DSC Capstone Sequence

Lecture 07
EDA

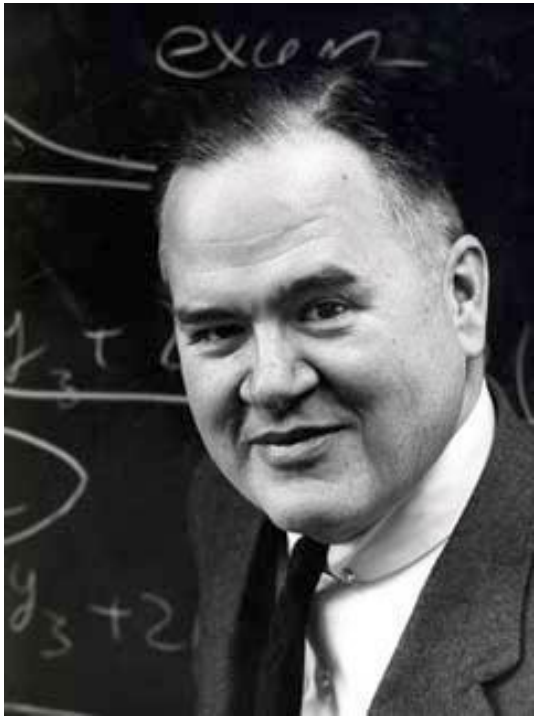
Lecture Outline

- The need for EDA
  - Why EDA (revisited)
  - Using an EDA in a focused project
- Notebooks in Data Science Software Development

The need for EDA

Tukey on Exploratory Data Analysis

“In exploratory data analysis there can be no substitute for flexibility; for adapting what is calculated—and what we hope plotted—both to the needs of the situation and the clues that the data have already provided.”



Tukey on Exploratory Data Analysis

“In exploratory data analysis there can be no substitute for flexibility; for adapting what is calculated—and what we hope plotted—both to the needs of the situation and the clues that the data have already provided.”

- EDA as detective work: meandering path that reveals story.
- Flexibility in the questions you ask and pursue
  - your questions evolve as you investigate.
- Flexibility in the code you write
  - efficient code enables faster evolution.

EDA in a Vacuum

Given a dataset and told to "do an EDA", you likely:

- Generate univariate histograms and bivariate scatterplots
- Calculate means and variance, across time, space, or other categories
- Plot (univariate and bivariate) box-plots and study percentiles
- Search for anomalies
- Search for corrupt data and work to understand how faithful the dataset is to the DGP (if you even know what it is...)

However, when you are starting with a specific question, you will likely focus your investigation quickly thereafter!

EDA's place in a DS Project

EDA occupies an interstitial space between open-ended exploration and your development of the final narrative.

It always has direction (even if it changes) and specific questions it's trying to illuminate and answer.

It occupies an important place in the *development* of the story that clarifies and answers the question at hand (i.e. the researcher uses it to tell the way).

A well structured EDA can be quickly cannibalized to support the narrative of the final project.

How an EDA is included in a project write-up

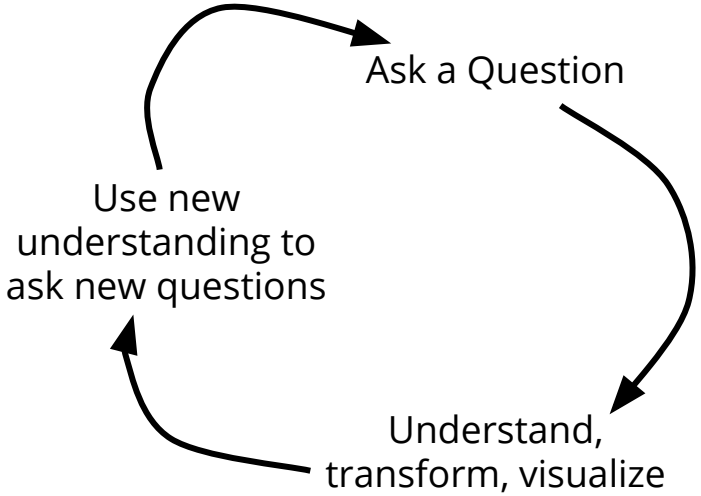
The work done in EDA provides data-driven descriptions in a report:

- Setting context for the reader. (Intro)
- Describing for the reader the quality of the observed data -- and its appropriateness to answer the question. (Intro)
  - Is the dataset representative of the population?
- Informing the next steps of development (Methods)
  - Data cleaning choices; feature/model selection; reductions and model assumptions.
  - Only those choices directly relevant to results are included!
- Understanding and drawing conclusions about (the robustness of) the result (Conclusions).

How an EDA is included in the project write-up

Each step in an investigation has questions; how does one move forward?

1. Exploration and understanding (development)
2. Transform understanding into explanation and narrative
    a. Keep the best tables and graphs for the paper!
3. Move to the next step & the next unknown...



Notebooks and DS Software Development

When a notebook is used beyond development

- EDA and project development take place in notebooks
- Mature project code uses library code and build scripts

What about code that *needs* to be presented, not computed?

- Such code stays in notebooks, but should be for human consumption only!
- Notebooks should *not* generate targets, they *are* targets!
  - A notebook is a report!

Notebooks for project development should not be included in a final project!

The purpose of notebooks in your project

- You may write your report in a notebook
  - strip out the code; convert to HTML or PDF.
- Notebooks may function as an "appendix" to your main report:
  - Detailed analyses that are disruptive to the flow of the main paper
- Notebooks should be for reading and understanding
  - It is *not* raw plots and tables without explanation.
  - *Always* include written explanations of what is being done.

You may develop code in notebooks, but you should keep them in a development branch. A development notebook has no place in `main`!

Guidelines for Notebooks in your project

- Notebooks should never generate data outside the notebook
  - Use library code and a build script for generating data
- Notebooks contain minimal code
  - It should *never* take more than 1-2 minutes to run; long running jobs should run in the background!
  - If the code is large enough to create a function, make it library code (and import!). This will clarify what is being done in the notebook by future project developers.
- When possible, use a build script to save tables/figures and simply import these into a notebook (w/o project specific code).
- In your build script, execute notebooks and strip out code to auto-generate your report!

Example Notebook workflow

Example project template:
https://github.com/DSC-Capstone/project-templates/tree/EDA

This project template:

- contains a development EDA notebook that uses library code to import the data, then does *light* computing to generate plots.
- contains a report notebook, similar in content to the EDA dev notebook:
  - simply reads in tables/graphs generated by library code (images/tables saved to file)
  - is automatically built and converted to an HTML report in run.py