



# Data Science Upskilling Workshop

## Session 3: Data Model, Data Wrangling, and Visualization

Oren Livne and Jiangang Hao  
Educational Testing Service

NCME 2022 Training Workshop – 4/11/2022

# This session featuring...

- Data science basics
  - Data science overview
  - Data storage – data lake, warehouse, mart
  - Data model – document, relational, graphic
  - Data processing
- Data wrangling
  - Parsing **XML and JSON structured formats**
  - Parsing **unstructured files**
- Interactive visualization
- Dashboard



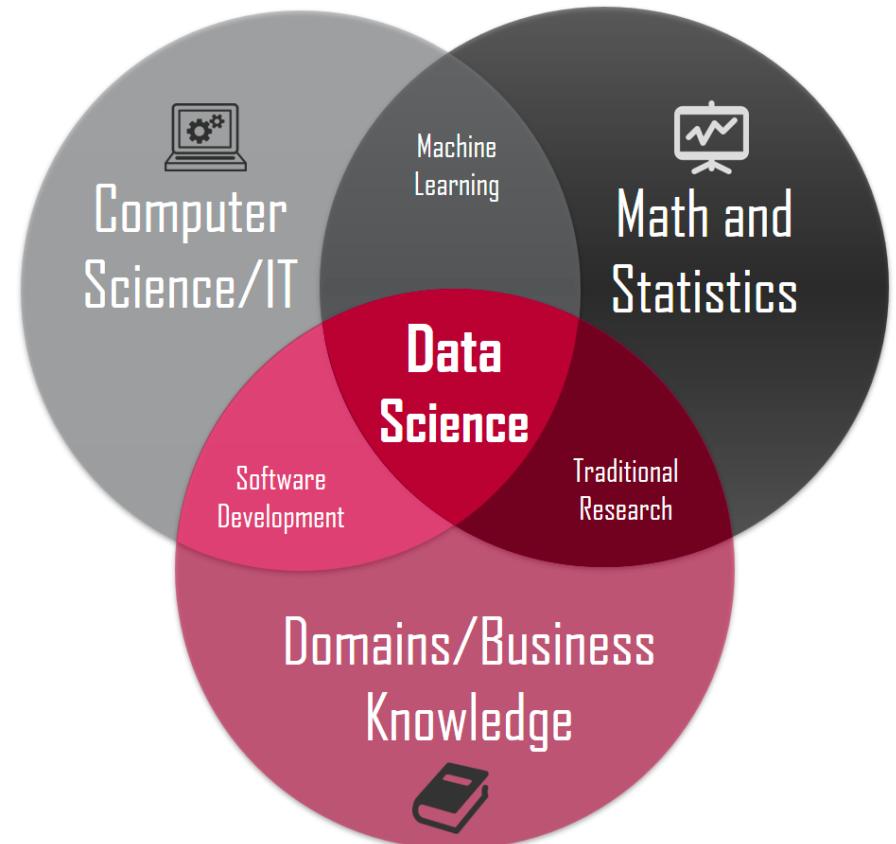


# What is Data Science

# What is Data Science

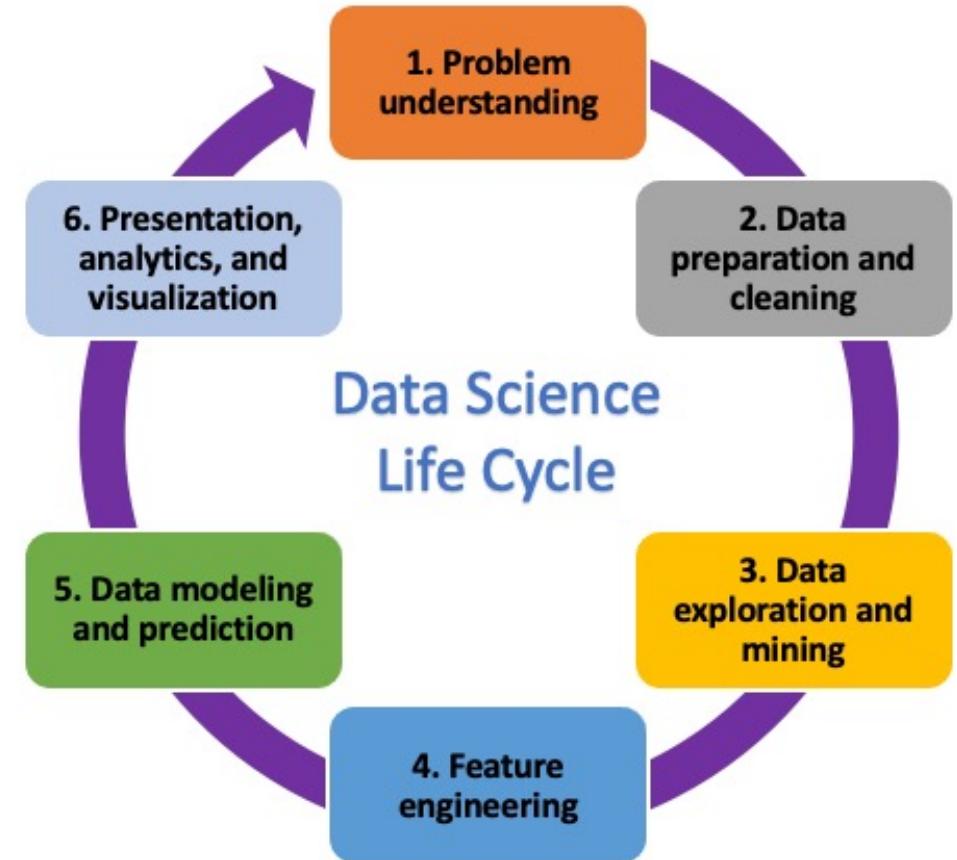
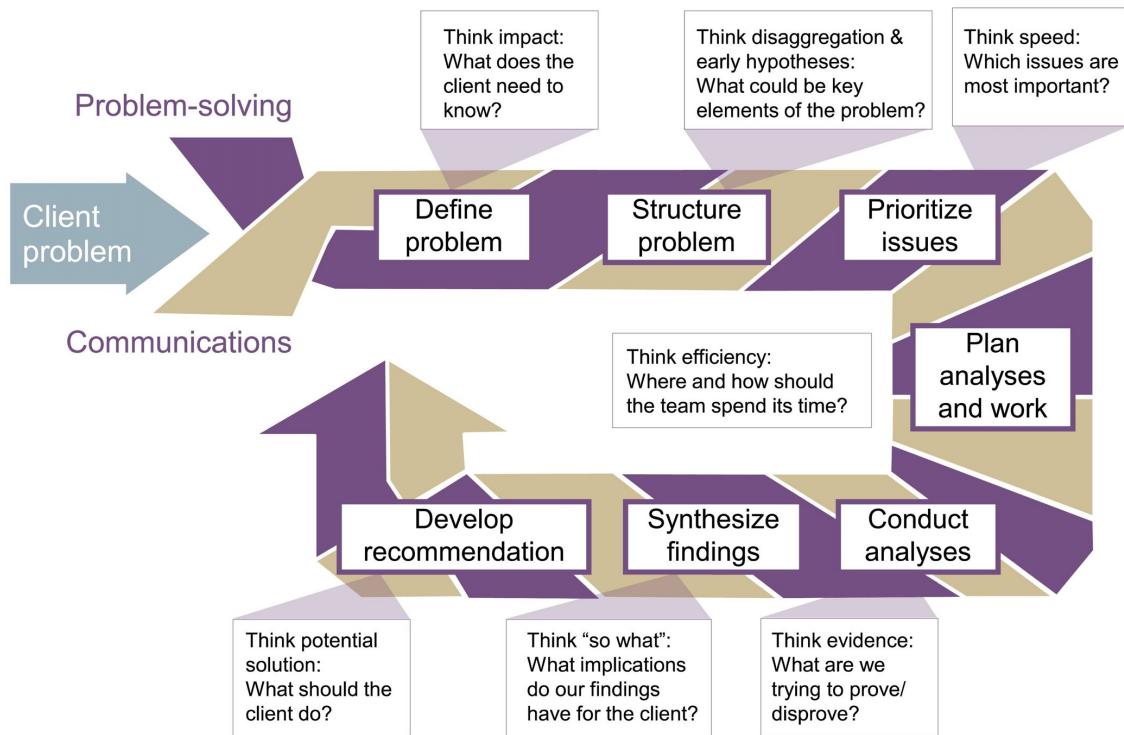
**Data science** is an inter-disciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from many structural and unstructured data

- Wikipedia

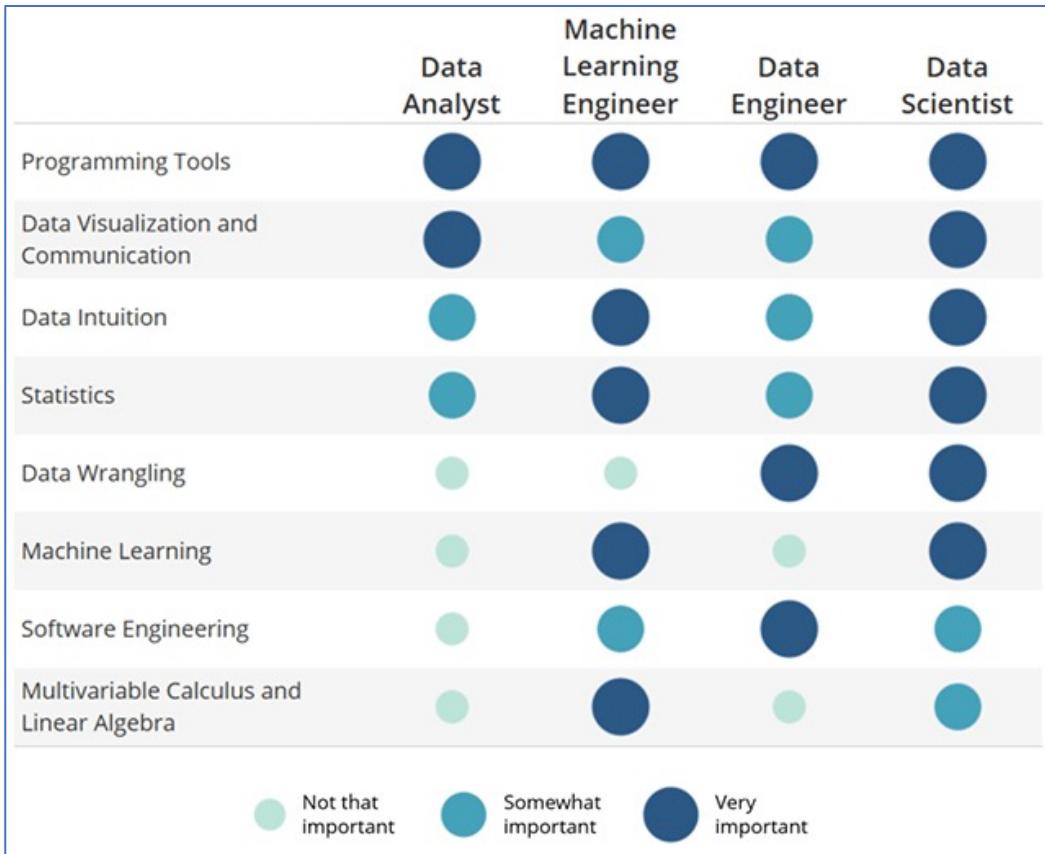


# Life Cycle of Data Science Projects

McKinsey 7-step problem-solving process



# Skills Needed



<https://www.udacity.com/blog/2014/11/data-science-job-skills.html>

To be a good scientist, one also needs

- Curious & creative
- Patient, detail-oriented, & persistent
- Open-minded & courageous
- Computational thinking
- Critical thinking
- Problem solving
- Hands-on and doing
- Collaborative and communicative

[https://www.canr.msu.edu/news/what\\_makes\\_a\\_good\\_scientist](https://www.canr.msu.edu/news/what_makes_a_good_scientist)





