Introduction to Computer Science + Society: A Multidisciplinary Course for All

1. Introduction

Engineering education with a narrow focus on technology may have made sense in the past, it fails to meet the needs of 21st-century students, who will enter industries that fuse the Humanities with various forms of technology. Having a strong background only in STEM fields will not prepare tomorrow's leaders for the complex social issues they will navigate. Broad, rigorous training in the liberal arts will meaningfully complement Engineering education. Multidisciplinary training in a variety of methods of research and interpretation prepares graduates to tackle complex problems with the humility and confidence to conceptualize their investigation in a nuanced and comprehensive manner.

Computer Science encompasses far more than programming. Skuse et al. [1] observe, "Computer Science is the study of algorithms, i.e., of processes for solving problems. … Learning to program, itself, is not Computer Science any more than learning French is the study of French Literature; a computer scientist may have to learn a programming language to implement an algorithm." Thus, Computer Science and Society are essential complements of each other.

Faculty at University of British Columbia developed a CS 0.5 course [2] in addition to for non-CS majors because that population was increasingly taking CS 1 classes and having a negative experience and not meeting the learning outcomes. The new course covers 40% of the learning outcomes of CS 1 (6 modules), 2 modules of CS 0.5 content, and spends the rest of the time on a project designed for applied learning. This approach is more like a temporary fix on student experience by making it less demanding on the CS side, but the majority of CS 1 stays the same that will still cause the frustration for non-CS students.

In this paper, we present the design and development of Introduction to Computer Science + Society course that is totally different from the Computer Science 1 course offered in our school. It not only incorporates extensive training in the powerful tools of Computer Science, but also utilizes analytical frameworks of the Humanities and Social Sciences to equip students with skills needed to navigate the digital landscape of the future. Moreover, this course provides an entry point to Computer Science field, and it is designed to be flexible that students of any levels and backgrounds can have a good experience with. We also share our successes and improvements as the result of teaching the course for one semester.

The paper is organized as follows: section 2 describes the design of the course in detail; sections 3 and 4 discuss our successes and improvements respectively. We conclude our paper in section 5 with some plans for future work.

2. Course Design

This course offers an introduction to the intersections between Computer Science and Society through a survey of digital projects and techniques. The frameworks of the Humanities and Social Sciences to relevant problems or scenarios include identifying, procuring, and interpreting various types of sources and data. Computer Science tools will vary and may include textual analysis, spatial analysis, data visualization, web technologies, mobile technologies, 3D printing, and other forms of inquiry and applications relevant to themes drawn from the Humanities and Social Sciences. The theme for the first run of the course was How can the approaches of Computer Science, the Humanities, and the Social Sciences be used to address climate change? Students were introduced to the key issues and debates around climate change from a global perspective. Students learned how to conduct research, work with relevant sources and data sets, and carry out data analysis and visualization using a range of digital tools, such as Pandas [3], Jupyter Notebook [4], and ArcGIS [5].

There are three key components of this course: lectures, labs, and projects.

The course is multidisciplinary because it is co-taught by one professor in Humanities and one professor in Computer Science and the knowledge and expertise are shared between the professors. Lectures cover topics in humanities delivered by the Humanities professor, such as climate change and social inequality. Those topics make students aware of social issues that they see in real life and inspire them to do further research on those topics, especially in the final project. Lectures also cover CS topics in working with data delivered by Computer Science professor, such as data exploration, data cleaning, data analysis, and data visualization followed by live coding sections and hands on labs.

This notebook is for the interactive live coding section for the Data Exploration lecture. Students can follow the steps listed below and write code under each Task during lecture time.

Part 1: Basic Data Exploration with Pandas

1. read_csv()

In [15]:

```
# Task: read file Cal_Fire_Incidents.csv into DataFrame df
import pandas as pd
df = pd.read_csv('Cal_Fire_Incidents.csv', header = 1)
df
```

Out[15]:

	incident_name	incident_is_final	incident_date_last_update	incident_date_created	incident_administrative_unit	incident_administrative_unit_
0	Bridge Fire	True	2018-01-09T13:46:00Z	2017-10-31T11:22:00Z	Shasta-Trinity National Forest	N
1	Pala Fire	True	2020-09-16T14:07:35Z	2009-05-24T14:56:00Z	CAL FIRE San Diego Unit	N
2	River Fire	False	2013-02-28T20:00:00Z	2013-02-24T08:16:00Z	CAL FIRE San Bernardino Unit	N
3	Fawnskin Fire	True	2013-04-22T09:00:00Z	2013-04-20T17:30:00Z	San Bernardino National Forest	N
4	Gold Fire	True	2013-05-01T07:00:00Z	2013-04-30T12:59:00Z	CAL FIRE Madera-Mariposa-Merced Unit	N
...	...	...	...	...	...	
1734	Wolf Fire	True	2021-01-20T07:12:29Z	2021-01-19T11:47:00Z	Kern County Fire Department	N

