The Data Mine

The Data Mine is a supportive learning community where undergraduate students from any major, in any year, can learn and apply data science skills to real-world data and problems. Students will have hands-on experience with computational tools for representing, extracting, manipulating, interpreting, transforming, and visualizing data, especially big data sets, and in effectively communicating insights about data. Topics include: the R environment, Python, visualizing data, UNIX, bash, regular expressions, SQL, XML and scraping data from the internet, as well as selected advanced topics, as time permits.

https://datamine.purdue.edu

Contacts (datamine@purdue.edu works for all):

- Dr. Mark Daniel Ward, Director, mdw@purdue.edu
- Ms. Ellen Gundlach, Managing Director, gundlach@purdue.edu
- Mrs. Maggie Betz, Corporate Partners Senior Manager, betz@purdue.edu
- Mr. Kevin Amstutz, Senior Data Scientist and Instruction Specialist, kamstut@purdue.edu

There are 2 courses associated with The Data Mine without connections to other departments, which may qualify as technical electives. These courses do count toward Purdue's Applied Data Science Certificate: https://catalog.purdue.edu/preview_program.php?catoid=13&poid=17129 They are offered both Fall and Spring semesters.

- STAT 19000/29000/39000 1-credit data science seminar with weekly big data projects on the Scholar research computing cluster (taken by all students in The Data Mine)
- STAT 19000/29000/39000 3-credit Corporate Partners research experience (taken by most students in The Data Mine). There are 25 different corporate partner data science projects in 2020-21. Students work in teams, consulting weekly with corporate mentors. Regular progress reports are submitted, and the final team project is presented at a symposium.

The overarching learning outcomes for the program are:

1. Discover data science and professional development opportunities in order to prepare for a career.
2. Explain the difference between research computing and basic personal computing data science capabilities in order to know which system is appropriate for a data science project.
3. Design efficient search strategies in order to acquire new data science skills.
4. Devise the most appropriate data science strategy in order to answer a research question.
5. Apply data science techniques in order to answer a research question about a big data set.

A variety of modules will be selected from this list, depending on the student's level and which corporate partner project they work with:

Modules	Units	Learning objectives
UNIX	1. Getting acquainted with UNIX in Scholar	1.1 Accessing the Scholar cluster using ThinLinc.
		1.2 Distinguish differences in /home, /scratch, and /class.
		1.3 Navigating UNIX via a terminal: ls, pwd, cd, ., .., ~, etc.
		1.4 Analyzing file in a UNIX filesystem: wc, du, cat, head, tail, etc.
		1.5 Creating and destroying files and folder in UNIX: scp, rm, touch, cp, mv, mkdir, rmdir, etc.
		1.6 Utilize other Scholar resources: rstudio.scholar.rcac.purdue.edu, notebook.scholar.rcac.purdue.edu, desktop.scholar.rcac.purdue.edu, etc.
	2. UNIX proficiency	2.1 Use grep to search files effectively.
		2.2 Use diff to quickly understand differences between files.
		2.3 Use cut to section off data from the command line.
		2.4 Use man to read and learn about UNIX utilities.
		2.5 Use piping to string UNIX commands together.
		2.6 Use awk for data extraction, and preprocessing.
		2.7 Use sed to parse and transform text.
		2.8 Create bash scripts to automate a process or processes.
Python	1. Introduction to Python	1.1 Installing Python and setting up a working environment.
		1.2 List the differences between lists & tuples and when to use each.
		1.3 Explain what is a dict and why it is useful.
		1.4 Demonstrate a working knowledge of control flow in python: if/else statements, while loops, for loops, etc.
		1.5 Comprehend what a function is, and the components of a function in Python.
	2. Scientific computing part I	2.1 Distinguish the differences between numpy, scipy, and pandas.
		2.2 Use numpy, scipy, and pandas to solve a variety of data-driven problems.