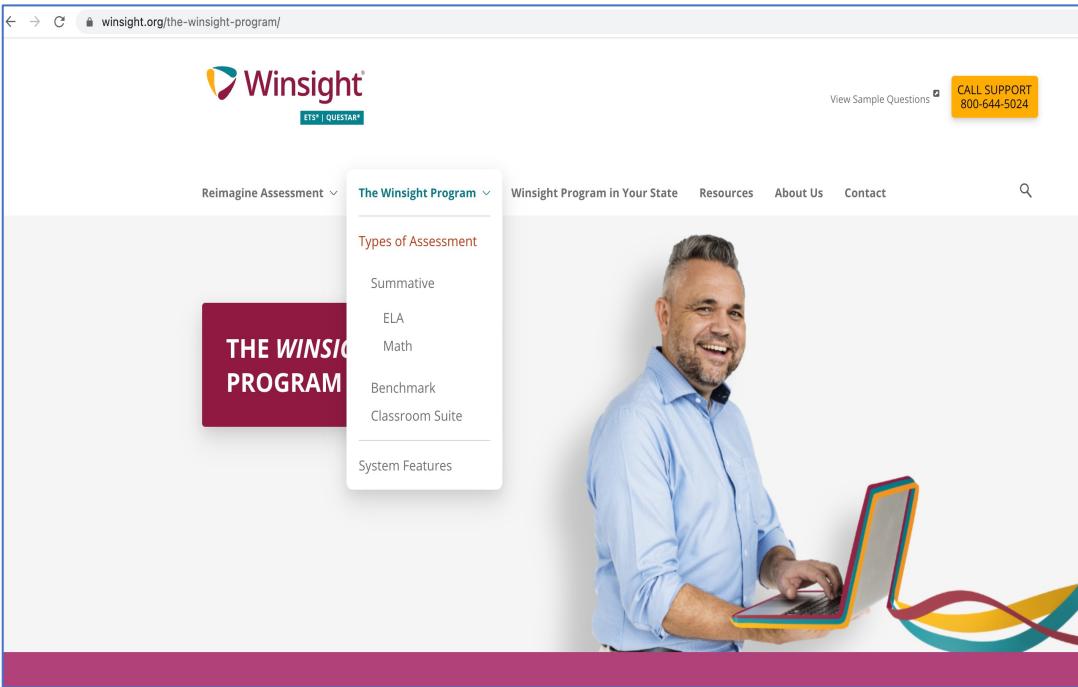
# Why it matters in assessments?

# Digital Learning and Assessment

- Advance of digital technology is one of the defining features of our age
  - It dramatically changed how people interact with each other
  - It requires new competencies
    - ✓ Communication, collaboration, critical thinking, creativity, etc.
  - It enable new ways of leaning and assessment
    - ✓ Online learning, games, simulations, virtual reality, AI, etc.
- Data are the key to materialize the promises of digital technology
  - New data require new techniques to process
  - Integrating methods from data science and machine learning into psychometrics is essential for making sense of the data from digital learning and assessments
- Over the last a few years, ETS has made major R&D efforts to make use of the response process data from digitally based assessments (DBAs).

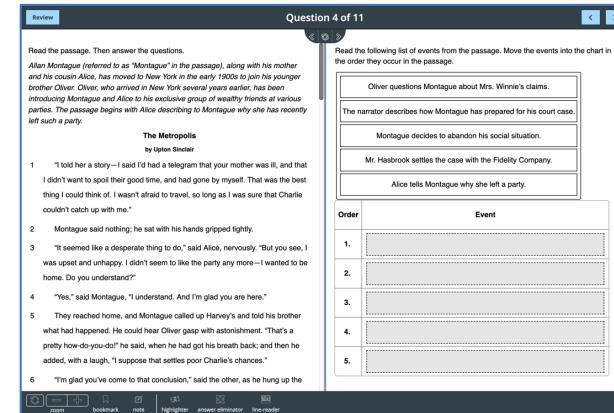


# An Example



The Winsight website homepage features a large banner with a smiling man working on a laptop. The banner text reads "THE WINSIGHT PROGRAM". The navigation bar includes links for "Reimagine Assessment", "The Winsight Program", "Types of Assessment" (Summative, ELA, Math, Benchmark, Classroom Suite, System Features), "Winsight Program in Your State", "Resources", "About Us", "Contact", and a search icon.

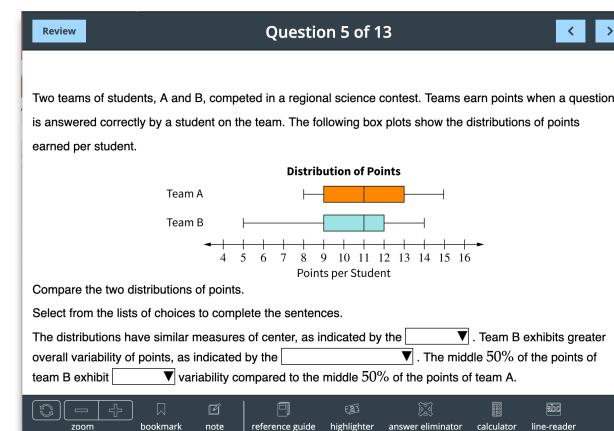
**Acknowledgement:** the findings presented in the following slides was conducted using field test data from the Winsight® summative assessment developed by ETS.



Question 4 of 11: Read the passage. Then answer the questions.  
Alan Montague referred to as "Montague" in the passage, along with his mother and his cousin Alice, has moved to New York in the early 1900s to join his brother Oliver, who arrived in New York several years earlier, has been introducing Montague and Alice to his exclusive group of wealthy friends at various parties. The passage begins with Alice describing to Montague why she has recently left such a party.

**The Metropolis**  
by Upton Sinclair  
1 "I told her a story—I said I'd had a telegram that your mother was ill, and that I didn't want to spoil their good time, and had gone by myself. That was the best thing I could think of. I wasn't afraid to travel, so long as I was sure that Charlie couldn't catch up with me."  
2 Montague said nothing; he sat with his hands gripped tightly.  
3 "It seemed like a desperate thing to do," said Alice, nervously. "But you see, I was upset and unhappy. I didn't seem to like the party any more—I wanted to be home. Do you understand?"  
4 "Yes," said Montague, "Understand. And I'm glad you are here."  
5 They reached home, and Montague called up Harvey's and told his brother what had happened. He could hear Oliver gasp with astonishment. "That's a pretty how-do-you-do!" he said, when he had got his breath back, and then he added, with a laugh, "I suppose that settles poor Charlie's chances."  
6 "I'm glad you've come to that conclusion," said the other, as he hung up the telephone.

Order Event  
1. \_\_\_\_\_  
2. \_\_\_\_\_  
3. \_\_\_\_\_  
4. \_\_\_\_\_  
5. \_\_\_\_\_



Question 5 of 13: Two teams of students, A and B, competed in a regional science contest. Teams earn points when a question is answered correctly by a student on the team. The following box plots show the distributions of points earned per student.

**Distribution of Points**

Team	Median	Q1	Q3	Min	Max
Team A	11	9	13	6	16
Team B	11	9	12	6	14

Compare the two distributions of points.  
Select from the lists of choices to complete the sentences.  
The distributions have similar measures of center, as indicated by the [ ] . Team B exhibits greater overall variability of points, as indicated by the [ ] . The middle 50% of the points of team B exhibit [ ] variability compared to the middle 50% of the points of team A.

[ ] zoom [ ] bookmark [ ] note [ ] reference guide [ ] highlighter [ ] answer eliminator [ ] calculator [ ] line-reader

From ETS Winsight item sampler



# Tool Usage Difference

## Highlighter tool

knee, the stainless steel spoon in my hand.

2 Violet the goat had gotten sick, suddenly and alarmingly. One day she was browsing the pasture, playing head-butt with the other goats, and the next day she could barely stand. Her eyes twitched, independent of her control, in a way that was both comical and horrifying. She seemed to have seizures; when she tried to walk, her legs disobeyed her and she fell heavily and awkwardly, legs stiff. We thought she was dying. The vet said she had a thiamine deficiency and gave us a bewildering array of shots and pastes to administer to her three times a day for ten days. She hated every shot—not that I could

 highlighter

 answer eliminator

 line-reader

## Line reader tool

blame her. The needles were thick and two inches long. Her shoulders must have grown sore from the repeated pokes. The yogurt was our own idea, a probiotic to encourage the essential bacteria in her gut, the original source of the missing thiamine.

3 Now, as Martin, Violet's owner, held her face between his hands, I

 highlighter

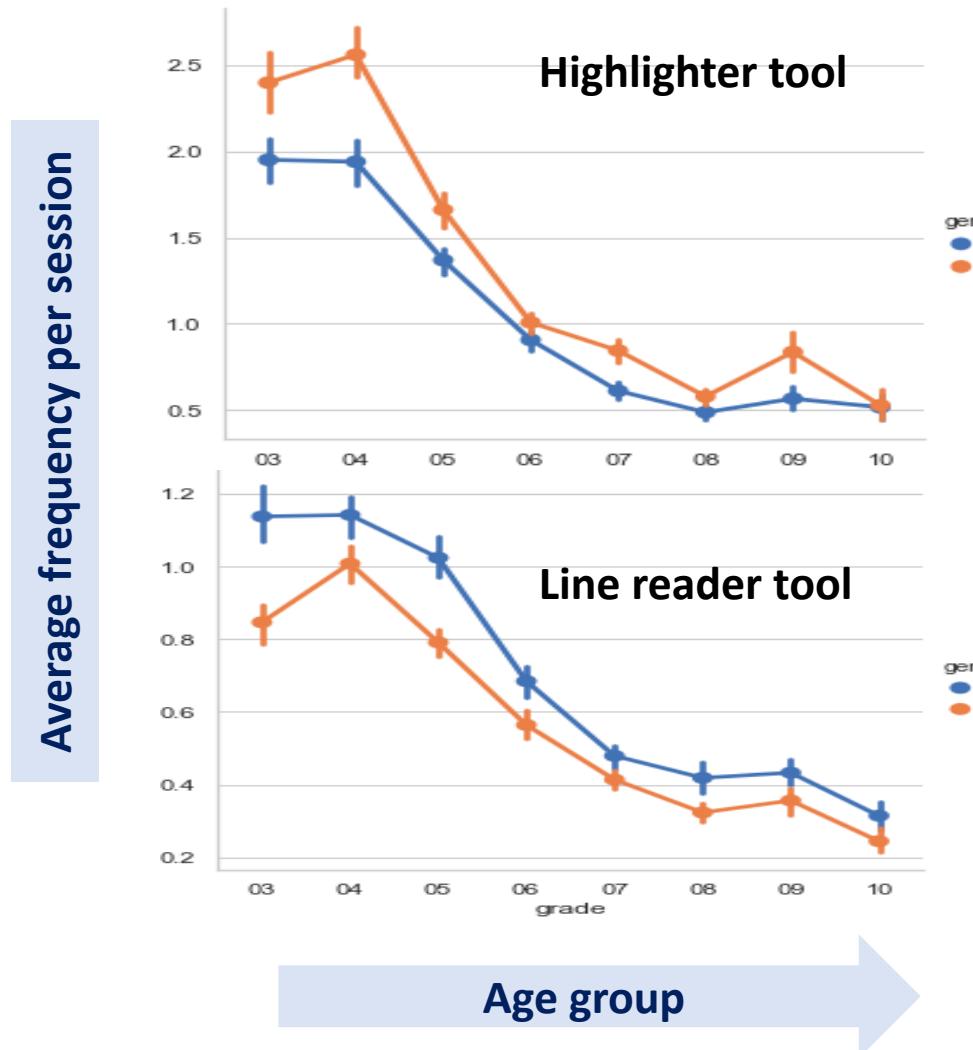
 answer eliminator

 line-reader

From ETS Winsight Assessment Sampler



# Tool Usage Difference Cont'd



## Observations:

- Girls tend to use more highlighter tool
- Boys tend to use more line reader tool
- Decreasing trend of tool usage

**How will the tool usage affect the performance? What should be the right tools to be provided?**

# Main Challenges

## Technical challenge

- How to handle the large volume of data efficiently – cloud computing (AWS)
- Domain generic, and the solutions **can** be shopped from the “market”.