Fig. 1. Data Exploration with Pandas in Jupyter Notebook

Labs are developed in Pandas and Jupyter Notebook. Pandas is a data analysis and manipulation tool built on top of the Python programming language and has been one of the most commonly used tools for Data Science and Machine learning to handle data. The main advantages of Pandas over Java are its simplicity and ease of use. Jupyter Notebook on the other hand is an open-source web-based notebook environment that allows you to write code, view it, and execute it within a single user interface, thus making it easy to make changes on the go and immediately see the impact of the changes. The usage of Pandas and Jupyter Notebook in this course is one of the uniqueness and turning point that help students with little to no coding experience to ramp up quickly and start producing meaningful results from day 1 in class. Figure 1 shows a snapshot of an interactive lecture where an instructor can code along with students step by step using Pandas in Jupyter Notebook to accomplish some tasks.

Project is the signature of this course. Labs provides students with all the tools and framework needed to work on the project. Indeed, students have used ArcGIS to make maps and the Pandas library for Python to create additional types of data visualizations. Along the way, students practice data collection, exploration, cleaning, and analysis. The final project for this course allows them to combine these skills to answer a research question of your own design. Students will work in pairs to complete this assignment. Here are the project deliverables:

- Research Plan
  The form asks the following:
  1. What topic related to climate change will you study?
  Students may study any aspect of the topic in any part of the world as long as they can find relevant data.
  2. What is the overarching research question you plan to answer?
  3. What are the smaller research questions you will answer to support your answer to the larger question?
  Example:
  Big, overarching question: How will climate change shape life for residents of New Orleans?
  • Smaller question 1: How have rising sea levels shaped life for residents of New Orleans in recent years?
  • Smaller question 2: How have residential and commercial development patterns in recent years changed the future of the city with regard to climate change?
  • Smaller question 2 is really two questions in one—one about residential development and one about commercial development.
  • You should generate a series of at least 6 data questions—you may not actually use all of them, but it will be good to have backup questions in case some others don't work out.
  4. What quantitative data sources can be used to answer questions 1 and 2 above? Identify the source you will use to answer each data question you list. You should have quantitative (numerical) data such as CSV files that can answer these questions.
  Example:
  Question 1
  • Sea level changes in the last 10 years

• Number of flood events in the area in the last 10 years
Question 2
• Residential/Commercial development patterns?

Number of new housings/neighborhoods/buildings/developments in different areas with/without potential flooding
Sample data sets
https://floodfactor.com/city/new-orleans-louisiana/2255000_fsid
https://www.rand.org/content/dam/rand/pubs/rgs_dissertations/2010/RAND_RGSD262.pdf
5. What questions can you answer about the human dimensions and social/political/economic context of your topic?
Some human experience questions are better answered through qualitative research. For instance, to capture the way some people feel about an event or trend, you may use a short story written by a resident of that area or a newspaper article about a protest, a new law, or a public vote. You should identify two human experience or context questions you want to answer and the sources you will use to answer them.

- Data visualization: Student will submit a Jupyter notebook showing your method of creating your data
  visualizations. You will also present your draft data visualizations to the class. Each student should prepare at least one figure using Pandas and one map. Your group will thus have a total of at least 4 data visualizations—at least two figures and two maps. The class will discuss the visualizations and offer feedback to help you improve them for the final submission.
- Poster: This poster will include four visualizations and explanation of the findings. It will also include your answers to questions about the human experience.
- Presentation: Groups present their results in front of the class.

3. Successes

- Multidisciplinary: The design and development of this course bring together the Humanities and Computer Science Schools in the creation of an intro class in Computer Science + Society major. This type of faculty partnership is rare at other universities and exemplifies the kind of multidisciplinary teaching and learning any university seeks to develop.
- Programming: The Pandas library for Python enables students with no background in computer programming to carry out sophisticated tasks. This exceeds all initial plans for the course and for the major in Computer Science + Society, which were predicated on all course enrollees having a co-requisite of Computer Science I. All students in the course demonstrated the ability to use Pandas to import CSV files, explore the data, clean the data, and visualize the data. Students demonstrated their knowledge of this through labs, in-class quizzes, and the final project. From the midterm student feedback, students liked that the course focused on real life issues and provided a Computer/Data Science solution to analyzing the problems. The introduction of how to analyze and visualize data provides a great way to let student learn how to present data. They also mentioned that in most courses

where they learn how to code, they never had a chance to learn how to present their data in an appealing way.

- GIS: Students were introduced to the fundamentals of mapping with GIS software. All students demonstrated the ability to create basic maps with ArcGIS, locating data files, importing them to the software, and styling the resulting map. Students showed their high interest in using GIS software and demonstrated their knowledge of this through in-class exercises and the final project.
- Humanistic Context: Students used the key aspects of the climate crisis to shape their final projects. Students demonstrated their knowledge of this through in-class exercises and on the final project.

