Data Management

Pandas Library for Data Manipulation

Presented by:
Sean Campbell
Software Engineer
Computer Science Adjunct Instructor, Iona University

Link to these slides

https://tinyurl.com/2k6u9d38

Outline

1. Intro and Background
2. Review: Setting up Jupyter notebook
3. Finding data sources
4. Loading data
5. Summarizing and aggregating data
6. Cleaning and filtering data

Intro and Background

My Background

- Graduated Iona in 2016, with a BS in Computer Science
- Adjunct professor of Computer Science here at Iona
- Software engineer at Google for ~5 years
    - Focused on internal infrastructure, but did a number of data-focused side projects
- Software engineer at Devron, a startup focused on data privacy
- Software engineer at Gaiascope, a startup focused on decarbonizing the electric grid

Intro

- What are you hoping to get out of this workshop?
- How are you hoping to apply what you learn today in your life?
- What is your background in Python?

What is Pandas?

- Python library for data analysis and manipulation
- Think “Excel for Python”
- Handles structured (table-like) data
- Stores data in-memory, may not be best tool for very large datasets
- Building blocks
    - DataFrame: table-like data
    - Series: column, has data type

Why Pandas?

- Easy-to-use compared to many alternatives
- More powerful and flexible than Excel
- Handles larger datasets than Excel
- Automate workflows you do a lot

Review: Setting up Jupyter notebook

Setting up Jupyter notebook

- You can set up a notebook on your own computer
- For this workshop, I'll use Google Colab: https://colab.research.google.com
  - No setup required – runs in a Google data center with a bunch of common Python libraries pre-installed
  - Need a Google account to run code
  - Good especially if you don't need lots of processing power or non-standard libraries
- Jupyter notebook for this workshop: https://colab.research.google.com/drive/1HocRL5iCdTZ7PVQnW5FRACrmUrfl-pHF?usp=sharing

Finding data sources

Data sources

Source	URL
Kaggle	https://kaggle.com/
Google Datasets	https://datasetsearch.research.google.com/
Data Commons	https://datacommons.org/
New York City Open Data	https://opendata.cityofnewyork.us/
United Nations Data	https://data.un.org/

Datasets we'll use today

Data	URL
Greenhouse Gas Emissions	https://www.kaggle.com/datasets/unitednations/international-greenhouse-gas-emissions
NYT Best Restaurants	https://www.kaggle.com/datasets/rummagelabs/nytimes-best-restaurants-2024
NYC Job Postings	https://data.cityofnewyork.us/City-Government/Jobs-NYC-Postings/kpav-sd4t/about_data

Loading data

Loading data

Documentation: https://pandas.pydata.org/docs/reference/io.html

```
import pandas as pd

# Read CSV file
df = pd.read_csv(‘<path to csv file>’)

# Read Excel file
df = pd.read_excel(‘<path to Excel file>’)

# And others (SQL, JSON), but we won’t get to them
```

Summarizing and aggregating data

Summarizing data

```
df.columns      # List columns in the data set
df.head()       # Look at first few rows
df.tail()       # Look at last few rows
df.info()       # Information about columns, data types, etc.
df.describe()   # Summary statistics for each column
```

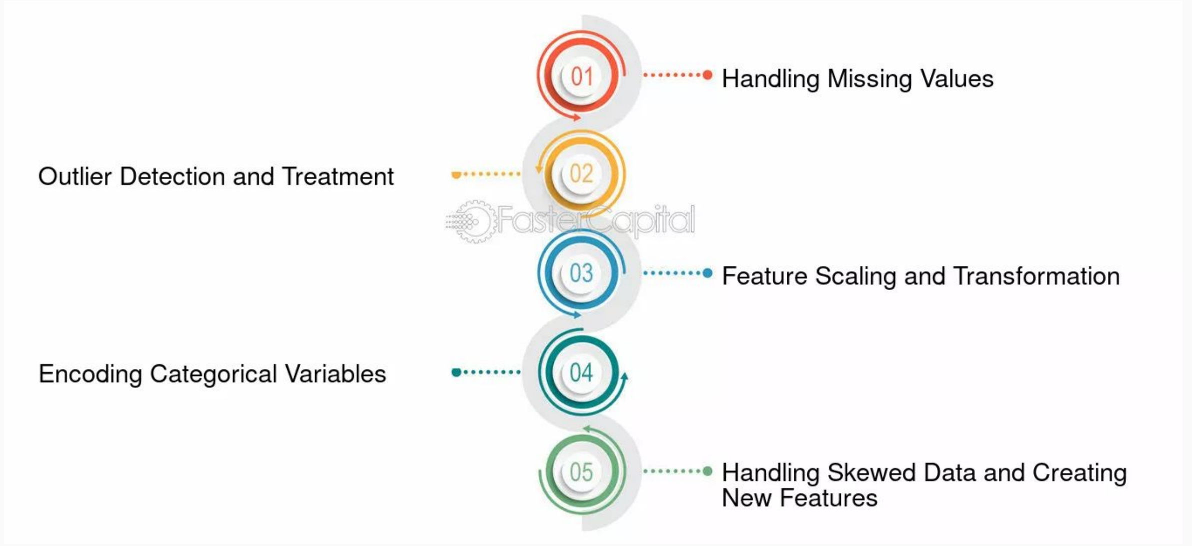
Selecting rows and columns

```
# Single column
df['col']

# Multiple columns
df[['col1', 'col2']]

# Single row
df.iloc[0]

# Multiple rows
df[0:10]
```