## Scientific challenge

- How to make sense of the data
- Domain/application specific, and the solutions **cannot** be directly shopped from the “market”.

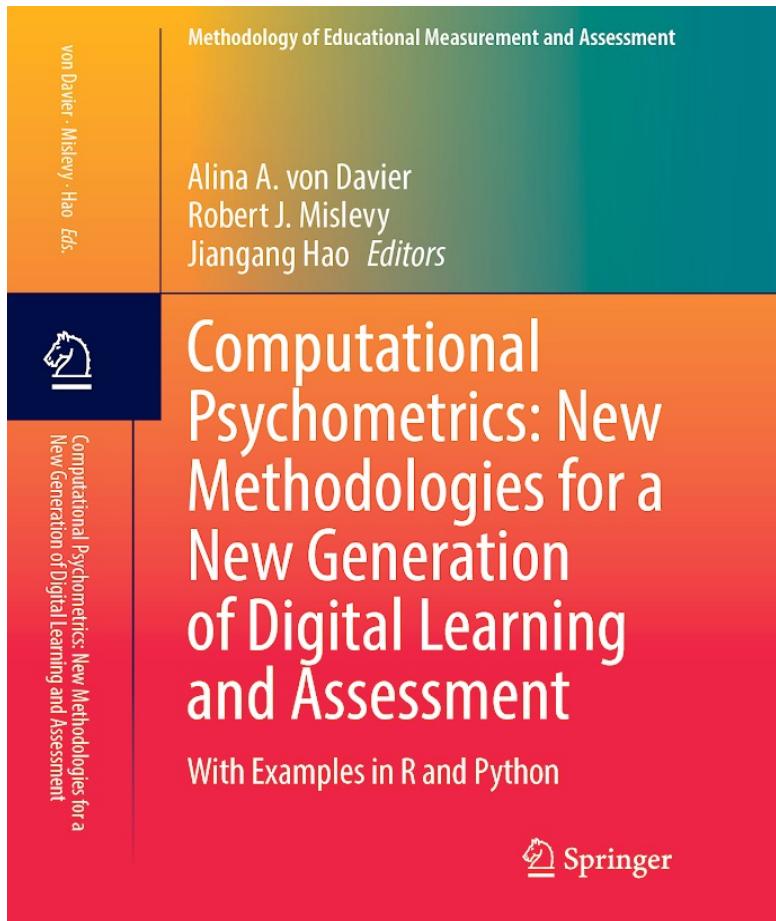
## Organizational challenge

- How to get the people with the right skills?
- How to keep them work together effectively?
- Company specific

Psychometrics researchers need to garner new data science skills



# A Springboard for Psychometrics Researchers



**Chapter 8**  
**A Data Science Perspective**  
**on Computational Psychometrics**

Jiangang Hao and Robert J. Mislevy

**Abstract** Digitally based learning and assessment systems generate large volumes of complex process data. The next generation psychometricians need to acquire new data science skills to meet the data challenge. In this chapter, we summarize data science skills and identify the subset that psychometricians need to prioritize. We introduce an evidence identification centered data design (EICDD) process during the task design, as an important way to address the data challenges from digitally based assessments. We describe some specific data techniques to parse and process complex process data with example codes in Python programming language. We also outline the general methodological strategies when dealing with process data from digitally based assessments.

**8.1 Introduction**

Digitally Based Assessments (DBAs) enable the capture of test takers' response process information at finer time granularity than traditional forms of assessment, and these rich process data can provide new opportunities for validating assessments, improving measurement precision, revealing response patterns/styles, uncovering group differences (fairness), detecting test security breaches, identifying new constructs, and providing feedback to learners and other stakeholders (Ercikan & Pellegrino, 2017; Mislevy et al., 2014). But these potential benefits do not come for free. The significantly increased volume, velocity, and variety of data pose new challenges to psychometricians for handling, analyzing, and interpreting the

The R or Python codes can be found at the GitHub repository of this book: [https://github.com/jgbrainstorm/computational\\_psychometrics](https://github.com/jgbrainstorm/computational_psychometrics)

J. Hao (✉) · R. J. Mislevy  
Educational Testing Service, Princeton, NJ, USA

[https://github.com/jgbrainstorm/computational\\_psychometrics](https://github.com/jgbrainstorm/computational_psychometrics)





# Data Type and Data Model

# Types of Data

- ***Raw*** data (structured or non-structured)
  - Images, videos, audios
  - Documents
  - Records from a single testing sessions
  - ...
- ***Processed/value-added*** data (mostly structured)
  - Features from raw data
  - Rational tables
  - Records of many students' test score
  - ...



# Data Storage

- **Data Lake:** a centralized repository that allows you to store all your structured and unstructured raw data at any scale
  - File folders
  - Cloud-based folder - AWS S3 bucket
- **Data Warehouse:** a central repository of processed data that can be used for multiple purpose
  - RDBMS: Relational Database Management System
  - MySQL, Oracle, PostgreSQL, etc.
- **Data Mart:** a subsection of the data-warehouse, designed and built specifically for a particular department/business function.

	Most Important Use Group & Use-Cases	Time-to-Market Questions & Solutions	Cost Implementation & Ownership	Users (# & Types)	Data Growth Volume & Variety
Data Lake	Predictive & Advanced Analytics	 Weeks - Months	\$\$\$\$		
Data Warehouse	Multi-Purpose Enabler of Operational & Performance Analytics	 Hours - Days	\$\$\$\$		
Data Mart	Line of Business Specific Reporting & Analytics	 Minutes - Hours	\$\$\$\$		

<https://www.holistics.io/blog/data-lake-vs-data-warehouse-vs-data-mart/>



# Example



We need to standardize the data so that the process flows smoothly

How?



# Data Model

A **data model** is an abstract model that organizes elements of data and standardizes how they relate to one another and to the properties of real-world entities. (Wikipedia)

- **Relational Model (SQL)**

- Data is organized into relations (tables in SQL DB) where each relation is an unordered collection of tuples (rows in SQL) – Codd (1970)
- Suited for data where there are many-to-many relations
- SQL database: Oracle, PostgreSQL, MySQL, etc.

- **Document Model (NoSQL)**

- Hierarchical/tree structure (JSON/XML)
- Suited for data where mostly are one-to-many relations
- NoSQL database: Mongodb, Redis, CouchDB, Hbase, etc.

- **Graph Model**

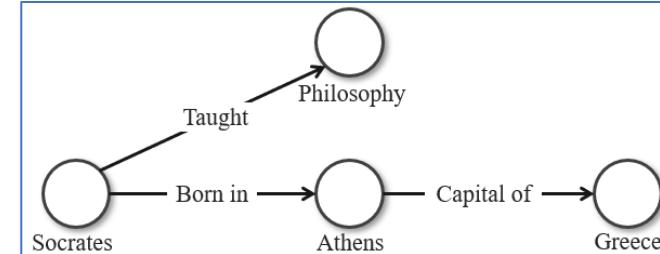
- Vertices (nodes or entities) and edges (relationship or arcs)
- Social graph: Vertices -> people, Edges -> which people know each other
- Web graph: Vertices -> web pages, Edges -> html links
- Suited for data where the many-to-many relationship is too complicated to be handled by the relational model
- Neo4j, ArangoDB, etc.



# Triple Store and RDF

- **Triple Store Model**

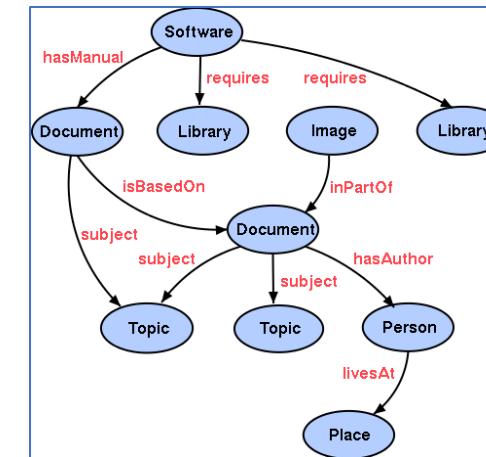
- (subject, predicate, object)
- E.g. (Jiangang, like, apple), (Oren, like, orange)
- Knowledge graph for search engine and AI applications



<https://towardsdatascience.com/auto-generated-knowledge-graphs-92ca99a81121>

- **Resource Description Framework (RDF)**

- Semantic web (Berners-Lee, 2001)
- Similar to the triple store model, but referring to webpages
- Uniform Resource Identifier (URI)
  - URN: Uniform Resource Name
  - URL: Uniform Resource Locator
- (URI\_1, URI\_2, URI\_3)



<https://vodkhang.com/intelligent-system/semantic-technology-and-its-application>

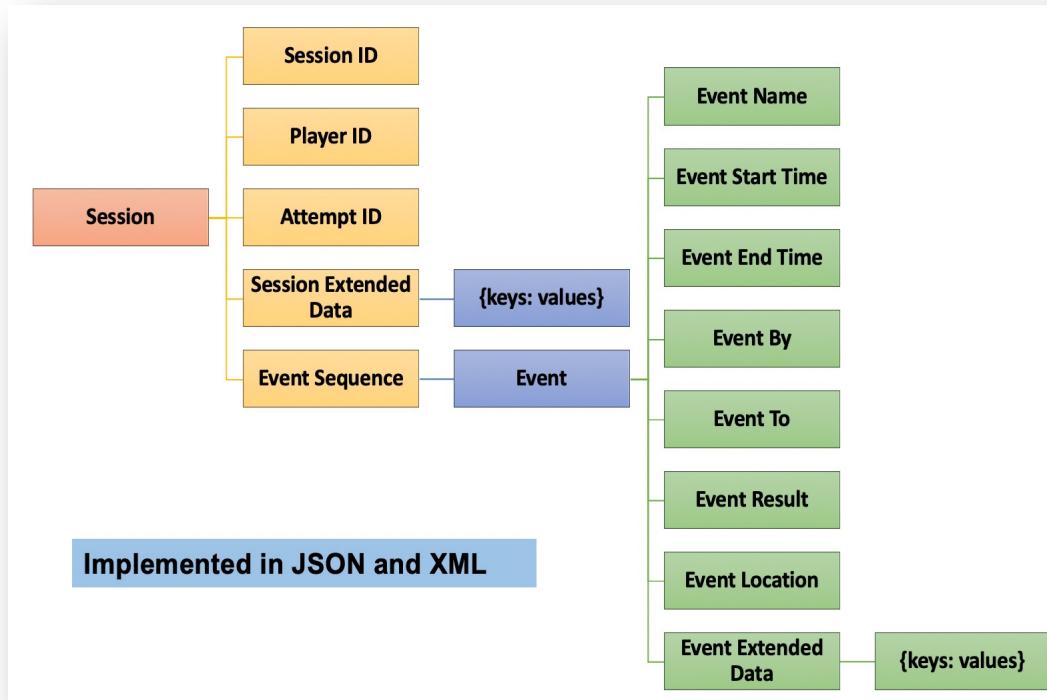


# Data Models in Learning and Assessments

- **xAPI:** experience API (or Tin Can API) is an eLearning specification that makes it possible to collect data about the wide range of experiences a person has within online and offline training activities.
  - Evolved from SCORM (Shareable Content Object Reference Model, <http://scorm.com> )
  - For learning management system (LMS)
  - Triple store style data model: Actor > Verb > Object (Activity)
  - <https://adlnet.gov/projects/xapi-architecture-overview/>
- **IMS Global Learning Consortium' Caliper:** IMS enables a plug-and play-architecture and ecosystem that provides a foundation on which innovative products can be rapidly deployed and work together seamlessly.
  - For learning management system (LMS)
  - Triple store style data model: Actor > Verb > Object (Activity)
  - <https://www.imsglobal.org/activity/caliper#caliperpublic>
- **ETS Data Model for Virtual Performance Assessment**
  - Document data model for process data from virtual performance assessments (VPAs, such as game/simulation-based assessments)
  - <https://onlinelibrary.wiley.com/doi/full/10.1002/ets2.12096>