		2.3 Demonstrate the ability to read and write data of various formats using various packages.
		2.4 Create a reproducible analysis using Jupyter Notebooks.
	3. Python: imports, scripts, modules, & classes	3.1 Given a python directory structure, demonstrate the ability to import functions.
		3.2 Write a python script that accepts user inputs and returns something useful.
		3.3 Explain the basics of object oriented programming, and what a class is. Provide examples.
		3.4 Use classes to solve a data-driven problem. Explain why classes are an appropriate tool to use.
	4. Scraping data with python	4.1 Use the requests package to scrape a web page.
		4.2 Use the beatifulsoup4 package to filter and parse data from a scraped web page.
		4.3 Demonstrate the ability to analyze a web page's structure and use the structure to crawl and scrape other web pages.
	5. Scientific computing part II	5.1 Distinguish the differences between previously explored scientific computing packages and pytorch+tensorflow.
		5.2 Briefly explain the origins and objectives behind both tensorflow and pytorch.
		5.3 Compare and contrast tensorflow and pytorch solutions to the same problem(s).
		5.4 Use tensorflow to solve a real-world problem.
		5.5 Use pytorch to solve a real-world problem.
	6. SQL in Python	6.1 Demonstrate the ability to interact with popular database management systems within Python.
		6.2 Solve data-drive problems combining SQL databases and programming abilities in Python.
R	1. Introduction to R	1.1 Install R and setting up a working environment.
		1.2 List the differences between lists, vectors, factors, and data.frames, and when to use each.
		1.3 Explain and demonstrate: positional, named, and logical indexing.
		1.4 Demonstrate a working knowledge of control flow in r: if/else statements, while loops, etc.

		1.5 Comprehend what a function is, and the components of a function in R.
		1.6 Read and write basic (csv) data.
	2. Apply functions	2.1 Explain what vectorized functions are.
		2.2 Demonstrate how apply functions are generally faster than using loops.
		2.3 Briefly explain the differences between the various apply functions.
		2.4 Utilize each apply function in order to solve a data-driven problem.
		2.5 Demonstrate the ability to use nested apply functions to solve a data-driven problem.
	3. Core R behavior	3.1 Demonstrate the ability to use the following functions to solve data-driven problem(s): mean, var, table, cut, paste, rep, seq, sort, order, length, unique, etc.
		3.2 Convert strings to dates, and format dates in a variety of ways.
		3.3 Explain and demonstrate how R handles missing data: NA, NaN, NULL, etc.
		3.4 Explain what "recycling" is in R and predict behavior of provided statements.
		3.5 Gain proficiency using split, merge, and subset.
		3.6 Demonstrate the ability to install and use packages.
	4. Scientific computing part I: tidyverse	4.1 Explain the differences between regular data frames and tibbles.
		4.2 Use mutate, pivot, unite, filter, and arrange to wrangle data and solve data-driven problems.
		4.3 String multi-step procedures together using pipelines.
		4.4 Convert strings to dates, and format dates using the lubridate package.
		4.5 Combine different data using joins (left_join, right_join, semi_join, anti_join), and bind_rows.
		4.6 Manipulate text data using the stringr package.
		4.7 Group data and calculate aggregated statistics using group_by, mutate, and transform functions.
		4.8 Create a reproducible research using Rmarkdown.
		5.1 Compare and contrast data frames and data.tables.

	5. Scientific computing part II: data.table	5.2 Demonstrate the ability to subset data.tables (row, column, and both).
		5.3 Aggregate data.table's and create summary statistics for each grouping.
		5.4 Demonstrate the ability to order data tables based on different columns.
		5.5 Demonstrate the ability to chain operations in a data table.
		5.6 Perform functions on many columns using lapply functions and .SD, SDcol.
	6. Scraping data with R	6.1 Use the xml2 and XML packages to parse XML documents.
		6.2 Use the rvest package to scrape web pages.
		6.3 Use the rvest, xml2, and XML packages to filter and parse data from a scraped web page.
		6.4 Demonstrate the ability to analyze a web page's structure and use the structure to crawl and scrape other web pages.
	7. Shiny	7.1 Explain and demonstrate how to use the basic components of a shiny app (user interface, server, inputs and outputs)
		7.2 Briefly explain and demonstrate the ability to use different input and output options (checkboxInput, sliderInput, plotOutput, textOutput, etc.)
		7.3 Create a basic shiny app to solve a data-driven problem.
		7.4 Briefly explain and showcase the use of reactivity in shiny apps (reactive, observeEvent, etc.).
		7.5 Customize shiny app layout (tabPanel, sidebarPanel, etc).
	8. SQL in R	8.1 Demonstrate the ability to interact with popular database management systems within R.
		8.2 Solve data-driven problems using a combination of SQL and R.
Data Visualization	1. Basics of data visualization	1.1 Explain when each of the following graphs are appropriate to use: histogram, bar graphs, scatter plot, line plots, piecharts.
	2. R: Core plots	2.1 Demostrate the ability to create basic graphs with default settings.