4. Improvements

- Pacing: The structure of the course must be refined to allow pacing more suited to student learning. At the start of the semester, we could not clearly anticipate how long students would need to master each module and concept. As a result, we underestimated the amount of class time that would be needed for live coding sessions and practice activities. We will reconceptualize the pacing now that we have detailed, and first-hand knowledge of the time needed for each task. Overall, the computer programming modules took longer than we anticipated. This variation from our plan results from the fact that students were learning computer programming fundamentals alongside fundamentals of data science, we realized. As a result, this material must be spaced out more effectively through the course. This term, we taught the background material first, then transitioned to the programming material. Next time, we will begin the programming material immediately. We will then intersperse that material with that from the Humanities/Social Science context. This pacing will allow us to better ensure student learning of the programming material within the context of the humanistic framework.
- Cognitive Load: There was a very reasonable amount of work assigned. However, we anticipated that students would complete more review and study outside of class than was the case. Computer programming is inherently complex and individual learners master material at wildly different paces. To some extent this is shaped by their background with programming. This will be a factor for us to manage going forward, as the course will be populated with both first-year students and upper-level students taking the course as an elective. Our pacing and “chunking” of the material will be designed to keep a consistent level of cognitive load for learners as the course progresses.
- Humanistic Context: Two primary changes are required for students to better master the contextual material we are teaching. For one, as noted above, we will teach such content intermittently with the programming content. As one student suggested on the mid-semester course survey we conducted, providing students with a new programming concept, some related context, and then applied activities involving that concept within that context would be a great approach to integrating these diverse types of knowledge. Weaving the material together that tightly can be done, but it is complex work that requires expert knowledge of the student learning patterns at play. We are confident that we can do this next year.

Secondly, students struggled with the analytical portion of their final projects. They carried out the technical work (to varied success, as is to be expected), but their reasoning for attempting specific types of analysis were sometimes unclear or inappropriate. We can better help students to carry out this work next time if we budget additional time for such analysis in class. They will gain more practice through the shorter programming activities described above.

- Scaffolding: For the final project we will begin it earlier in the term and provide more support for learners along the way. We did create five graded assessments as part of the final project, to ensure that students received ample feedback and support. However, we underestimated the extent to which students would simply not submit these scheduled assignments or would submit incomplete or unfocused work. Our approach would work well with upper-level students. This is evidenced by the stronger performance of the upper-level students in the course. To carry out this type of independent project with first-year students, we will need to create more in-class work where we can monitor student activity. We will also need to schedule meetings with individual groups at regular intervals to assess progress and make relevant suggestions. We relied too much on the students to ask questions of us as needed. This is appropriate with upper-level students, but not with first-year students, who do not know how to ask questions in many cases or feel uncomfortable doing so.
- Content: To create curricular space for the above changes, we will either need to cut the mapping section or transition it from GIS software to something more aligned with Python. We are researching additional libraries that are sometimes used for mapping, such as Bokeh [6] and will decide based on our findings.
- Groups: The first-year students will need more specific guidelines and in-class coaching on managing group projects. While the teams of advanced students did not struggle with this, the first-year students experienced problems with uneven participation and lack of communication between group members. We will have to give more specific guidance on the roles of each group member. We will also have to carefully monitor the tasks each group member is doing, especially at the start of the assignment, to ensure that there is parity and appropriate communication.

5. Conclusion and Future Work

In summary, the design and development of this course provide a framework to integrate Computer Science/Data Science into Humanities and Social Sciences courses. This multidisciplinary course will fit students of all disciplines as a light-weight introduction to gauge student interests in the Computer Science/Data Science fields. For future classes, we are taking the steps outlined above to foster greater success for students of all levels and backgrounds. We will also collect feedback of students from different majors taking the course and data on their performance to evaluate the effectiveness of our course design.

Acknowledgement

We would like to thank the Wentworth Provost’s Office for the opportunity to hold a Bistline grant in support of the development of this course in the CSAS major. We would also like to thank the School of Computing and Data Science and the School of Sciences and Humanities for the resources to deliver this course in Fall 2021.

References

1. Gary R. Skuse, Daniel A. Walzer, Kathryn Tomasek, Douglas Baldwin and Mark Bailey, “Computer Science and the Liberal Arts: Hidden Synergies and Boundless Opportunities,” New Directions for Computing Education, eds. S.B. Fee et al., Springer, 2017, 48.
2. Dawson, J. Q., Allen, M., Campbell, A., & Valair, A. (2018). Designing an Introductory Programming Course to Improve Non-Majors’ Experiences. Proceedings of the 49th ACM Technical Symposium on Computer Science Education, 26–31. https://doi.org/10.1145/3159450.3159548
3. Pandas. (2022). [Online]. Available: https://pandas.pydata.org/
4. Jupyter Notebook. (2022). [Online]. Available: https://jupyter.org/
5. ArcGIS. (2022). [Online]. Available: https://pro.arcgis.com/
6. C. Harper. Visualizing Data with Bokeh and Pandas. Accessed: April 2022. [Online]. Available: https://programminghistorian.org/en/lessons/visualizing-with-bokeh