# ETS Data Model for Virtual Performance Assessment



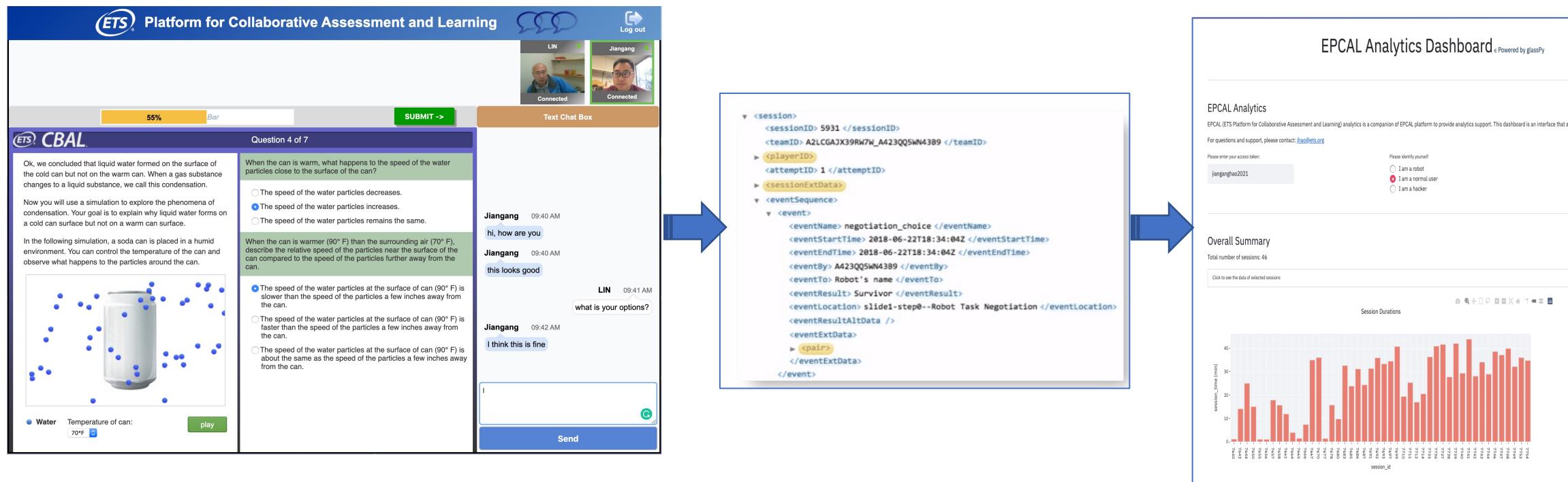
- An extra process in Evidence Centered Design
  - Log file -> Evidence Trace File

# JSON Implementation

XML Implementation ETS®

# A Full Implementation in EPCAL

Full implemented in the **ETS Platform for Collaborative Assessment and Learning**



<https://onlinelibrary.wiley.com/doi/full/10.1002/ets2.12181>





# Data Processing

# General Steps

- Data integration through ETL
  - Extract data from different sources (e.g., data lakes)
  - Transform the data to improve quality and consistency
  - Load data into a target database
- Data mining and feature engineering
- Analytics, statistical modeling, machine learning



# Analytics, Statistical Modeling, Machine Learning

- Analytics
  - Discovery, interpretation, and communication of meaningful patterns in data and apply these patterns towards effective decision making.
  - Suited for data from most digital learning and assessment tasks
- Modeling
  - The use of mathematical models and statistical assumptions to generate sample data and make predictions about the real world.
  - Psychometric/statistical modeling (e.g., Bayes net, IRTs, CDM, many others)
  - Cognitive modeling (e.g., ACT-R, SOAR, many others)
  - Better suited for data from carefully designed virtual performance tasks
- Machine learning
  - Methodologies that allow computers to “learn” the relationship among numerical representations of data without explicit instructions by human experts.
  - Supervised, unsupervised, semi-supervised, reinforcement



# Types of Data Processing

- Batch processing
  - Process data after the data are collected
  - E.g., Apache Hadoop, ...
- Stream processing
  - Process live data
  - E.g., Apache Kafka, Storm, Streamz, ...
- Micro-batch processing
  - Something in between the Batch and Stream processing
  - Apache Spark

The landscape of the software tools changes very fast and new tools emerge from time to time



# Analytics Strategy for Response Process Data

- Directly parse the JSON/XML file to extract information
- Transform sequential data to tabular data to leverage the existing tools for handling tabular data (e.g., pandas)

```
{  
  "TestSessions": {  
    "description": "Single container for the list of test sessions.",  
    "type": "array",  
    "items": {  
      "description": "Represents a student session. Each session  
      represents a student and an individual form part.",  
      "type": "object",  
      "properties": {  
        "ExamineeIdentifier": {  
          "description": "Identifies an individual student.",  
          "type": "number",  
          "examples": [  
            "1061387"  
          ]  
        },  
        "ExamineeCode": {  
          "description": "String identifier augmenting  
          ExamineeIdentifier.",  
          "type": "number",  
          "examples": [  
            "001031751"  
          ]  
        },  
        "TestName": {  
          "description": "Client-facing, friendly name of a testing  
          admin/season/window.",  
          "type": "string",  
          "examples": [  
            "MS Fall 2017 EOC"  
          ]  
        },  
        "TestSubjectCode": {  
          "description": "Internal identifier of subject/content area.",  
          "type": "number",  
          "examples": [  
            "1004"  
          ]  
        },  
        "SubjectName": {  
          "description": "Client-facing, friendly name of  
          subject/content area.",  
          "type": "string",  
          "examples": [  
            "English II"  
          ]  
        }  
      }  
    }  
  }  
}
```



Table 3. A schematic of the data frame converted from sequential data.

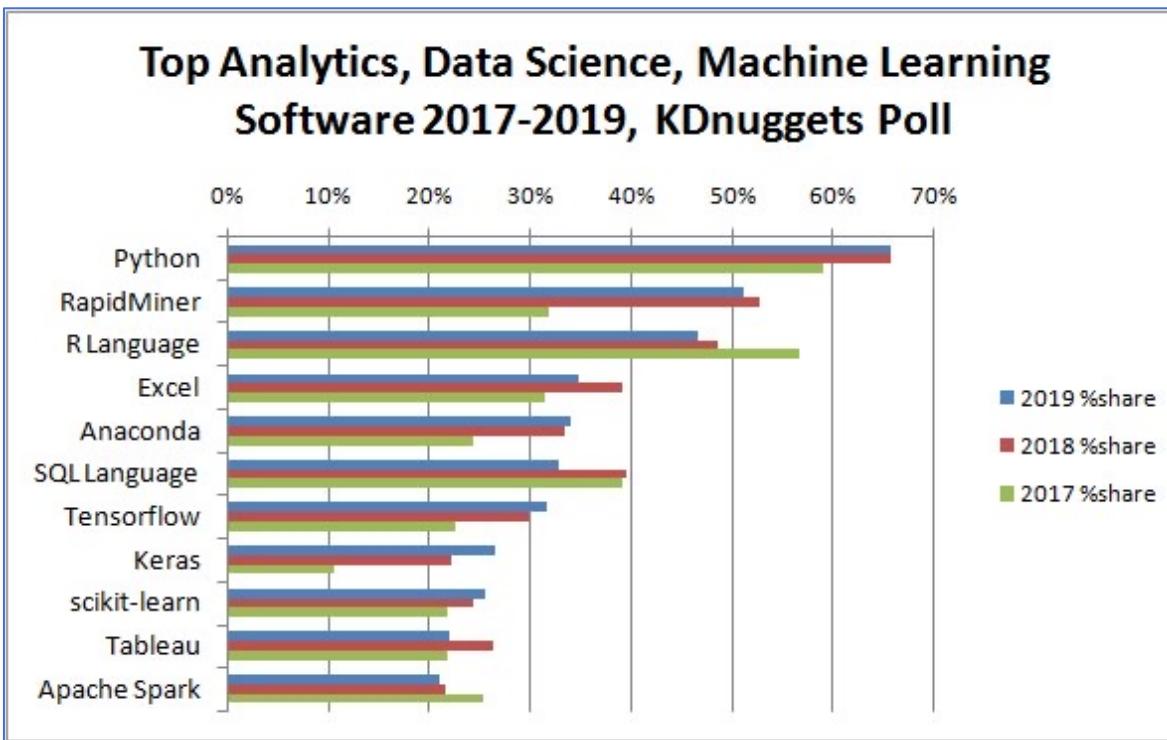
Sequence Number	Event Time	Test Taker ID	Session ID	Event Name	Attribute 1	Attribute 2	...
1	Timestamp 1	A	1	...	...	...	...
2	Timestamp 2	A	1	...	...	...	...
3	...	A	1	...	...	...	...
4	...	A	1	...	...	...	...
5	...	B	2	...	...	...	...
6	...	B	2	...	...	...	...
7	...	B	2	...	...	...	...





# Data Science Tools

# Survey from KDnuggets



## Python Programming Basics

### Course Description

Python is a widely used tool for data science and the general programming skill of python is agnostic to specific domains. Given that both *Data Science Academy* and *Introduction to AI* courses assume that the participants can program in Python and there are many good external courses on Python, we encourage participants who are not familiar with Python to take this python programming course provided by Udemy at their own pace. After completion, Udemy will provide a certificate.

### Sign up the course

We are in the process of finalizing a process to help you register this course on Udemy. So, please first list your name in the form on the overview page of this site and then wait instructions on how to enroll in the course on Udemy.

### Syllabus

Please go to the following link to start the course.

---

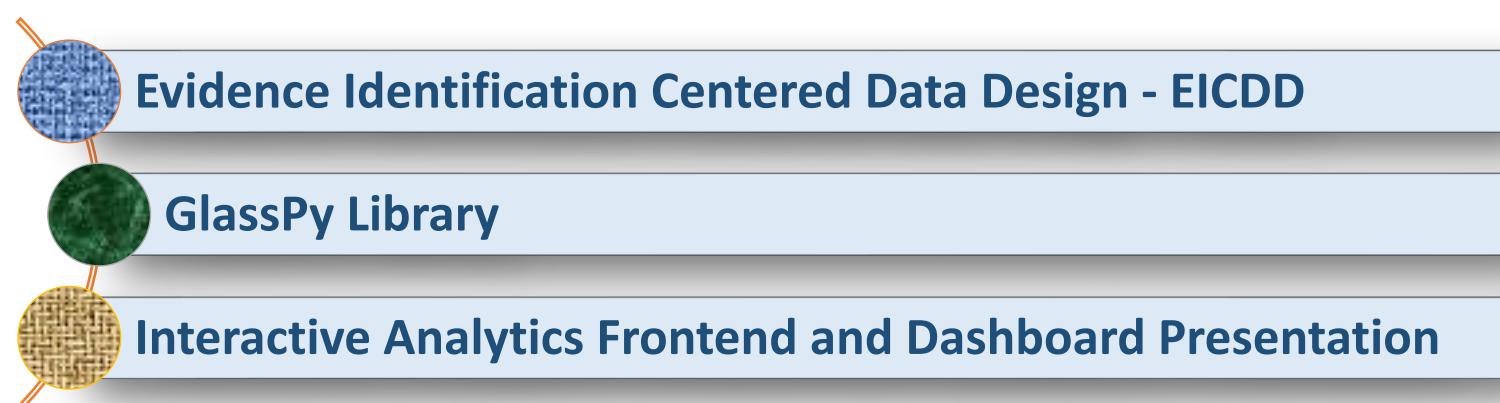


Python Bootcamps: Learn Python Programming and Code Training | Udemy  
www.udemy.com