		2.2 Demonstratre the ability to modify axes labels and titles.
		2.3 Incorporate legends using legend().
		2.4 Demonstrate the ability to customize a plot (color, shape/linetype).
	3. R: ggplot2	3.1 Demostrate the ability to create basic graphs with default settings.
		3.2 Demonstratre the ability to modify axes labels and titles.
		3.3 Explain and demonstrate how to use aesthetics (x, y, color, shape).
		3.4 Showcase customization by using different themes (theme_bw, theme_classic, packages ggthemr).
		3.5 Comprehend and illustrate side by side plots using facets (facet_wrap, facet_grid).
	4. Python: matplotlib	4.1 Demostrate the ability to create basic graphs with default settings.
		4.2 Demonstratre the ability to modify axes labels and title.
		4.3 Utilize the legend function to provide useful information on plots.
		4.4 Customize colors and legend names using color and label arguments.
		4.5 Create side by side plots using subplots function.
Data Formats	1. CSV/JSON/HTML	1.1 Review and summarize the most prevalent data formats, and associated terminologies: csv/tsv, json/yaml.
		1.2 Match HTML terms to sections of HTML demonstrating working knowledge.
	2. XML	2.1 Review and summarize the differences between XML and HTML/CSV.
		2.2 Match XML terms to sections of XML demonstrating working knowledge.
	3. Encoded formats	3.1 Explain what an encoded format is, give an example of an encoded format, and explain why we have encoded formats.
SQL	1. Introduction to SQL	1.1 Explain the advantages and disadvantages of using a database over a tool like a spreadsheet.

		1.2 Describe basic database concepts like: rdbms, tables, indexes, fields, query, clause.
		1.3 Compare and contrast popular database management systems like sqlite3, postgresql, mysql, etc.
		1.4 Basic clauses: select, order by, limit, desc, asc, count, where, from, etc.
	2. SQL Fundamentals	2.1 Briefly explain and demonstrate syntax rules.
		2.2 Explain difference between primary and foreign key.
		2.3 Showcase the ability to filter, alias, and write subqueries involving mainly in, not in, like, min, case.
		2.4 Briefly explain the differences between left and inner join and demonstrate the ability to use the join statements to solve a data-driven problem.
		2.5 Perform grouping and aggregate data using group by and the following functions: count, max, sum, avg, like, having. Explain when to use having, and when to use where.
		2.6 Create or insert data into a database using SQL.
Git & Github	1. Getting started with Git & Github	1.1 Explain what version control is, and why it is important.
		1.2 Briefly explain and demonstrate ability with basic git commands to initialize a repository, clone a repository, stage changes using add, and making commits using commit.
		1.3 Demonstrate the ability to fetch repository changes using fetch, pull changes from a remote repository into your local repository using pull, and push changes from your local repository to a remote repository.
		1.4 Demonstrate the ability to merge changes into a repository and resolve any conflicts.
		1.5 Explain what a Pull Request is in GitHub, and practice creating a pull request.
		1.6 Demonstrate and practice the common git cycle: create a new branch, make changes to the code base, create a pull request.
VPNs	1. Understanding and using a VPN	1.1 Articulate and describe the purpose of a VPN quickly and effectively to someone with little to no technological acumen.
		1.2 List reasons why a VPN is useful.

		1.3 Quickly describe what an internal and external network is and how VPNs fit in the picture.
		1.4 Successfully connect to the Purdue VPN using Cisco AnyConnect, and ping Scholar.
Docker	1. Introduction to Docker	1.1 Define an app or software container, and list multiple reasons why a containerization software like Docker is useful.
		1.2 Explain the difference between Docker, Dockerfiles, Docker Images.
		1.3 Pull a Docker Image using pull, and run that image using run.
	2. Docker fundamentals	2.1 Use the basic Docker instructions to write a Dockerfile that has an up-to-date R instance and a user-added R file to run.
		2.2 Build, and run the Dockerfile. Get a shell to the running image.