Summarizing columns

```
df[‘col’].min()             # Minimum value in column
df[‘col’].max()             # Maximum value in column
df[‘col’].mean()            # Mean value in column
df[‘col’].count()           # Count of non-NA values in column
df[‘col’].value_counts()    # Unique values and their counts
df[‘col’].unique()          # Unique values in column
df[‘col’].idxmin()          # Index of minimum value in column
df[‘col’].idxmax()          # Index of maximum value in column

df.groupby([‘col’]).min()    # Find min for each value in `col`
df.sort_values([‘col’])      # Sort dataframe by column
```

Cleaning and filtering data


01
Handling Missing Values
02
Outlier Detection and Treatment
FasterCapital
03
Feature Scaling and Transformation
04
Encoding Categorical Variables
05
Handling Skewed Data and Creating New Features

Filtering data

```
df[df[‘col1’] == ‘some value’]                          # Equal
df[df[‘col1’] != ‘some value’]                          # Not equal
df[df[‘col2’] < 100]                                    # Less than
df[df[‘col2’] > 100]                                    # Greater
df[(df[‘col2’] > 100) & (df[‘col2’] < 200)]             # AND
df[(df[‘col2’] < 100) | (df[‘col2’] > 200)]             # OR
df[df[‘col1’].isna()]                                   # Missing values
```

Cleaning data

```
df.dropna()           # Drop missing values
df.fillna('value')    # Replace missing values with a given value
df[df.duplicated()] # Show duplicates
df.clip(lower=0, upper=100)   # Confine values to range
df.apply(<some function>)     # Apply function over values
df['col'].astype(<type>)      # Change data type of column
```

NYC Job Postings Dataset

- What questions do you want to ask of it?
    - Highest and lowest salaries posted

https://data.cityofnewyork.us/City-Government/Jobs-NYC-Postings/kpav-sd4t/about_data

A Note on ChatGPT (and other AIs)

- It works pretty well for programming.
- It can help you write more advanced queries, and can explain what it’s doing.
- At least for now, it needs to be supervised.
    - It’s not a substitute for knowing the data.
    - Not a substitute for knowing what questions to ask.
    - It makes mistakes, assumptions, etc. And you have to catch them!
    - Depending on sensitivity of the data and organization policy, may or may not be able to use it.

Takeaways

- You can think of Pandas as the “Excel of Python”.
- Works well with data in a tabular format.
- Put in some time to understand your data. Otherwise what you think it’s telling you might not really be what it’s telling you.
- Current iterations of AI chatbots are a great tool to help with analyzing data and to learn more about how to use Pandas and other analysis tools.
- Ask a question. Code a solution. Double check your assumptions. Iterate.

NYC Posting Dataset

- Min, max salary
- Convert hourly to annual
- Average salary range per level
- Postings per year

Greenhouse Gas Emissions Dataset

- Min, max salary
- Convert hourly to annual
- Average salary range per level
- Postings per year

Synthetic Dataset

```
prices_df = pd.DataFrame({
    "product": ["apple", "banana", "yogurt", "apple", "apple",
"yogurt"],
    "price": [1.0, 0.5, 7.0, 0.9, 0.9, 5.0],
    "store": ["Stop & Shop", "Walmart", "Stop & Shop",
"Walmart", "Amazon", "Amazon"]
})
```