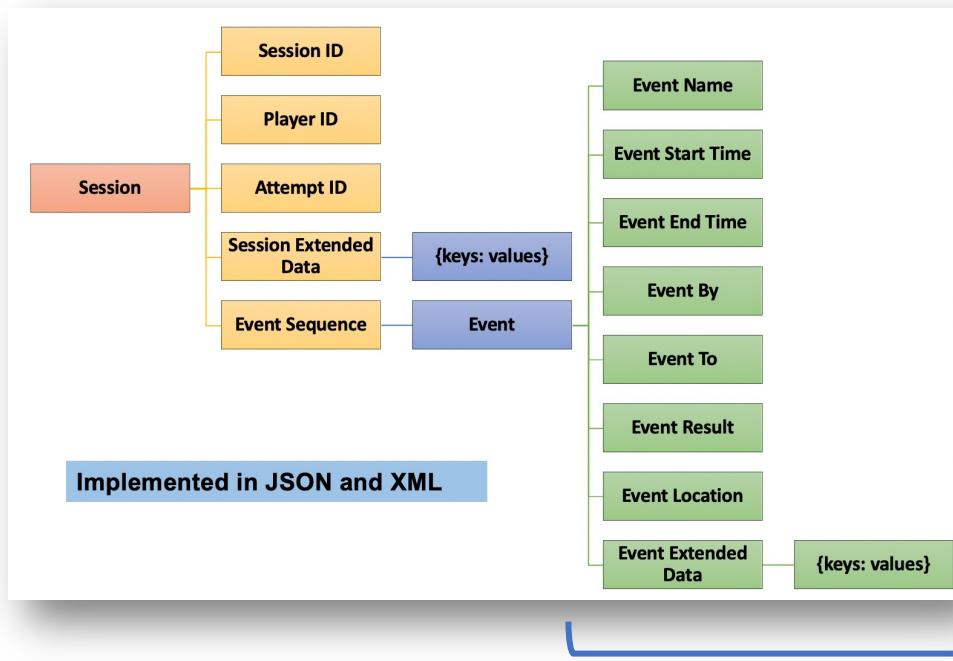
# GlassPy Analytics Solution at ETS

- GlassPy: Game Log Analysis in Python, started in 2014 at ETS
- We envision glassPy more as a generic framework for handling process data from digital-based assessments (DBAs)
- It is an “adaptor” to connect the Evidence Centered Design (ECD) to data in practice
- GlassPy analytics solution Includes three main components



# Evidence Identification Centered Data Design

## Data Model



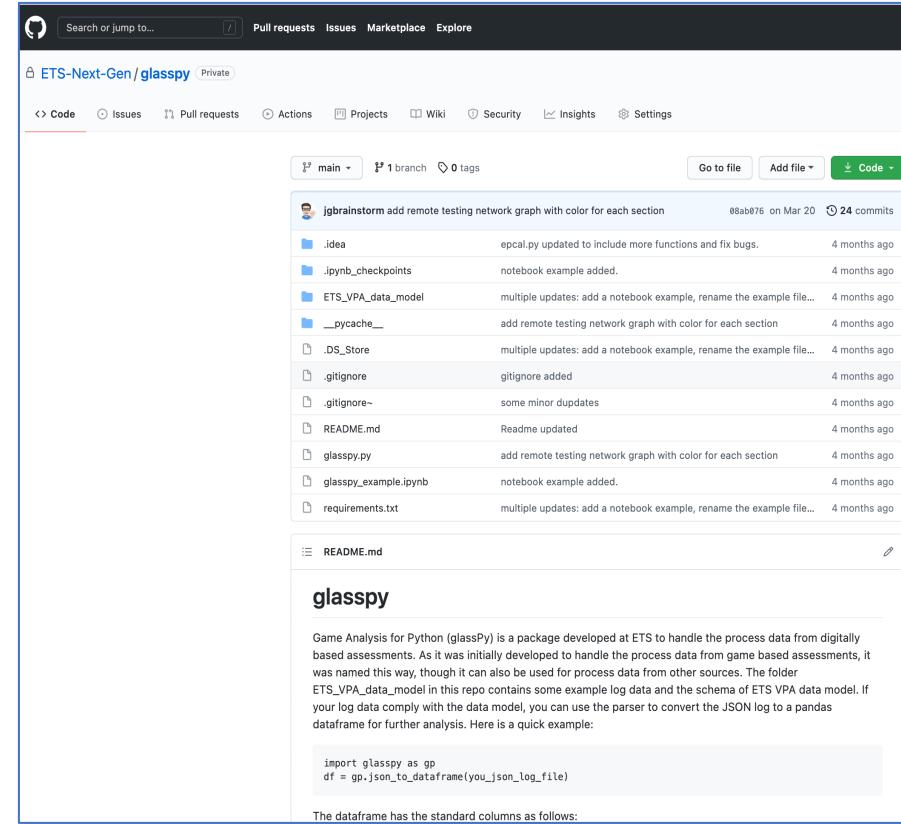
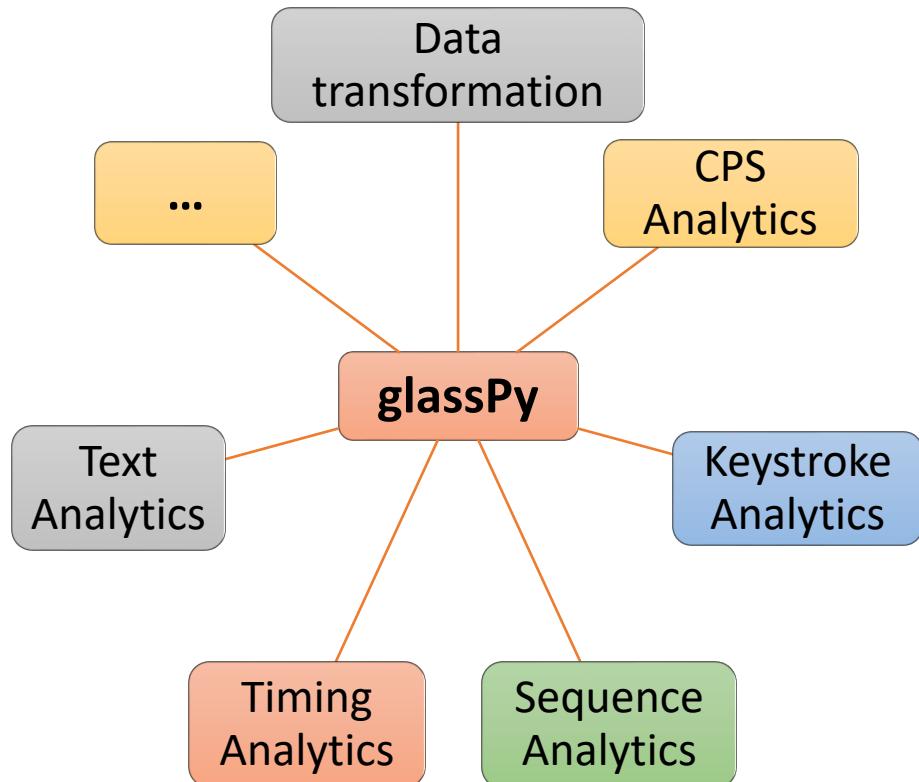
## Standard Operational Procedure

Timeline ↓	X: participant      XX: coordinator	Roles				
		Learning Scientist/ Assessment Developer	Psychometrician	Programmer	Data Scientist	Data analyst
Step 1: All-party meeting to brief the procedures	X	X	X	XX	X	
Step 2: Specification of "Q matrix" for constructs and evidences	X					
Step 3: Specification of additional evidence for psychometric modeling need		X				
Step 4: Translate the evidences into operable task codes			X	X	XX	
Step 5: Design log file structure and schema				X		
Step 6: Implement the evidence logging into the log file and tryout			X	X	XX	
Step 7: Log file parsing and evidence extraction based on the tryout data				X		
Step 8: QA the log files to check whether the recovered evidence is sufficient. If not, restart from Step 6. If everything is good, proceed to next step	X	X		X	XX	
Step 9: Finalize the data reduction pipeline and QA procedure				XX	X	
Step 10: Implement the data reduction				X	XX	

Evidence Trace File



# GlassPy Library



Not open sourced yet and under active expanding



# Interactive Analytics Frontend

Jupyter/Zeppelin Notebook/Lab + Plotly Dash/Shiny /Streamlit+ FastAPI/Flask+ others

jupyter GameLogSafari Last Checkpoint: 04/24/2017 (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 2 O

 GameLogSafari: An Interactive Analytics Frontend for GlassPy

Jiangang Hao

Educational Testing Service, Princeton, NJ 08541 [jhao@ets.org](mailto:jhao@ets.org)

Citing: Hao, J., Smith, L., Mislevy, R., von Davier, A., & Bauer, M. (2016). Taming log files from the game and simulation-based assessment: Data model and data analysis tool. ETS Research Report RR-16-11. Princeton, NJ: Educational Testing Service.

Copyright © Educational Testing Service

**1. Loading the needed python packages**

```
In [1]: %matplotlib inline
import re, mlpd3, nltk, json, warnings#, ggrid
import glasspy as gp
import numpy as np
import pandas as pd
import matplotlib.pyplot as pl
import networkx as nx
import cufflinks as cf
from ipywidgets import interact
import networkx as nx
import pygraphviz as pv
import graphviz as gv
from networkx.drawing.nx_agraph import graphviz_layout
warnings.filterwarnings('ignore')
cf.go_offline()
mlpd3.disable_notebook()
```

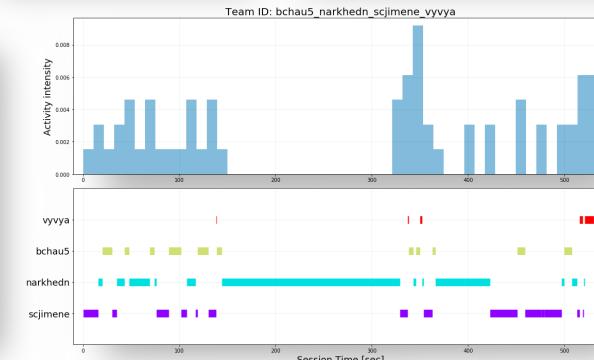
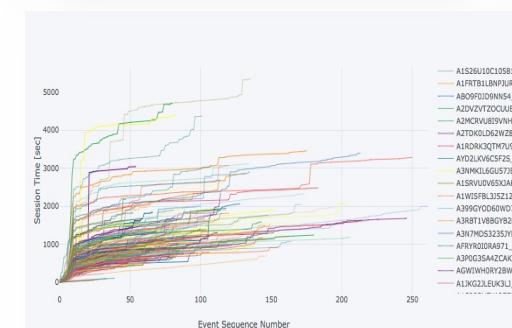
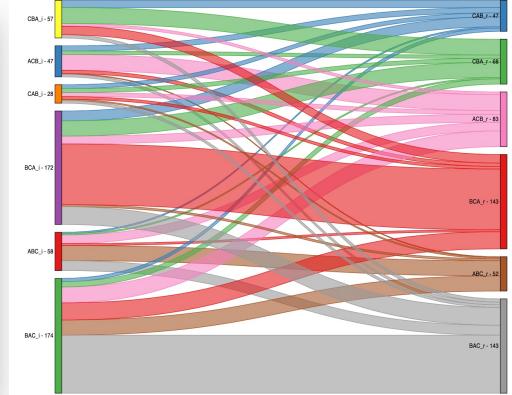
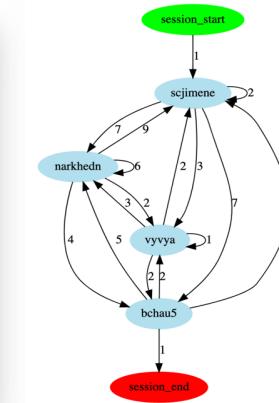
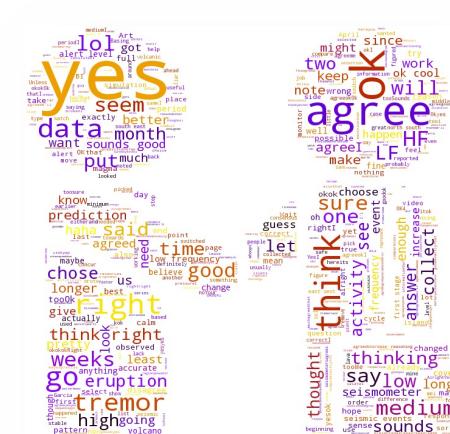
**2. Specifying the file name**

```
In [2]: #xml_file = "/Users/jhao/PycharmProjects/glasspy/exampleLogFile/simcitEDU_sampleLog_short.xml"
#xml_schema = "/Users/jhao/PycharmProjects/glasspy/schema/gamelog_schema.xsd"
json_file = "/Users/jhao/research/extended_game/434-1-80-2-1451534571.json"
json_schema = "/Users/jhao/PycharmProjects/glasspy/schema/gamelog_schema.json"
```

