Lecture 10: Preparing Data and Exploratory Analysis

Heidi Perry, PhD

Hack University

heidiperryphd@gmail.com

11/10/2016

Overview

1. Getting Data
2. Cleaning Data
3. Missing Data
   - Mechanisms
   - Strategies to Deal with Missing Data
4. Exploratory Analysis
5. Transforming Data

Getting Data: Sources

Quality and Veracity

Of foremost importance is that the source of the data set is reliable, that the data comes with information about how it was collected, and is of high quality.

- File, comma or tab-delimited.
- API
  - See DataSources file for links to APIs with Python wrappers.
  - `pandas.DataFrame.read_json` for JSON (see other pandas options for function to read other formats)
- Webscraping
  - See "Data Science from Scratch" for a good tutorial to write your own web scraper.
  - Or use Scrapy

Not all of these options gives data in a nice table format. Pandas has functionality to read many common formats into a DataFrame, but for unstructured data, you may need to write your own parser.

Getting Data: Formats

XML (eXtensible Markup Language)

Markup construct begins with < and ends with >. Three types of tags:

- start-tags `<section>`
- end-tags `</section>`
- empty-element tags `<line-break />`

Content is the text between the tags. Source: Wikipedia

JSON (JavaScript Object Notation)

Syntax looks like python dictionary. Example (compressed):

```
{ "firstName": "John", "lastName": "Smith",
  "phoneNumbers": [ {"type": "home",
                     "number": "212 555-1234"} ]
}
```

Source: Wikipedia

Tidy Data

Tidy Data

- Each variable in one column, with a human-readable variable name.
- Each observation is in one row.
- One table (dataframe) for each kind of variable.
- Multiple tables linked by a common column.

Code Book

- Describe variables in the data set, including the units.
- Explain summary and transformation choices made in preparing the tidy data set.
- Include a thorough explanation of how the data was collected.

For your analysis to be reproducible, you must record every step of the obtaining and preparing the data.

Reproducible Analysis: Cookiecutter Data Science

"A logical, reasonably standardized, but flexible project structure for doing and sharing data science work." [Driven Data]

Directory structure

```
├── LICENSE
├── Makefile           <- Makefile with commands like `make data` or `make train`
├── README.md          <- The top-level README for developers using this project.
├── data
│   ├── external       <- Data from third party sources.
│   ├── interim        <- Intermediate data that has been transformed.
│   ├── processed      <- The final, canonical data sets for modeling.
│   └── raw            <- The original, immutable data dump.
│
├── docs               <- A default Sphinx project; see sphinx-doc.org for details
│
├── models             <- Trained and serialized models, model predictions, or model summaries
│
├── notebooks          <- Jupyter notebooks. Naming convention is a number (for ordering),
│                         the creator's initials, and a short `-` delimited description, e.g.
│                         `1.0-jqp-initial-data-exploration`.
│
├── references         <- Data dictionaries, manuals, and all other explanatory materials.
│
├── reports            <- Generated analysis as HTML, PDF, LaTeX, etc.
│   └── figures        <- Generated graphics and figures to be used in reporting
│
├── requirements.txt   <- The requirements file for reproducing the analysis environment, e.g.
│                         generated with `pip freeze > requirements.txt`
│
├── src                <- Source code for use in this project.
│   ├── __init__.py    <- Makes src a Python module
│   │
│   ├── data           <- Scripts to download or generate data
│   │   └── make_dataset.py
│   │
│   ├── features       <- Scripts to turn raw data into features for modeling
│   │   └── build_features.py
│   │
│   ├── models         <- Scripts to train models and then use trained models to make
│   │   │                 predictions
│   │   ├── predict_model.py
│   │   └── train_model.py
│   │
│   └── visualization  <- Scripts to create exploratory and results oriented visualizations
│       └── visualize.py
│
```

Operations to Consider

- Merge or split tables
- Cast variables into appropriate type
- String manipulation to standardize text variables
- Unify terms
- Create intervals

Missing or Bad Data

- [Re]code missing data.
- Remove duplicates
- Identify bad data
  - Values out of range or of wrong type.
  - Outliers

    Warning: data *cleaning* is not data manipulation, so careful consideration must be given when deciding to remove bad records or outliers.
- Examine the missing data
  - Consider impacts to conclusions and inference.
  - Is there an understandable reason for the data, or does this indicate a problem with data collection?
  - Does the missing data prevent you from answering the question under investigation?

