Exploratory Data Analysis Tools

An investigation into python libraries for EDA

Today's Agenda

1. Introduction
2. Graphical User Interfaces for Pandas
3. EDA Dashboard libraries
4. Key considerations

Some basic EDA steps with Python

Some basic EDA steps with Python

Identify Variables and Data Types

df.head()

df.dtypes()

df.shape

Some basic EDA steps with Python

Identify Variables and Data Types

- df.head()
- df.dtypes()
- df.shape

Null and Duplicate Value Detection

- df.isnull().sum()
- df.duplicated().sum()
- df[(df['var'] == 0)]

Some basic EDA steps with Python

Identify Variables and Data Types

- df.head()
- df.dtypes()
- df.shape

Univariate Analysis & Outlier Detection

- df.describe()
- sns.boxplot()
- df['var'].hist()
- df['var'].nunique()
- df['var'].value_counts()

Null and Duplicate Value Detection

- df.isnull().sum()
- df.duplicated().sum()
- df[(df['var'] == 0)]

Some basic EDA steps with Python

Identify Variables and Data Types

- df.head()
- df.dtypes()
- df.shape

Univariate Analysis & Outlier Detection

- df.describe()
- sns.boxplot()
- df['var'].hist()
- df['var'].nunique()
- df['var'].value_counts()

Null and Duplicate Value Detection

- df.isnull().sum()
- df.duplicated().sum()
- df[(df['var'] == 0)]

Bivariate Analysis

- df.corr()
- sns.pairplot()

EDA tools can augment
this process in the following ways:

- automation
- structure
- enhanced visualization
- shareable results

Graphical User Interfaces

GRAPHICAL USER INTERFACE

D-Tale

a Flask back-end and a React front-end to bring you an easy way to view & analyze Pandas data structures

kind of feels like a specialized version of excel on top of your pandas dataframe

GRAPHICAL USER INTERFACE

D-Tale

a Flask back-end and a React front-end to bring you an easy way to view & analyze Pandas data structures

kind of feels like a specialized version of excel on top of your pandas dataframe

let's demo!

1. **Tabular data view:** data flags, sorting filtering
2. **Describe view:** univariate distributions
3. **Correlations view:** bivariate relationships
4. **Charts view:** example of plotly chart creation

You can download D-Tale here

GRAPHICAL USER INTERFACE – HONORABLE MENTION

PandasGUI

PandasGUI is an open source GUI for doing EDA in Pandas

Differences from D-tale (as described by the creator):

PandasGUI advantages:

- MultiIndex support
- Multiple DataFrames in a single UI
- Faster load time
- Standalone window instead of in-browser tabs

D-Tale advantages:

- can output code to replicate your actions
- highlighting ranges/outliers/missing
- correlations
- nicer Describe view (more stats, shows a histogram/value counts)
- can be embedded in a Jupyter Notebook

You can download Pandas GUI here

EDA Dashboards

EDA DASHBOARDS

Pandas Profiling

The pandas df.describe() function is great but a little basic for serious exploratory data analysis. pandas_profiling extends the pandas DataFrame with df.profile_report() for quick data analysis.

EDA DASHBOARDS

Pandas Profiling

The pandas df.describe() function is great but a little basic for serious exploratory data analysis. pandas_profiling extends the pandas DataFrame with df.profile_report() for quick data analysis.

let's demo!

1. **Overview & Warnings:** duplicates, nulls, collinearity, univariate distribution
2. **Variables:** field-level warnings and univariate analysis
3. **Interactions & Correlations:** bivariate analysis

You can download Pandas Profiling here

EDA DASHBOARDS – HONORABLE MENTION

SweetViz

an open source Python library that generates beautiful, high-density visualizations to kickstart EDA with a single line of code.

Output is a fully self-contained HTML application.The system is built around quickly visualizing target values and comparing datasets.

Salient differences from Pandas Profiler:

- works as standalone html app
- default orientation is to compare two datasets (eg train vs test)
- provides less robust data description/cleanup, but better comparison across categorical variables

You can download SweetViz here

SUMMARY & CONSIDERATIONS

Benefits

These python libraries can augment your existing EDA approach in the following ways:

- provide structure to your process to help ensure you don't miss any steps
- make EDA more visual
- "automate the boring stuff" --> so you can spend more time exploring interesting questions
- create share-able dashboards and charts to make your findings more accessible to stakeholders, especially less-technical team members

SUMMARY & CONSIDERATIONS

Benefits

These python libraries can augment your existing EDA approach in the following ways:

- provide structure to your process to help ensure you don't miss any steps
- make EDA more visual
- "automate the boring stuff" --> so you can spend more time exploring interesting questions
- create share-able dashboards and charts to make your findings more accessible to stakeholders, especially less-technical team members

Key Considerations

But there are some reasons to be careful about how and when to use these tools:

- **reproducibility:** if your work was done entirely through a GUI, it means it won't be reproducible by others.
  - **recommendation**: limit use of GUI to act as a supplement for your code. Any charts/ insights/etc that you or others would want to check or reproduce should be crated using code.
- **learning**: becoming overly reliant on a third-party UI could detract from your ability to practice and master python code.
  - **recommendation**: limit use, and frequently sense-check whether you can reproduce your results with code.
- limited ability to explore categorical variables
- dashboard tools may be too slow on big data

Thank you!