Data Analytics - Unit Objectives & FAQs

Students can use this document as a reference for the main topics of each unit, and to get an understanding of the skills they should expect to learn by the end of each week. *This document is intended for enrolled students, and contains confidential and proprietary information. Do not forward.*

TABLE OF CONTENTS

Unit 1.1 - 1.3: Excel

Unit Objectives:

By the end of this unit, you will:

- Gain an understanding of the course structure and general direction of the program
- Be exposed to high-level analytic strategies and tools
- Feel fully proficient in basic Excel navigation and functionality
- Gain familiarity with the value of Pivot Tables and the steps for their utilization
- Gain comfort utilizing VLookups and HLookups
- Understand how to implement conditional formatting based on logical rules
- Create, modify, and stylize basic charts from scratch using Microsoft Excel
- Feel comfortable creating scatter plots and trend lines
- Have a firm understanding of how to create charts that contain filtered data
- Know how to create regressions and calculate moving averages using Excel

Unit 2.1 - 2.3: VBA

Unit Objectives:

By the end of this unit, you will…

- Understand the fundamental building blocks of all programming languages: variables, arrays, conditionals, loops, and functions.
- Create simple VBA macros to trigger pop ups and change cell values.
- Gain practice in writing VBA subroutines that utilize variables and conditionals.

- Begin to develop essential coding skills of syntax recollection, pattern recognition, problem decomposition, and debugging.
- Understand the basic syntax of a VBA for loop.
- Understand how to utilize for-loops in conjunction with conditionals to direct logic flow.
- Understand the value of a nested for-loop and gain basic proficiency in their use.
- Refine fundamental coding skills (syntax recollection, pattern recognition, problem decomposition, and debugging).
- Be comfortable formatting spreadsheets using VBA code.
- Understand how to loop through a table using VBA code and check for changes in values.

Unit 3.1 - 3.3: Python (& Git)

Unit Objectives:

By the end of this unit, you will…

- Be able to navigate the desktop via the terminal.
- Create Python scripts and run them in the terminal.
- Begin to understand programming concepts in Python.
- Feel confident reading data into Python from CSV files.
- Feel confident writing data from Python into CSV files.
- Know how to zip two lists together and when this is helpful.
- Have a firm understanding on how to create and use Python functions.
- Be able to create and use Python dictionaries.
- Be able to read data in from a dictionary.
- Be able to add, commit, and push code up to GitHub from the command line.
- Have a firm understanding of coding logic and reasoning.

Frequently Asked Questions:

What's up with this crazy indentation?

With Python, indentation is more than just organization and readability. Python's functionality actually depends on proper indentation!

In this snippet, we're using indentation to tell our code where our for loops begin and end. In Python, indenting creates blocks of code that work together. Similarly, indenting backwards tells the program when to end a loop.

```
for x in range(10):
    print(x)
for x in range(20, 30):
    print(x)
```

The code you will write in Python will eventually be seen by someone else. Focusing on organization and readability is important because you want colleagues to be able to read your code. If your code is poorly organized it will be difficult to read later on.

4.1 - 4.3 Pandas

Unit Objectives:

- Be able to serve Jupyter notebook files from local directories and connect to their development environment.
- Be able to create Pandas DataFrames from scratch.
- Understand how to run functions on Pandas DataFrames.
- Know how to read/write DataFrames from/to CSV files using Pandas.
- Understand how to navigate through DataFrames using Loc and Iloc.
- Understand how to filter and slice Pandas DataFrames.
- Understand how to create and access Pandas GroupBy objects.
- Understand how to sort DataFrames.
- Know how to merge DataFrames together whilst understanding the differences between inner, outer, left, and right merges.
- Be able to slice data using the cut() method and create new values based upon a series of bins.
- Feel more confident with fixing Python/Pandas bugs within Jupyter Notebook.
- Be able to use Google to explore additional Pandas functionality when necessary.

