Data Science - Portfolio tasks

Your portfolio makes up 20% of your grade.[1] It is made up of 10 coding challenges (CC) each of which produce visualisations on your website which we will grade.

Each of the ten tasks will be graded out of 2%, with full marks for complete, 1% given for semi-complete or well-attempted, and 0% for missing or badly incomplete.

In week 10 you have two options. You can complete both, if you wish. There is no extra credit for doing this.

We **strongly advise** you to complete each task in the week it is set.

Portfolio – list of coding challenges

1

CC1. Hosting. Display charts in your own site.

Set up your own Github account and live page using GitHub pages. Make sure you understand the difference between your repo and your site. For example, my repo and web sites are:

https://github.com/RDeconomist/RDeconomist.github.io

https://rdeconomist.github.io

Add your first index.html file to your repo, and check that your live site is working.

Now add two charts using the vegaembed function.

Tips:

Chart options can be found here:

https://www.richarddavies.io/library

For a given chart spec, you can visualise it (check what it looks like) using the Vega-Lite editor:

https://vega.github.io/editor

CC2. Building. Create your own visualisations

Set up an account on the Economics Observatory Data Hub.

Build two separate charts using the “create” tool.

When you are happy with the charts, take the json code that generates them, and save this as two new .json files in your GitHub repo.

Now embed these two charts in your page.

Stock take: By this stage you should have a live web site, with four embedded charts.

CC3. Debating. Use a visualisation in economic/policy commentary

Produce two charts that support or refute (or are related to in some way) a topic discussed at the Festival of Economics.

https://www.economicsobservatory.com/festival-of-economics-2024

Your task:

- Set out (not more than 25 words) a topic that was discussed. [Hint: use an html <p> tag to insert a paragraph].
- Make two charts that support or refute it or a related to an argument on this topic. This could be two that support, two that refute, or a mixture.
- Comment (not more than 25 words) on what you find. Your commentary should be placed near the chart in a way that makes it intuitive to your site's users (i.e., above, below, or to the side).

Stock take: By this stage your web site, should have 6 embedded charts.

CC4. Replication. Re-create, then improve, someone else's chart.

Find a chart that a journalist, think tank, television channel or company has used. Your challenge this week is to replicate it, and then improve on it. Refer back to our discussion of the do's and don'ts of visualisation for guidance.

Your task:

- Crate an image file of their chart. In Windows this can be done using the "snipping tool" for example. Display this file on your page. [It could be a .jpg or .png file, for example].
- Now replicate their chart in Vega-Lite. You are free to trace (i.e. make up, manually) the data to do this. Your task is not to get the data perfect, but rather to align your axes, titles and colours to look like theirs. So, if it is the Financial Times, for example, you would need to set a pink background.
- Now improve their chart. Change the titles, axes, colours and other elements of the chart specification to make it better. Explain briefly (not more than 25 words) what you have done.

Alternative. If you cannot find a suitable chart, then use Tufte's 'worst graphic' example of a bad (i.e. very low) data-to-ink ratio. Display an image of this (see Annex), then try to (roughly) re-create it in Vega; and, finally, improve it in Vega. The result of this, as above, should be one image, and two new charts.

<u>Stock take</u>: By this stage your live web site should have 8 embedded charts, and one embedded image file.

CC5. Scraper. Implement a data scraper of your own.

This week we have discussed the simple ways of scraping data from websites. Using a Google Colab python notebook, scrape a website through one of the presented methods in class (pandas read_html or beautifulsoup). This can be any source, any data.

Then clean and normalise the data and export into TIDY (long form) format.

1. A link to the Google Collab python notebook where you conducted your data analysis (make sure you make it publicly accessible!)

2. A chart built with the data from the previous point. It is up to you what how do you reference and include the data in the Vega-lite chart specification.

3. Comment (not more than 25 words) why did you choose this site.

CC6. Loops. Build a dashboard.

Use the ONS API to batch download nine different series as JSON files using a loop.

Save these to your GitHub account, and use these (as “raw” files) to supply the data to nine (or more) charts on a theme or themes of your choice.

Notes:

- We have discussed the ONS API in class. The generic form is:
    - https://api.ons.gov.uk/timeseries/{SERIES_ID}/dataset/{DATASET_ID}/data
    - You need to provide the series and data IDs, looping over one of these.
- A working example is:
    - https://api.ons.gov.uk/timeseries/cpoa/dataset/pusf/data
- (Note: if using direct in a chart spec, you may need to use a prefix).
    - https://api.allorigins.win/raw?url=
- We have discussed using loops to batch download from APIs, using FRED. Apply these skills to the ONS API.
- Charts should be time series, and use the time series API above.
- You can cover any topic, or topics of your choice.
- Given that you are making 9 very similar charts, they should be smaller than the normal chart size you are using on your page.
- Ensure you add a link to the Google Collab python notebook where you conducted your data analysis (make sure you make it publicly accessible!)

Alternative. If your project has a related API, then you could use that API to get 9 series and build a dashboard related to your project. This could be used in your project also.

