DATS200
Functional Methods and Coding
Spring 2020

Instructor: Marvine Hamner D.Sc.
Email: mhamner@apus.edu
Office Hours: **By Appointment**
Live Sessions: As scheduled in MyClassroom

Course Description

This course provides the student with the basic knowledge and skills to handle and analyze data using a variety of methods, as well as a variety of programming languages and tools. Students are introduced to current industry standard data analysis packages and tools such as those in R/RStudio, Matlab/Octave, SAS or SPSS. Depending on current industry standards, the student will be provided with the opportunity to develop knowledge and skills in programming environments such as R, Octave, and Python. In addition, students are introduced to data analysis packages and tools such as those in various scripting languages, SQL, Java/NetBeans, JavaScript or Julia.

Course Objectives

At the conclusion of this course students will be able to:

- Describe the roots of modern functional programming that relate to the analysis of data.
- Be able to differentiate a variety of programming and scripting languages by comparing and contrasting their relative advantages and disadvantages in data science.
- Given a specified research problem be able to:
    - acquire the required data,
    - determine the best analysis tools, techniques and methods,
    - complete a data analysis, and draft a report of the results of that analysis.

Textbook (Required)

This course is a "lab" course that will be taught through the use of videos, course notes and a variety of articles.

Textbook (Optional)

Optional reference books and textbooks will be suggested throughout the course.

Software (Required)

1. Git/Github: sign up for an account at: https://github.com/
2. R and RStudio available at: https://www.r-project.org/ and https://www.rstudio.com/
3. Anaconda available at: https://www.anaconda.com/
4. Octave available at: https://www.gnu.org/software/octave/

Software (Optional)

1. Node.JS available at: https://nodejs.org/en/
2. Cygwin available at: https://www.cygwin.com/ NOTE: Cygwin is not a simple installation. You may want to wait to do this during a live session.
3. SQL available as MySQL at: https://dev.mysql.com/downloads/mysql/ to install MySQL on a Mac see the notes at: https://dev.mysql.com/doc/mysql-osx-excerpt/5.7/en/osx-installation.html
4. Python 3.X.X available at: https://www.python.org/
5. Ruby at https://www.ruby-lang.org/en/
6. Java (for developers): First, read the notes at: http://www.oracle.com/technetwork/topics/newtojava/learn-141096.html# and then go to http://www.oracle.com/technetwork/topics/newtojava/learn-141096.html# and find the bundle/cobundle you want to install. If you use Windows you many just want to download the cobundle JDK 8 with NetBeans. If you use a Mac you may want to read the notes at: https://netbeans.org/community/releases/82/install.html and follow the instructions (note that either way netbeans requires JDK 8).
7. Julia available at: https://julialang.org/downloads/

Homework

A variety of weekly exercises (Knowledge Checks) will be used to reinforce the material covered in this course. These exercises will be graded and will help with completion of assigned, graded laboratories and the final exam.

Grading

- Exams 20%
- Laboratories 60%
- Discussions 18%
- Homework 2%

Course Conduct

A few rules will help us to get the most of our investment in **DATS200**:

- Sakai is going to be our platform for almost all course activities.
- In addition, we will have at least one (optional) **live session** on weekly basis.
- **Attending live sessions** and actively participating in the discussions is highly recommended as it is where you can get help with the basic concept(s) of each module, where assignments are discussed and your questions answered.
- I anticipate that you will to budget need 3 to 4 hours for **each set of readings and solving module assignments**.
- **You are responsible for completing all the readings**, even if the material is not explicitly covered in an assignment. It will be included on a lab or a quiz. You should read the class materials prior to a live session, and be prepared to discuss and ask questions about the readings and assignments.

- You should also **re-read the material** after a live session as not every topic will be covered during that session. You may need to read passages in the material several times to gain clarity. Also, taking notes on the material you are reading and reflecting on the reading and these notes will help you better understand the issues, concepts and techniques that are being presented.
- All work must be completed and turned-in on or before the due date. Late means after the due date posted in Sakai.
- Your work should be properly referenced and adhere to standards of both academic integrity and proper form. Generally, I prefer the APA style (see http://www.apa.org/). However, you can use any style as long as you consistently use the same style within any single work.
- All class credit-related electronic mail should be done using the course sites message system. By 'credit-related' I mean all work to be evaluated for credit. Any work submitted through a different mail system might not be received and accepted.
- All activities will be assigned individually unless mentioned in the assignment.
- Students who participate in University-sanctioned events (such as clubs or athletics) must make prior arrangements and give the instructor ample notice.