5.1 - 5.3 Matplotlib

Unit Objectives:

By the end of this unit, you will:

- Understand Matplotlib's pyplot interface.
- Be able to create line; bar; scatter; and pie charts.
- Be familiar with basic plot configuration options, such as `xlim` and `ylim`.
- Feel comfortable creating plots using the `DataFrame.plot()` method.
- Understand the advantages and disadvantages of creating charts using the `DataFrame.plot()` method.
- Be able to work through a complex data set using Pandas and then chart some visualizations based upon the cleaned DataFrame.
- Be able to define mean, median, and mode, and choose which one is most appropriate to describe a given data set.
- Be able to explain the meaning of variance and standard deviation.
- Be able to describe standard error and the difference between a sample and a population.
- Be able to add error bars to plots.
- Be able to fit lines to data.

Frequently Asked Questions:

I'm having trouble picking up on this new syntax…
Luckily, there is documentation for that! Whenever we work with a library, we are using code that is already pre-written. We always need to reference the documentation when using a new library. Think of documentation as a set of detailed instructions. Without looking at the instructions, you will never know how to use a new piece of technology.

We recognize this may be your first time looking at a library that is less semantic, or readable, but this will be a constant for the future. You will see a lot of new technology throughout the course, and each time you will need to dive into the documentation. The documentation quality and organization will be different for each new technology, so learning how to read general documentation is a skill in itself.

6.1 - 6.3 Python APIs
Unit Objectives:

- Be able to make GET requests with `requests`.
- Be able to convert JSON into a Python dictionary.
- Read and apply API documentation.
- Sign up for and use an API key.
- Create applications from scratch using nothing but knowledge of Python and an API documentation.
- Load JSON from API responses into a Pandas DataFrame.
- Be able to use `try` and `except` blocks to handle errors.
- Successfully use the Google Maps and Places API to obtain information about geographic areas.
- Understand how to use the Census API wrapper.
- Understand the concept of rate limits and the importance of creating "test cases" prior to running large scripts.
- Have a firmer understanding of how to dissect new API documentation.

Where can I find more information on how to use these APIs?
Each API is a pre-written set of code that helps you interact with somebody else's information. Every API will come with its own set of documentation, and much like with a library, you will have to dive into it! There should be a developer section on each site that will explain the service and any potential limitations.

Any time we work with an API, it's a good idea to have the documentation ready and refer to it frequently. This is all part of the process when it comes to picking up and familiarizing yourself with new technologies.

7.1 - 8.3 Project 1
Unit Objectives:

- Students will be able to articulate the requirements for Project 1.
- Students will be able to draw and interpret diagrams of Git branching workflows.
- Students will be able to create new branches with Git.
- Students will be able to push local branches to GitHub.
- Students will be able to pull a branch from GitHub.
- Students will be able to merge branches with Git.
- Students will be able to open, review, and merge PRs with GitHub.
- Students will resolve merge conflicts in their working copy.
- Students will push branches to GitHub.
- Students will be able to open a PR against a given branch.

9.1 - 9.3 SQL

Unit Objectives:

- Create a localhost connection to a PostgreSQL server and have successfully connect to it.
- Create, use, and populate a SQL database with data.
- Create, populate, and select data from a SQL table.
- Import large CSV datasets into pgAdmin.
- Use pgAdmin to select specific rows/columns of data out from a table.
- Understand the different kinds of joins and how to use them to create new tables in pgAdmin.
- Solidify the foundations of writing basic- to intermediate-level SQL statements.
- Develop an introductory understanding of table design and database management.

10.1 - 10.3 Advanced Data Storage and Retrieval

Unit Objectives:

- Connect to a SQL database using SQLAlchemy.
- Perform basic SQL queries using engine.execute().
- Create Python classes and objects.
- Create, read, update, and delete data from a SQL database using SQLAlchemy's ORM.
- Reflect existing databases.
- Use the SQLAlchemy ORM to create classes that model tables.
- Use the ORM define relationships and foreign key constraints.
- Use joins to query related data.
- Extract query variable path values from GET requests.
- Use variable paths to execute database queries on behalf of the client.
- Return JSONified query results from API endpoints.

