Data Analysis for Business Analytics

CIS621

Week 2 - Data Types, Structures, Storage, and Qualities

Lecture Goals

1. Understand *common* types of data
2. Understand *common* structures in which data is stored when programming (particularly in Python)
3. Understand *common* ways in which data is persistently stored
4. Understand *common* qualities of data

General Types of Datasets

1. Time-series: a sequence of the same measure captured over time
    a. A baseball player’s daily hit total throughout a season
2. Cross-sectional: data that describes different observations at a given time
    a. Baseball stats for every player at the end of the 2021 season (each row represents a player’s stats at the end of the season)
3. Panel: data the describes different observations at different points in time
    a. Yearly baseball stats for all MLB players over the last 10 years (each row represents a player-year combination)

Data Types

1. Categorical (food: “bbq”, “asian”, “italian”)
2. Numeric
    1. Integer (1, 2, 3, 4, 5)
    2. Float (3.14, 42.0)
3. Boolean
    1. True and False
4. None

What kind of data is a zip code?

Ex: 64105

What kind of data is “2”?

(Hint: Being wrapped in quotation marks is relevant)

Demo: How do these data types look in Python?

Bonus: Representing Categories as Numeric Data

- Machine learning models cannot understand strings - they can only understand numbers. Therefore, if we want to use ML, we have to encode categorical data in a numeric way.
    - Some R libraries do this under the hood for us. Most of the time in Python (but not always - see Catboost, for example) you need to perform such a transformation on your end.

Bonus: Representing Categories as Numeric Data

- One-Hot Encoding
  - Most common way (though not always the best)
  - If the category is present, a value of 1 is assigned. If not, a value of 0 is assigned.
- Mean-Target Encoding
  - Replace the category level with the mean of the target (i.e. the value we want to predict)
  - Must be careful to avoid “leakage”
  - Several related methodologies exist (e.g., weight of evidence)
- Embeddings
  - Learn how “close” strings (categories) are.
  - Word2Vec was the original. We now have BERT.
  - For example, BERT will recognize that “zucchini” is more similar to “squash” than to “broccoli”

Built-In Python Data Structures

- List
- Set
- Tuple
- Dictionary

Other Common Data Structures

- Array (numpy)
  - Vectors (one dimension)
  - Matrix (two-dimensions)
  - Tensor (n-dimensions)
- Series (pandas)
- DataFrame (pandas)

Demo: How do these data structures look in Python?

Group Activity

- In which data structure would you store the following data?
    - Customer descriptions (e.g., age, location, interests)
    - State to zip-code mappings
    - Time-series of egg prices
    - Dimensions of various shapes (e.g. circle, square, rhombus, tetrahedron)
    - Answers to multiple choice questions

- Data structure options: list, set, tuple, dictionary, vector, matrix, tensor, series, dataframe.

Common Types of Flat Files

- CSV
- XLSX
- TXT
- JSON
- PKL (Python)

More Efficient Flat Files

- Compressed
- Parquet
- Feather

Demo: How can we interact with flat files in Python?

Getting Data from APIs

- An API is like a contract: “If you send me data in a specified format, I will return to you data in a specified format”.
- An API “lives” on a remote server accessible via HTTP methods (most often).

Demo: Getting Data from an API

Using a Webscraper

- A webscraper allows you to pull data from the HTML on a website.
- Not all websites permit webscraping! Some permit webscraping within certain limitations (e.g., don’t make requests more quickly than a human would).
- Many websites will also block your requests if you webscrape too much!

Demo: How do we build a webscraper?
Bonus: How do we use Selenium?

Interacting with Relational (SQL) Databases

- A relational database is comprised of a series of *tables* stored in *schemas* that can be joined on *keys*.
- A table stores data in rows and columns (think of an Excel spreadsheet).
- A schema is a related collection of tables.
- A key is a common column on which tables can be joined (e.g., a user ID).
- Follows a “schema-on-write” model.

Demo: Interacting with a MySQL database on AWS.

Interacting with NoSQL Databases

- A NoSQL “table” follows a “schema-on-read” protocol.
- That is, we can throw whatever data we want to into the database!

Demo: Interacting with a NoSQL database on AWS.

Class Discussion

- Under what circumstances would we want to use a NoSQL database?
- Under what circumstances would we use a SQL (relational) database?

General Qualities of Data (Assuming Availability)

- Completeness: How well does the data describe what we want?
- Missingness: How often is data missing?
- Reliability: Do we trust the values?
- Consistency: Are the values reported consistently over time?
- Cleanliness: Do the values need to be “cleaned up” to be more useful?

Group Activity

List *one qualitative* and *one quantitative* way to evaluate each data quality.

Completeness

Missingness

Reliability

Consistency

Cleanliness