DATS200 Course Outline – 32 Modules

Week 1: Introduction and Background

Learning Outcomes:

At completion of the modules this week students will be able to:

1. Describe different types of computer hardware, e.g. super-computers, mainframe computers, personal computers, etc.
2. Describe types of software, e.g. firmware, middleware, application programming interfaces, etc.
3. Describe a complete computing environment.
4. Discuss API's and explain why they are important in data science.
5. Describe how data are acquired to solve a specific problem or address a specific challenge.

   Module 1: Introduction and Hardware

   Module 2: Software

   Module 3: What's in the middle? (API's,…)

   Module 4: What's important? (data, analyses, using data to form information as the basis for actionable decisions)

Laboratory 1: Git/Github

Week 2: Introduction to R and RStudio

Learning Outcomes:

At completion of the modules this week students will be able to:

1. Describe different types of programming languages and different programming languages.
2. Describe development environments and differences between development environments for different programming languages.
3. Execute the R programming language in RStudio.
4. Interpret simple R code snippets and functions.

   Module 5: Using programming languages and codes in data science (What are IDE's?, Examples)

   Module 6: Intro to R and RStudio (basic commands, loading data, graphics,,…)

   Module 7: Working with data in R (indexing data, more on graphics,…)

   Module 8: Functions in R

Week 3: Using R in Data Science Analyses

At completion of the modules this week students will be able to:

1. Conduct simple data analyses using the R programming language in RStudio, descriptive and predictive.
2. Compare and contrast the R programming language with JAVA and JavaScript.

   Module 9: Using R to conduct data science analyses (descriptive)

   Module 10: Using R to conduct data science analyses (predictive)

   Module 11: Using R… Part 2

   Module 12: Comparing R to other coding/programming languages; Java and JavaScript

Week 4: Introduction to Python

At completion of the modules this week students will be able to:

1. Execute Python in the Spyder development environment.
2. Interpret simple Python code snippets and functions.

   Module 13: Intro to Python and Spyder (Anaconda) and Object-Oriented Programming

   Module 14: Python (basic commands – similarities and differences between codes)

   Module 15: Structured types (strings, tuples, ranges, and lists)

   Module 16: Functions in Python

Week 5: Using Python in Data Science Analyses

At completion of the modules this week students will be able to:

1. Conduct simple data analyses using the Python programming language, descriptive and predictive.
2. Compare and contrast the Python programming language with R, JAVA and JavaScript.

   Module 17: Python packages

   Module 18: Using Python to conduct data science analyses (descriptive)

   Module 19: Using Python to conduct data science analyses (predictive)

   Module 20: Comparing Python to other coding/programming languages

Laboratory 2: Financial (Stock) Analyses using R and Python

Week 6: Introduction to Matlab/Octave

At completion of the modules this week students will be able to:

1. Execute Octave in the Octave development environment.
2. Interpret simple Octave code snippets and functions.

   Module 21: Matlab/Octave (basic commands – similarities and differences between codes)

   Module 22: Matlab/Octave data types

   Module 23: Data types in Matlab/Octave

   Module 24: Functions in Matlab/Octave

Week 7: Using Matlab/Octave in Data Science Analyses

At completion of the modules this week students will be able to:

1. Conduct simple data analyses using the Octave programming language, descriptive and predictive.
2. Compare and contrast the Octave programming language with Python, R, JAVA and JavaScript.

   Module 25: Using Matlab/Octave in Analyses (descriptive and predictive)

   Module 26: Using the strengths of Matlab/Octave in analyses, e.g. evaluating data for periodicity, etc.

   Module 27: Data analyses versus engineering/scientific analyses
   https://www.mathworks.com/solutions/data-science.html

   Module 28: Comparing Matlab/Octave to other coding/programming languages

Laboratory 3: Using Matlab/Octave to evaluate fluctuations and/or periodicity in data

Week 8: Wrapping it all up

At completion of the modules this week students will be able to:

1. Describe different programming paradigms.

2. Describe different levels of flow control.
3. Evaluate simple programming requirements and determine the best programming language to use to satisfy those requirements.

   Module 29: Programming paradigms – declarative, imperative, Interpreted, and structured (object-oriented) (“compiled” languages versus “scripting” languages versus “interpreted” languages) – and flow control

   Module 30: Course review

   Module 31: Practice Final Exam

Module 32: Final Exam