Frequently Asked Questions:

What is SQLite and why use it?

SQLite is a dialect for using an SQL database. The syntax is very similar to the PostgreSQL syntax, with the main difference between the two being that SQLite is entirely serverless. SQLite allows for the power of database storage but with a tiny footprint. SQLite is also conveniently part of the standard Python library, so no additional setup is needed to get it up and running.

SQLite is not comparable with PostgreSQL or any other SQL language, rather it focuses on providing local data storage for individual applications.

What does ORM mean?
ORM stands for Object-Relational Mapper. Object-relational mapping allows us to write SQL queries using the object-oriented paradigm of the language with which you prefer to work. In other words, for our purposes in class, it allows us to interact with our database using Python.

How are classes relevant?
As you already know, Python is an "Object-Oriented Programming (OOP) language", which means that it is highly concerned with organization and reusability. Classes are crucial to OOP in that they allow us to group related things and keep them together.

A class is essentially a template that allows us to create objects, which have variables and behaviors associated with them. It helps streamline our code and create efficiencies whenever something needs to be used various times.

11.1 - 11.3 Web
Unit Objectives:

- Gain a high-level understanding of HTML, CSS, and JavaScript and what their roles are when creating websites.
- Understand the basic parts of an HTML web page and how to create one from scratch.
- Learn to cover and utilize some of the most common HTML tags and selectors.
- Understand how to deploy HTML web pages to the internet using GitHub Pages.
- Understand the basics of CSS styling.
- Position HTML elements on a webpage using CSS.
- Be able to discuss media queries, the technology that is used to create the responsive Bootstrap grid.
- Understand the Bootstrap Grid and discover how to utilize it to position the elements on the page.
- Discover how to quickly and easily build web pages using pre-built Bootstrap components.

12.1 - 12.3 Web Scraping and Document Databases

Unit Objectives:

- Create and connect to local MongoDB databases.
- Create, read, update, and delete MongoDB documents using the Mongo Shell.

- Create simple Python applications that connect to and modify MongoDB databases using the PyMongo library.
- Use BeautifulSoup to scrape their own data from the web.
- Save the results of web scraping into MongoDB.
- Use BeautifulSoup to scrape data.
- Use PyMongo to save data to a Mongo database.

13.1-13.3 ETL Project

Unit Objectives:
By the end of this unit, you will:

- Deliver a finalized group project.
- Understand the benefits and challenges of the ETL process.

14.1 - 14.3 Introduction to JavaScript
Unit Objectives:

- Understand JS fundamentals: arrays, conditionals, loops, functions, objects.
- Understand functional programming with map, forEach.
- Work with common data structures.
- Be introduced to data driven documents (d3.js).
- Understand how to select elements using d3.select.
- Use d3 for basic DOM manipulation.
- Understand how to use callbacks.
- Understand the structure of html tables.
- Populate a table using static data structures.
- Understand events.
- Use d3 to attach events to DOM elements.
- Dynamically manipulate the DOM through events.
- Filter data with JavaScript.

Frequently Asked Questions:
Oh no! More new syntax that I don't understand!?
Don't be afraid! Learning new syntax can certainly feel like learning a whole new language. And that's what we're doing! We don't expect you to be JavaScript masters! All of this information is iterative and your skills will only improve with time and practice. If it becomes overwhelming take advantage of office hours to spend time with the instructional staff to help get you over the hill. Please don't be afraid to reach out for help! We are all here for your success.

15.1 - 15.3 Interactive Visualizations and Dashboards
Unit Objectives:

- Use Plotly to create the fundamental charts: Box, scatter, bar, pie, and line plots.
- Use the Plotly `layout` object to customize the appearance of their charts.

- Annotate charts with labels; text; and hover info.
- Create and manipulate advanced Plotly charts.
- Create bubble charts to visualize three-dimensional data.