Missing Data Mechanisms

- **Missing completely at random** (MCAR): The probability of a missing value is independent of all other variables. Ignoring cases with missing MCAR data will not introduce bias into a model.
- **Missing at random** (MAR): The probability of a missing value is dependent on **other** variables, but not the value of the missing variable. Ignoring cases with missing MAR data is reasonable, as long as all of the predictors of the missing data are included in the model.
- **Missing not at random** (MNAR): Missing variables are conditional on other unavailable variables, or even on the value itself. If missingness is not random, it must be modeled explicitly, or the resulting model will include bias.

Handling missing data when the missingness is random is relatively easy, but we can never prove that the missingness is truly random.

Deletion Methods

- Complete case analysis
  - Remove all cases missing any variable.
- Available-case analysis
  - Remove only cases where variables in current analysis/model are missing.
- Nonresponse weighting
  - Build a model to predict the nonresponse in a variable with missing data using all other variables, use the inverse of predicted probabilities as weights to make the complete-case sample more representative.

Imputation Methods

To retain all the data, rather than delete cases missing variables, we may want to "impute" (i.e. fill-in) the missing values.

Simple Imputation Methods

- Mean imputation
- Indicator variables for missingness
- Use other information

Random Imputation Methods

- Simple random imputation (not recommended)
- Random regression imputation
- Hot-deck imputation

Imputation of Several Missing Variables

- Routine multivariate imputation
  - Fit a multivariate model to all the variables that have missing values, and generalize the random regression imputation for a single variable.
  - These imputations are only as good as the model.
- Iterative regression imputation
  - Impute all missing variables using a crude approach (e.g. simple random imputation), then use univariate regression to estimate values from one variable at a time, iterating through all variables with missing values until convergence.

Exploratory Analysis: Goals

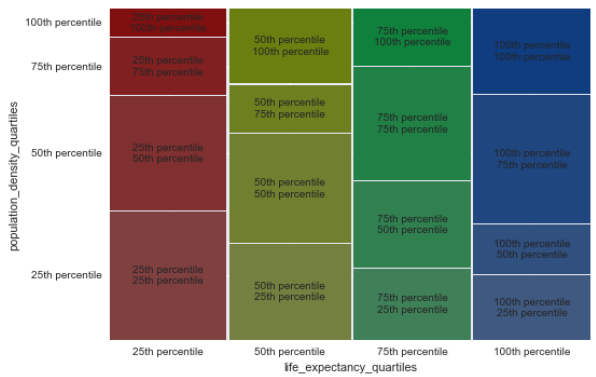
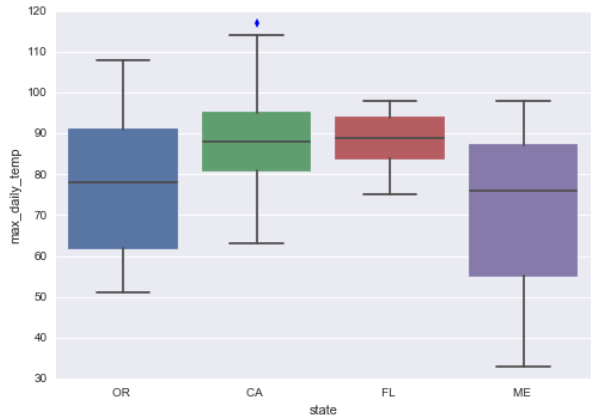
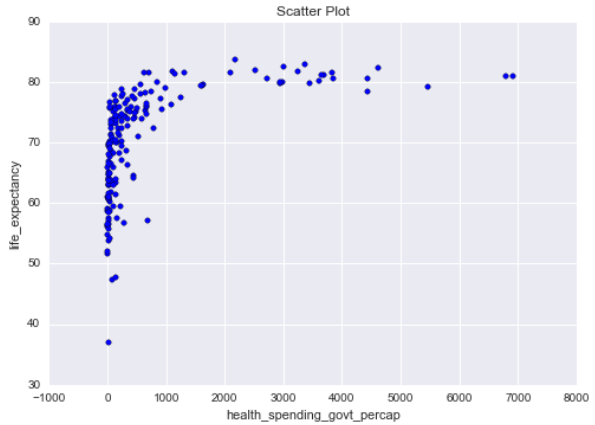
- Organize and summarize raw data.
- Identify potential problems with your data set.
- Determine if the question you are asking can be answered by the data that you have.
- Prepare the foundation to answer your question.
  - Verify that your hypothesis is worth pursuing.
  - Assess assumptions on which statistical inference is based.
  - Select appropriate modeling tools and techniques.
- Potentially develop new hypotheses.