**3. Check and read data**

**3.1. XML Schema validation, read in data**

```
In [5]: gp.xml_validation(xml_file, xml_schema)
```

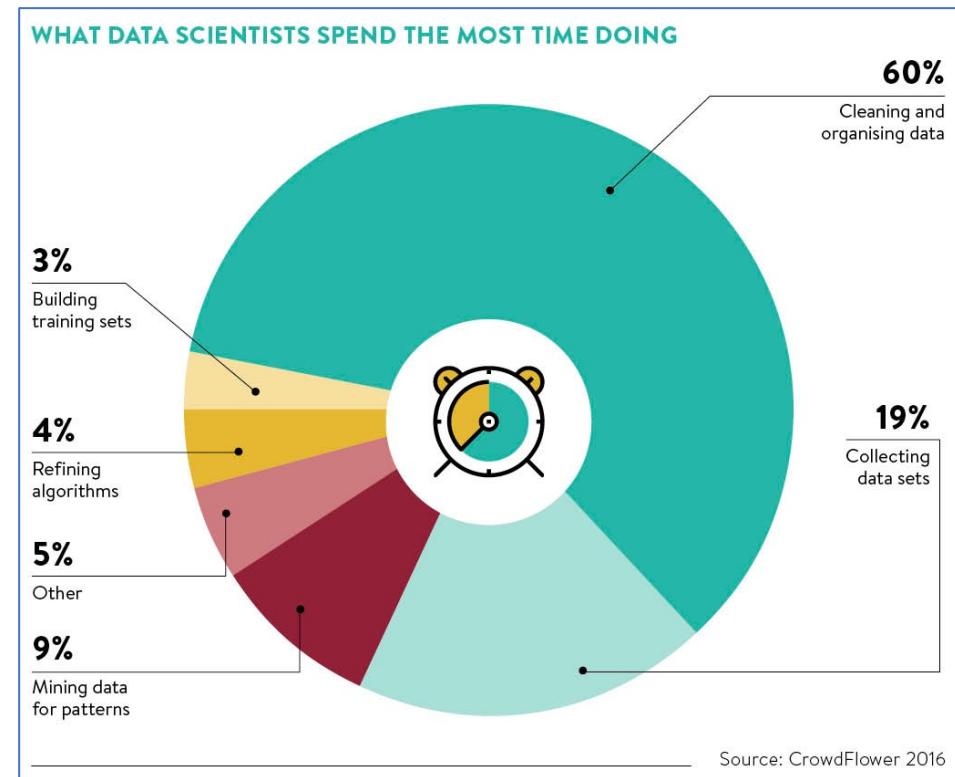




# Summary

# Take Home Message

- Data science is a teamwork, so choose the most suitable workflow for your project and your team.
- Getting data ready takes efforts, put deadlines for different deliverables.
- Spending time to plan the work will NOT slow you down
- Making sense of data is more an art than science!





# Data Wrangling

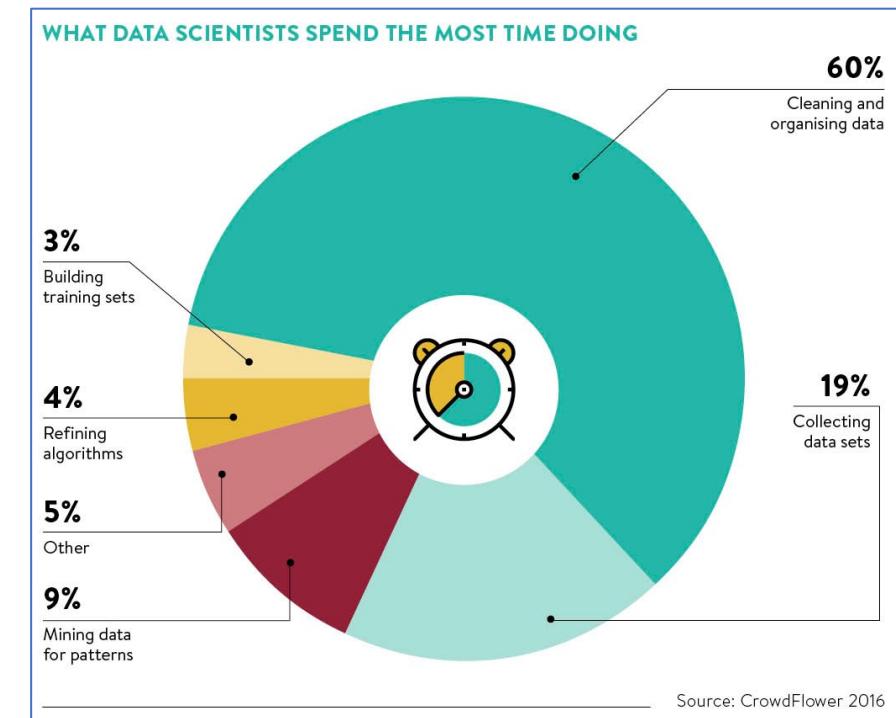


# What is Data Wrangling

# Data Wrangling

Data wrangling, also known as data munging, is the process of transforming and mapping data from one “raw” format into another format with the intent of making it more appropriate and valuable for a variety of downstream purposes such as analytics

- Wikipedia



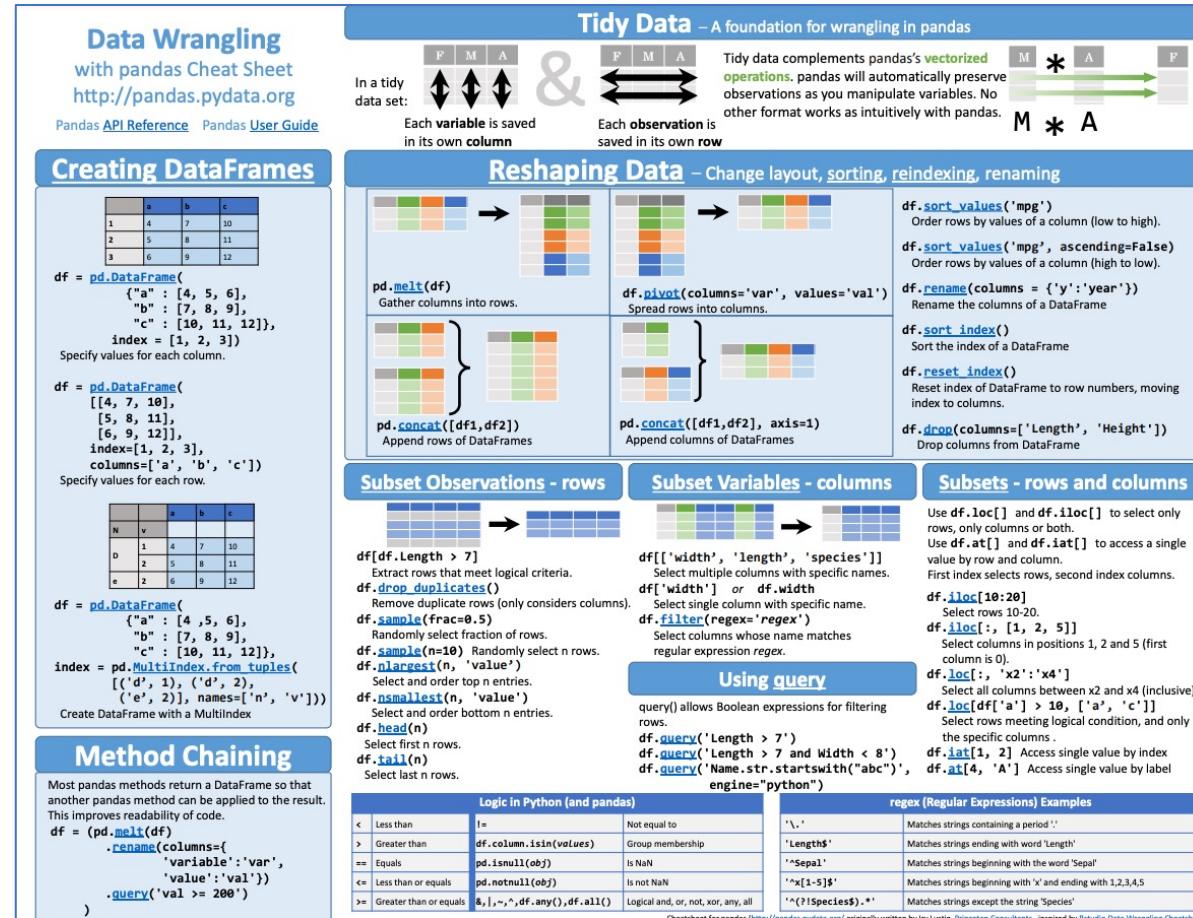
# Types of Raw Data

- **Unstructured data:** information that either does not have a predefined data model or is not organized in a pre-defined manner
- **Structured data:** information created by following a predefined data model
- **Semi-structured data** is a form of structured data that does not obey the tabular structure of data models associated with relational database or other forms of data tables, but nonetheless contains tags or other markers to separate semantic elements and enforce hierarchies of records and fields within the data. Therefore, it is also known as self-describing structure.
  - HTML, XML, JSON, YAML, etc.



# Pandas for Tabular Data Wrangling and Analysis

	<h1>pandas</h1>
<b>Original author(s)</b>	Wes McKinney
<b>Developer(s)</b>	Community
<b>Initial release</b>	11 January 2008; 13 years ago <a href="#">[citation needed]</a>
<b>Stable release</b>	1.3.0 <sup>[1]</sup> / 2 July 2021; 3 months ago
<b>Repository</b>	<a href="https://github.com/pandas-dev/pandas">github.com/pandas-dev/pandas</a> 
<b>Written in</b>	Python, Cython, C
<b>Operating system</b>	Cross-platform
<b>Type</b>	Technical computing
<b>License</b>	New BSD License
<b>Website</b>	<a href="https://pandas.pydata.org">pandas.pydata.org</a> 



[https://pandas.pydata.org/Pandas\\_Cheat\\_Sheet.pdf](https://pandas.pydata.org/Pandas_Cheat_Sheet.pdf)





# Unstructured Data

# Unstructured text file

```
[5/15/2013 2:17:26 PM] Session Start
[5/15/2013 2:17:26 PM] Leaving sequence: loadXML, moving forward.
[5/15/2013 2:17:30 PM] Player submitted name: Carl
[5/15/2013 2:17:30 PM] Leaving sequence: InputNameScreen, moving forward.
[5/15/2013 2:17:31 PM] Player submitted name: Carl
[5/15/2013 2:17:31 PM] Leaving sequence: startScreen, moving forward.
[5/15/2013 2:17:50 PM] Player submitted name: Carl
[5/15/2013 2:17:50 PM] Leaving sequence: slide2, moving forward.
[5/15/2013 2:17:55 PM] Player submitted name: Carl
[5/15/2013 2:17:55 PM] Leaving sequence: slide2b, moving forward.
[5/15/2013 2:18:34 PM] Player submitted name: Carl
[5/15/2013 2:18:34 PM] Leaving sequence: slide2c, moving forward.
[5/15/2013 2:20:09 PM] Player submitted name: Carl
[5/15/2013 2:20:09 PM] Leaving sequence: slide3, moving forward.
[5/15/2013 2:20:13 PM] Player submitted name: Carl
[5/15/2013 2:20:13 PM] Leaving sequence: slide4, moving forward.
```