16.1 - 16.3 D3

Unit Objectives:

- Gain a high-level understanding of SVG elements and how to append/modify them using D3.
- Understand how to bind data to SVG elements using D3 so as to create basic bar charts from scratch.
- Create a bar chart with axes using D3 so as to visualize data.
- Create different types of charts and graphs using D3.
- Cover scales in greater depth.
- Plot multiple columns from a dataset, either simultaneously or in alternation.
- Gain a better understanding of reusable code.

17.1 - 17.3 Leaflet.js & GeoJSON

Unit Objectives:

- Understand the benefits that visualizing data with maps can provide.
- Learn the basics of creating maps and plotting data with the Leaflet.js library.
- Gain an understanding of the GeoJSON format.
- Understand the concept of layers and layer controls and how we can use them to add interactivity to our maps.
- Gain a firm grasp of mapping with GeoJSON.
- Learn about and practice using Leaflet plugins and third-party libraries.
- Learn how different maps can effectively visualize different datasets.
- Gain a Leaflet mastery by completing an in-class project.
- Learn the basics of creating maps with CARTO, including writing custom CSS and SQL queries to style and filter data, while also incorporating multiple data sets within the same map.
- Understand how different types of maps are better for visualizing different datasets.

18.1 - 19.3 R & Project 2

Unit Objectives:

- Learn the basics of R syntax.
- Learn the fundamental R data types.
- Gain familiarity with c.
- Learn how to create tibbles.
- manipulate data in tibbles.
- Compare and contrast the features of Python and R.
- Load data into tibbles.
- Use the pipe operator to sequentialize operations.
- Create tibbles.

- Manipulate data in tibbles.

20.1 - 20.3 Tableau

Unit Objectives:

- Use Tableau to rapidly manipulate tables of data and create visualizations using a drag-and-drop style interface.
- Connect various data formats such as CSV and Excel Workbooks to Tableau.
- Perform exploratory data analysis using Tableau.
- Create groups and sets.
- Create maps and use built-in U.S. Census data.
- Create custom calculations.
- Understand what LOD calculations entail.

Why are we using the public version and not student?

Typically, you'll find that a lot of platforms that offer student discounts or trials do not extend this to students of continuing education programs or students without university email addresses. Please ask your SSM about your university policies on this.

What is the difference between public and pro?

Tableau Public is a free version of the software that provides all the same tools as Pro. However, because it's free, it requires all data that is input to be freely accessible to everyone, and has a limit of 1 million rows. This version should not be used if you're dealing with any sensitive or privileged datasets. This version also does not allow saving to your machine and you must save to their server.

21.1 - 21.3 Machine Learning

Unit Objectives:

- Calculate and apply regression analysis to datasets.
- Understand the difference between linear and non-linear data.
- Understand how to quantify and validate linear models.
- Understand how to apply scaling and normalization as part of the data preprocessing step in machine learning.
- Understand how to calculate and apply the fundamental classification algorithms: logistic regression, SVM, KNN, decision trees, and random forests.
- Understand how to quantify and validate classification models including calculating a classification report.
- Understand how to apply `GridSearchCV` to hyper tune model parameters.
- Understand unsupervised learning and how to apply the k-means algorithm.
- Articulate specific problems on which neural nets perform well.
- Use sklearn's to build and train a deep neural network.
- Use Keras to build and train a deep neural network.

22.1 - 22.3 Big Data

Unit Objectives:

- Identify the pieces of the Hadoop ecosystem.
- Identify the differences and similarities between Hadoop and Spark.
- Write MapReduce jobs locally with MRjob.
- Manipulate data using PySpark dataframes.
- Explain why NLP is necessary in a big data toolkit.
- Apply transformations resulting from NLP data processing to PySpark dataframes.
- Explain and utilize PySpark text processing methods like tokenization, stop words, n-grams, term and document frequency.
- Utilize a NLP data processing pipeline to create a spam filter.