CC7. Maps. Base maps and choropleths

Produce two maps and embed them in your portfolio page. Both should be of the same country, region, area, city.

Your map must NOT be any of the examples we have used in class or on the www.rapidcharts.io/maps page.

- Map 1. Your base map. A simple map which shows the borders of sub areas. Must have the "tooltip" encoding added so that when the user hovers over the area they are told the names of the places. Example: the US map, with tooltip that reveals the name of the state.
- Map 2. A choropleth map. Use the "lookup" transform to match your map GeoJSON with another dataset, and produce a colour tinted map of this.

Comment – not more than 25 words – explain what your map shows.

This homework was discussed at length in the lecture. Code for reproducing maps shown in the lecture and class is available at:

https://github.com/RDeconomist/RDeconomist.github.io/tree/main/charts/maps

CC8. Analytics charts.

Produce two charts that use advanced analytics. This could include any two of:

Scatter, bubble, histogram of distributions, de-trended (including univariate regression) shock analysis, Diff-in-Diff, heat maps. Examples, including code, were discussed in the lecture.

Note: These charts can be related to (and used in) your project.

CC9. Big Data. Extracting a story from millions of prices.

Produce two charts using the UK price data provided in class.

Explain, no more than 50 words, what you have done.

Notes:

You can read about the price data here: https://richarddavies.io/prices/

CSVs and notes can be found HERE.

CC10 – Option 1. Machine Learning

Conduct an applied data analysis using any 2 of the 4 learnt machine learning techniques.

One should be applying a Supervised learning (Regression or Classification), and the other an Unsupervised learning (Clustering or Dimensionality reduction) method.

Make sure that you:

- Transform the input data in the standardised X matrix form for both tasks.
- Transform the target data in the standardised y vector form for the Supervised learning task.
- Use sklearn inside python to apply the machine algorithms. You may use any algorithm found in the library, not just the ones discussed in class.
- Submit a link to your Google Colab workbook. You may choose to include both tasks in a single workbook.
- Make sure your workbook has read access, and there is a clear link to it from your portfolio site.

For each task:

- Set out (not more than 25 words) a data hypothesis that you will examine.
- Conduct the data analysis in python. Make at least one chart. This can be done in python (matplotlib) or you may choose to export the data and visualise it in Vega-Lite.
- Comment (not more than 25 words) on what you find.

CC10 – Option 2. Interactive Visualisations

Produce two visualisation that have interactive elements.

The simplest interactivity—tooltip—does not count.

Examples of interactivity are discussed regularly in class and can be found on the chart library.

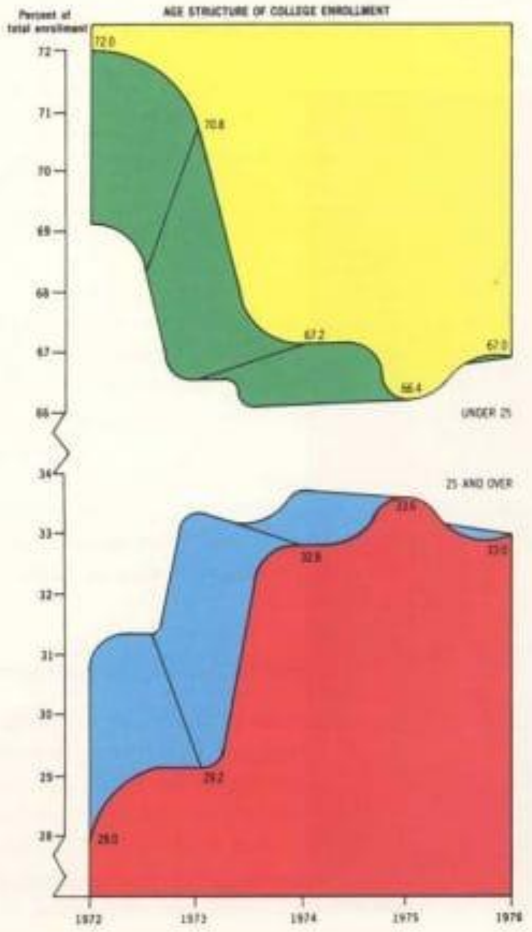
www.richarddavies.io/library

Annex

The Data-to-Ink ratio. Tufte’s worst example.

The addition of a fake perspective to the data structure clutters many graphics. This variety of chartjunk, now at high fashion in the world of Boutique Data Graphics, abounds in corporate annual reports, the phony statistical studies presented in advertisements, the mass media, and the more muddled sorts of social science research.

A series of weird three-dimensional displays appearing in the magazine *American Education* in the 1970s delighted connoisseurs of the graphically preposterous. Here five colors report, almost by happenstance, only five pieces of data (since the division within each year adds to 100 percent). This may well be the worst graphic ever to find its way into print:


AGE STRUCTURE OF COLLEGE ENROLLMENT
Percent of total enrollment
UNDER 25
25 AND OVER