Demo of parsing unstructured file using Python





# Structured Data

XML, JSON

# XML - Extensible Markup Language

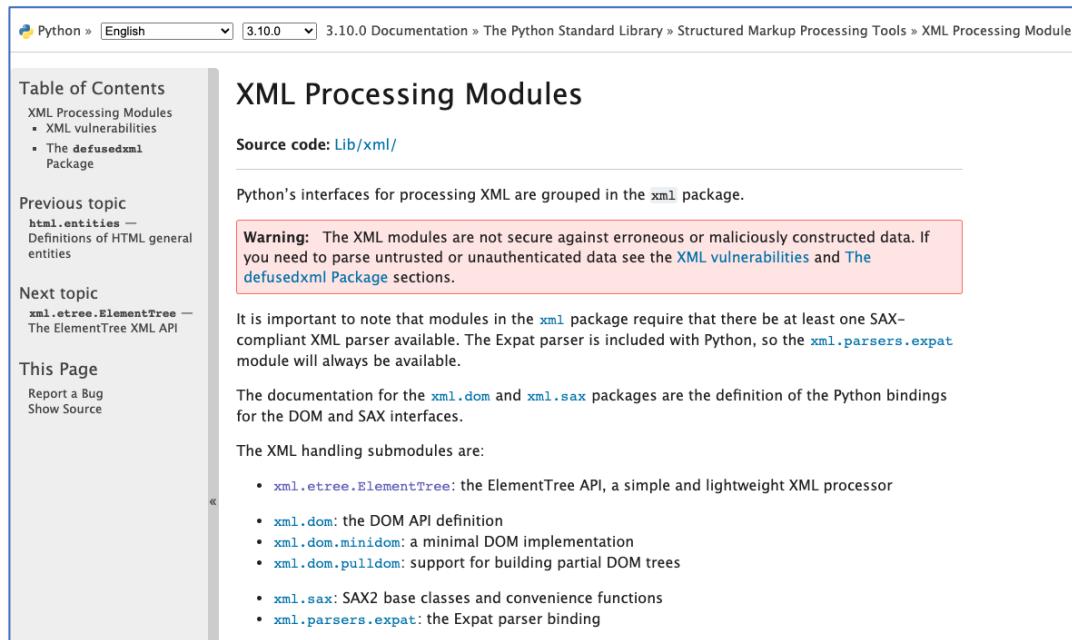
- XML -W3C 1998
  - Markup and contents
    - <start\_time> 10:10 </start\_time>
    - <![CDATA[ this is the comments ]]>
  - Tag
    - Start-tag <shape>
    - End-tag </shape>
    - Empty-tag <shape/>
  - Elements
    - <start\_time> 10:10 </start\_time>
  - Attributes: name-value pair
    - <start\_time timezone="EST"> 10:10 </start\_time>
  - Declaration: <?xml version="1.0" encoding="UTF-8"?>



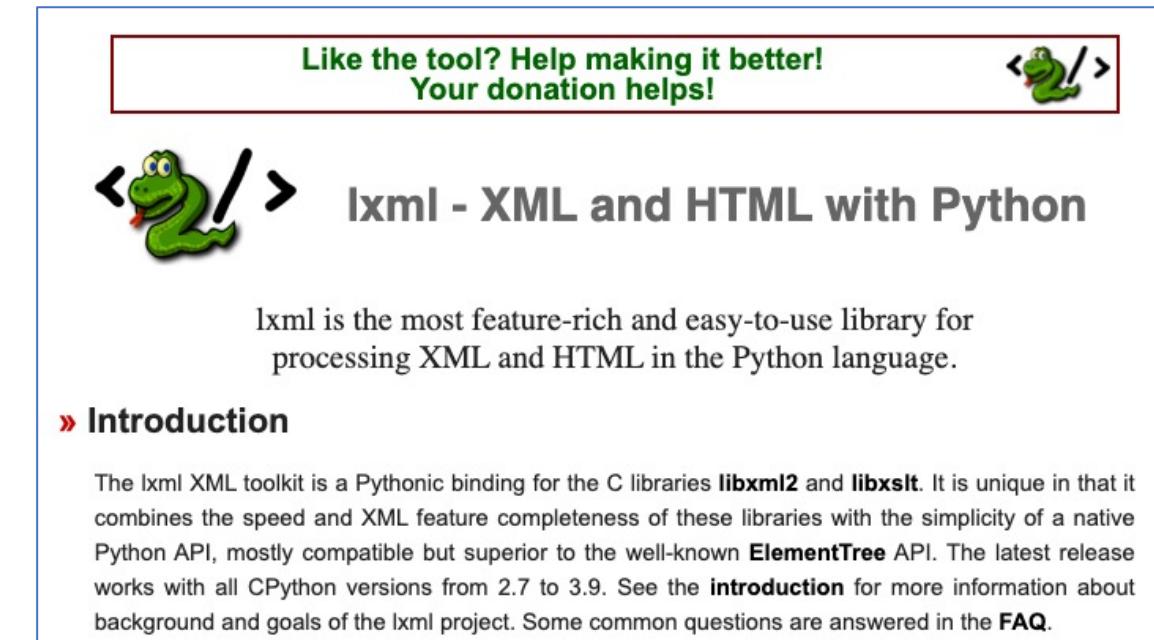
Data from EPCAL

# Python Tool for XML

- “Official” XML library:  
<https://docs.python.org/3/library/xml.html>
- LXML: <https://lxml.de/>



The screenshot shows the Python 3.10.0 Documentation page for the XML Processing Modules. The URL is https://docs.python.org/3.10.0/3.10.0/Documentation/The\_Python\_Standard\_Library/Structured\_Markup\_Processing\_Tools/XML\_Processing\_Modules. The page title is "XML Processing Modules". It includes a "Table of Contents" sidebar with links to XML Processing Modules, XML vulnerabilities, and defusedxml. The main content area has a "Source code: Lib/xml/" link and a note about Python's interfaces for processing XML being grouped in the `xml` package. A red warning box states: "Warning: The XML modules are not secure against erroneous or maliciously constructed data. If you need to parse untrusted or unauthenticated data see the [XML vulnerabilities](#) and [The defusedxml Package](#) sections." Below this, it notes that at least one SAX-compliant XML parser is required, mentioning the Expat parser included with Python. It also mentions the `xml.dom` and `xml.sax` packages for DOM and SAX interfaces, and lists submodules like `xml.etree.ElementTree`, `xml.dom`, `xml.dom.minidom`, `xml.dom.pulldom`, `xml.sax`, and `xml.parsers.expat`.



The screenshot shows the LXML project homepage. The URL is https://lxml.de/. The page features a green banner with the text "Like the tool? Help making it better! Your donation helps!" and a green snake icon. The main title is "lxml - XML and HTML with Python". It describes lxml as a feature-rich and easy-to-use library for processing XML and HTML in Python, using the C libraries libxml2 and libxslt. The "Introduction" section provides background on the project, stating it is a Pythonic binding for libxml2 and libxslt, combining speed, completeness, and simplicity. It is compatible with Python versions 2.7 to 3.9.



# JSON

## JavaScript Object Notation



Filename extension	.json
Internet media type	application/json
Type code	TEXT
Uniform Type Identifier (UTI)	public.json
Type of format	Data interchange
Extended from	JavaScript
Standard	STD 90  (RFC 8259  json.org <img alt="link icon" data-bbox="648 938 678 968/</a>

Introduced in 2000

# Valid Data Types

In JSON, values must be one of the following data types:

- a string
  - a number
  - an object (JSON object)
  - an array
  - a boolean
  - *null*

JSON values **cannot** be one of the following data types:

- a function
  - a date
  - *undefined*

<https://www.w3schools.com/js/>



# JSON vs. XML

## JSON is Like XML Because

- Both JSON and XML are "self describing" (human readable)
- Both JSON and XML are hierarchical (values within values)
- Both JSON and XML can be parsed and used by lots of programming languages
- Both JSON and XML can be fetched with an XMLHttpRequest

## JSON is Unlike XML Because

- JSON doesn't use end tag
- JSON is shorter
- JSON is quicker to read and write
- JSON can use arrays

The biggest difference is:

XML has to be parsed with an XML parser. JSON can be parsed by a standard JavaScript function.

## Why JSON is Better Than XML

XML is much more difficult to parse than JSON.  
JSON is parsed into a ready-to-use JavaScript object.

Both XML and JSON are widely used for instantiation of document data models





# Schema: Define and Validate Data

# XML Schema

- XML schema: define the structure and data type of a xml file
- Practical use: validation- comparing the xml file to the schema to check for compliance

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">

<!-- Game Log Schema
Version 2.0
Authors:
  Lonnie Smith (lsmith@ets.org)
  Jiangang Hao (jhao@ets.org)

Note: this is the final version as of 7/27/2015

(c) 2014, Educational Testing Service
--&gt;

<!-- Root element and children --&gt;
&lt;xs:element name="gameLog"&gt;
  &lt;xs:complexType&gt;
    &lt;xs:sequence&gt;
      &lt;xs:element name="session" minOccurs="1" maxOccurs="unbounded"&gt;
        &lt;xs:complexType&gt;
          &lt;xs:sequence&gt;
            &lt;xs:element name="sessionId" type="idType" minOccurs="1" maxOccurs="1"/&gt;
            &lt;xs:element name="teamID" type="idType" minOccurs="1" maxOccurs="1"/&gt;
            &lt;xs:element name="playerID" type="dictType" minOccurs="1" maxOccurs="1"/&gt;
            &lt;xs:element name="attemptID" type="idType" minOccurs="1" maxOccurs="1"/&gt;
            &lt;xs:element name="sessionExtData" type="dictType" minOccurs="0" maxOccurs="1"/&gt;
            &lt;xs:element name="eventSequence" minOccurs="1" maxOccurs="1"&gt;
              &lt;xs:complexType&gt;
                &lt;xs:sequence&gt;
                  &lt;xs:element name="event" type="eventType" minOccurs="1" maxOccurs="unbounded"/&gt;
                &lt;/xs:sequence&gt;
              &lt;/xs:complexType&gt;
            &lt;/xs:element&gt;
          &lt;/xs:sequence&gt;
        &lt;/xs:complexType&gt;
      &lt;/xs:element&gt;
    &lt;/xs:sequence&gt;
  &lt;/xs:complexType&gt;
&lt;/xs:element&gt;

<!-- Data type definitions --&gt;

<!-- ID definition. All identifiers must follow this rule --&gt;
&lt;xs:simpleType name="idType"&gt;
  &lt;xs:restriction base="xs:string"&gt;
    &lt;xs:pattern value="[a-zA-Z0-9_]{1,}&gt;
  &lt;/xs:restriction&gt;
&lt;/xs:simpleType&gt;

<!-- Timestamps must follow subset of ISO 8601 standard, be resolved to (at least) the milisecond, and must use UTC --&gt;
&lt;xs:simpleType name="timestampType"&gt;
  &lt;xs:restriction base="xs:dateTime"&gt;
    &lt;xs:pattern value="20\d{2}-\d{2}-\d{2}\T\d{2}:\d{2}:\d{2}Z"/&gt;
  &lt;/xs:restriction&gt;
&lt;/xs:simpleType&gt;</pre>
```

Example schema from ETS VPA data model



# JSON Schema

- Define JSON file/string
- Can be used to validate JSON file