Exploratory Analysis: Checklist

- Formulate your question.
- Load the data.
- Clean the data (this will be on-going).
- Prod the data.
  - Number of rows and columns.
  - Data types of each column.
  - View top and bottom, look for format and reasonable values.
  - Check spread of numerical variables, and sets of categorical variables.
- Validate with external data source.
- Plot the data.
  - Univariate Distributions: histogram quantitative variables, bar chart of frequency for categorical variables.
  - Bivariate Graphs: Determine explanatory and response variables in your question and plot together to check for a signal of the expected relationship.
  - Check other combinations for potential new hypotheses (bias warning).
- Challenge your assumptions.
  - Do you have the right data to answer your question?
  - Do you need additional data?
  - Do you have the right question?

Exploratory Analysis: Bivariate Plots

Response

Explanatory	Categorical	Numerical
Categorical		
Numerical	Bin the quantitative variable to create a categorical variable.	

Standardize, Normalize

Transformations to compare variables measured on different scales.

- Ordinal Scale

  Convert the values to a rank order.

- Interval scale

  Standardize to a mean of zero and standard deviation is one.

$$x_i^* = \frac{x_i - \bar{x}}{s_x}$$

- Ratio scale

  Normalize the variable vector, i.e. transform so vector length is equal to 1.

$$x_i^* = \frac{x_i}{\sqrt{\sum_i {x_i}^2}}$$

Normalizing, Rescaling

Other transformations to make disparate data comparable.

- Rescaling

  For some applications, you may want to change the data range, most likely to $[0, 1]$.

$$x_i^* = \frac{x_i - min(x)}{max(x) - min(x)}$$

- Normalize (another sense of the word)

  If measurement samples are different, divide by the value of another variable.

References


Roger Peng & Elizabeth Matsui (2015)
The Art of Data Science, Leanpub


Jeffrey Leek
Lecture: "The Components of Tidy Data"


Jacqueline Kazil and Katharine Jarmul
Data Wrangling with Python, O'Reilly Media (2016)


Andrew Gelman & Jennifer Hill (2006)
Data Analysis Using Regression and Multilevel/Hierarchical Models Cambridge University Press


Cookiecutter Data Science project
https://drivendata.github.io/cookiecutter-data-science/

Recommended Reading

Data Wrangling with Python, Chapters 6-8
Data Science from Scratch, Chapters 9-10
Art of Data Science, Chapter 4

Resources
Treatment of Missing Data - Part 1, Part 2
Data Science Design Pattern #3: Handling Null Values

Articles for discussion:
For Big Data Scientists Hurdle to Insights is Janitor Work
The Grammar of Data Science
What Went Wrong in Flint Water Crisis, a cautionary tale about removing outliers.

Example Jupyter Notebooks

- Cleaning Data
  - Using the AgCensus data set from the USDA, learn how to perform common operations: load in data with missing value code, rename variables, cast types, check for duplicates, create tidy subset.
- Transforming Data
  - This notebook demonstrates some functionality in pandas using climate data from four metropolitan areas: groupby to operate on partitions of data; apply to normalize; create dummy variables; cut to bin a numerical variable to transform into categorical.
- Exploratory Analysis
  - Datasets on health, economy, and education from Gapminder are used to demonstrate merging, binning, and plotting.
- Common Functions
  - Learn to identify a few common, non-linear relationships.

Project Proposal

Presentation on Tuesday (November 15)

- 2-5 minutes
- Which Hack Oregon Project?
- A brief summary of the data.
- What question are you seeking to answer with the data?
- Why is that an interesting question? Who is the audience?
- Q&A/discussion: answer questions from the class (and ask questions!)

Team topics

- What is the goal of the project?
- How was the data collected?
- Describe the data set - observations and variables.

Exercise and Homework

Exercise: Prepare your data for the exploratory phase of your course project.

- How was the data collected?
- Describe the data fields in your data set: variable name and type.
- How is missing data encoded?
- Load the data into a pandas dataframe.