python has a datatype.
- Since everything is an object in Python programming, data types are actually classes and variables are instance (object) of these classes.
- There are various data types in Python.
- Some of the important types are listed below.

Python Numbers

- Integers, floating point numbers and complex numbers falls under Python numbers category. They are defined as int, float and complex class in Python.
- We can use the type() function to know which class a variable or a value belongs to and the isinstance() function to check if an object belongs to a particular class.

```
a = 5
print(a, "is of type", type(a))
a = 2.0
print(a, "is of type", type(a))
a = 1+2j
print(a, "is complex number?", is instance(1+2j,complex))
```

Python Numbers

- Integers can be of any length, it is only limited by the memory available.
- Complex numbers are written in the form, x + yj, where x is the real part and y is the imaginary part. Here are some examples.

```
>>> a = 12345678901234567890
>>> a
12345678901234567890
>>> b = 0.12345678901234567890
>>> b
0.1234567890123457
>>> c = 1+2j
>>> c
(1+2j)
```

- Notice that the float variable b got truncated

The End
?