```
{  
    "$schema": "http://json-schema.org/draft-04/schema#",  
    "description": "Gamelog Schema v1.2, Created by Jiangang Hao @ ETS",  
    "type": "object",  
    "properties": {  
        "gameLog": {  
            "type": "array",  
            "items": {  
                "type": "object",  
                "properties": {  
                    "sessionID": {  
                        "type": "string",  
                        "pattern": "[a-zA-Z0-9_\\-]{1,}"  
                    },  
                    "playerID": {  
                        "type": "string",  
                        "pattern": "[a-zA-Z0-9_\\-]{1,}"  
                    },  
                    "attemptID": {  
                        "type": "integer"  
                    },  
                    "sessionExtData": {  
                        "type": "object"  
                    },  
                    "eventSequence": {  
                        "type": "array",  
                        "items": {  
                            "type": "object",  
                            "properties": {  
                                "eventName": {  
                                    "type": "string",  
                                    "pattern": "[a-zA-Z0-9_\\-]{1,}"  
                                },  
                                "eventStartTime": {  
                                    "type": "string",  
                                    "pattern": "20\\d{2}-\\d{2}-\\d{2}T\\d{2}:\\d{2}:\\d{2}Z"  
                                },  
                                "eventEndTime": {  
                                    "type": "string",  
                                    "pattern": "20\\d{2}-\\d{2}-\\d{2}T\\d{2}:\\d{2}:\\d{2}Z"  
                                },  
                            }  
                        }  
                    }  
                }  
            }  
        }  
    }  
}
```

Demo of parsing structured file using Python





# Hands-on Exercise

# Lab Study- Data from ETS VPA

**ETS Platform for Collaborative Assessment and Learning**

E95 will complete Trial 4. The numbers 0-9 have been coded as the letters A-J in some random order. Try to find out which letter matches which number as efficiently as possible, using only addition and subtraction. Your response will appear in red, and the computer feedback will be underlined in black. Only a response with all 10 letters correctly matched with numbers will receive "Correct" as feedback.

You Respond 68% Time left: 01:56 SUBMIT

Enter your guesses for some or all the letters (each number 0 to 9 can only be used once)

Trial	Equation	Hypothesis	Feedback	A	B	C	D	E	F	G	H	I	J	Feedback
1	$A+A=F$	$F=1$	True						1					Incorrect
2	$A-A=E$	$E=0$	True		5				0	1				Incorrect
3	$AE+AE+AE+AE+AE-FE-F-F$ $F=F=CDH$	$H=6$	True	5	2	3	0	1		6				Incorrect
4	$AEE-CE-F=BGI$	$J=9$	True	5	4	2	3	0	1	7	6	8	9	Incorrect
5														
6														
7														
8														
9														
10														

You can write notes to yourself here. They will stay here until the problem is finished.  
500-20-1=479

Type to chat... Send

```

▼ 0:
  sessionID: "7369"
  teamID: "hao_jiangang"
  ▶ playerID: [...]
  attemptID: 17
  ▶ sessionExtData: {...}
  ▶ eventSequence:
    ▼ 0:
      eventName: "chat"
      eventStartTime: "2019-11-06T14:18:31Z"
      eventEndTime: "2019-11-06T14:18:31Z"
      eventBy: "jiangang"
      eventTo: "others"
      eventResult: "hi"
      eventLocation: "slide1-step0"
      eventExtData: {}

    ▼ 1:
      eventName: "question"
      eventStartTime: "2019-11-06T14:18:42Z"
      eventEndTime: "2019-11-06T14:18:42Z"
      eventBy: "jiangang"
      eventTo: "cbal-1-0"
      eventResult: "1"
      eventLocation: "slide2-step0"
      eventExtData: {}

    ▼ 2:
      eventName: "question"
      eventStartTime: "2019-11-06T14:18:42Z"
      eventEndTime: "2019-11-06T14:18:42Z"
      eventBy: "jiangang"
      eventTo: "cbal-1-1"
      eventResult: "dfsa"
      eventLocation: "slide2-step0"
      eventExtData: {}
  
```





# Data Visualization

# Steps to Visualization

- Understand the nature of your data
  - Categorical/continuous
  - Sparse
- Understand your goal
  - Audience
  - Production/exploration?
  - Interactivity required?
- Plan the types of visualizations you'll create
  - Scatter plot, distribution, dendrogram, etc.
- Choose a visualization framework(s)
- Do it!

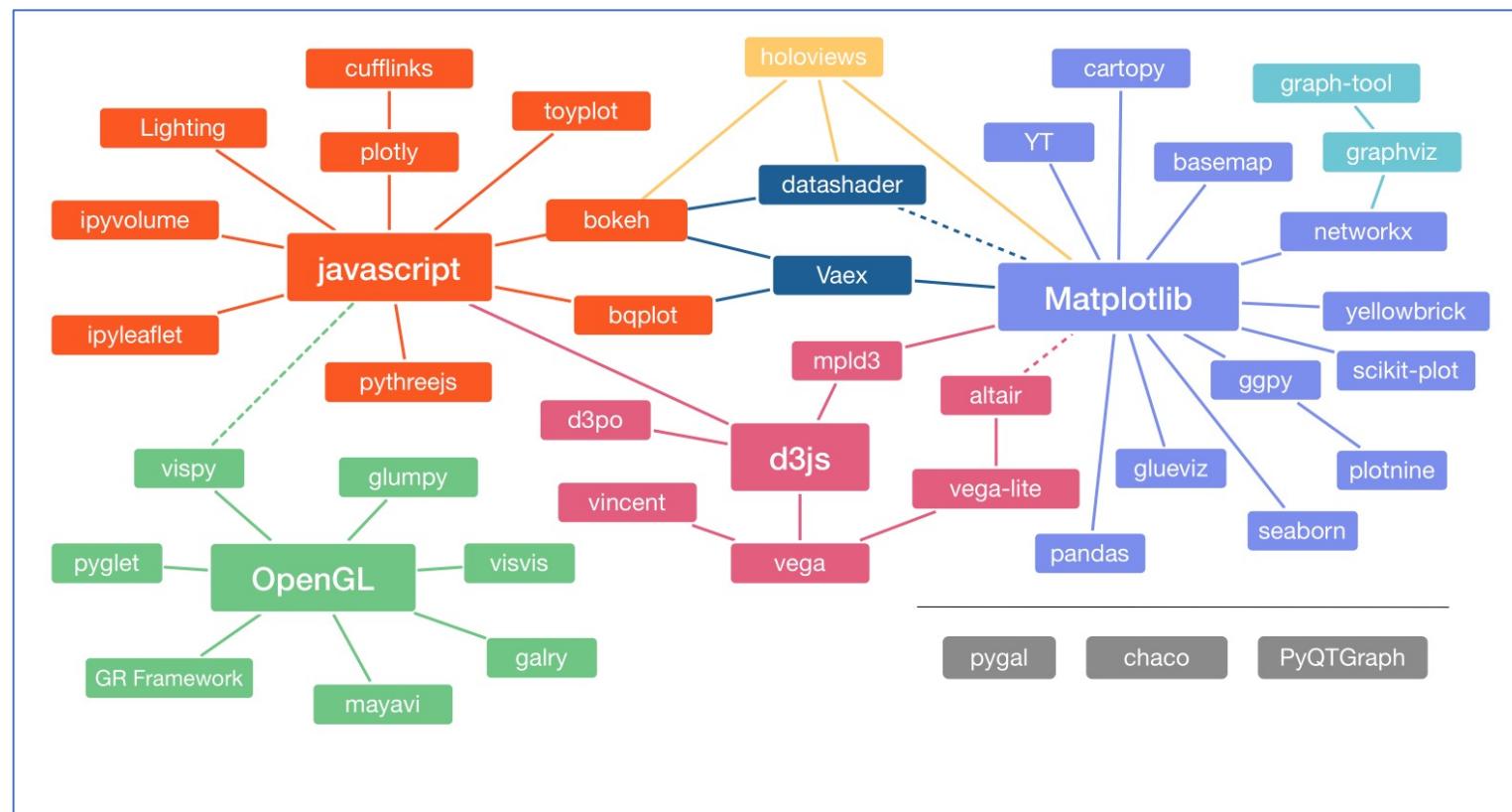


<https://chart.guide/topics/chartguide-poster-4-0/>



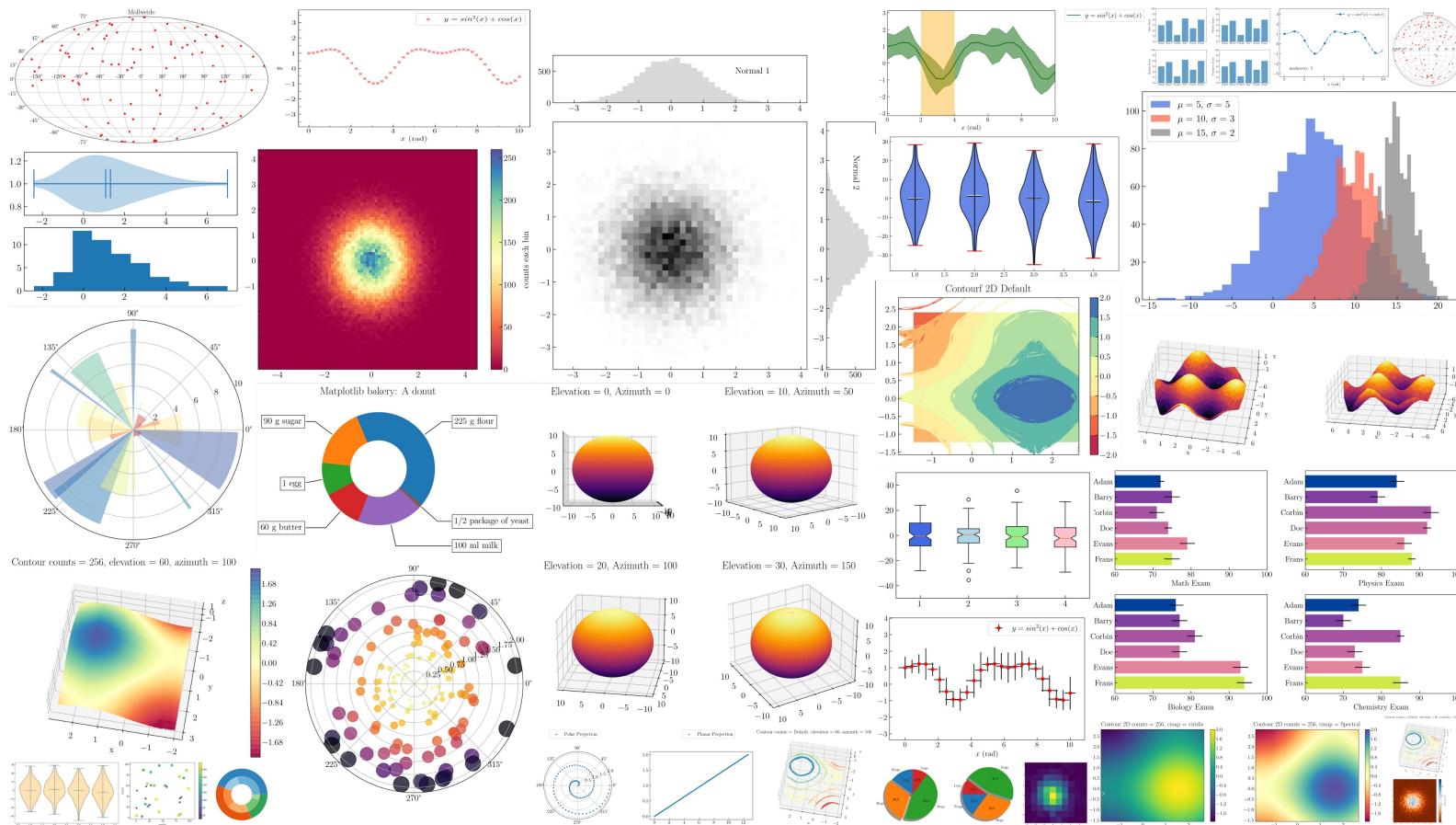
# Visualization in Python

- PyViz is your gateway: [www.pyviz.org](http://www.pyviz.org)
- Visualization paradigms



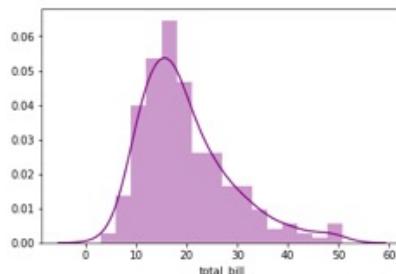
# Matplotlib

<https://matplotlib.org/>

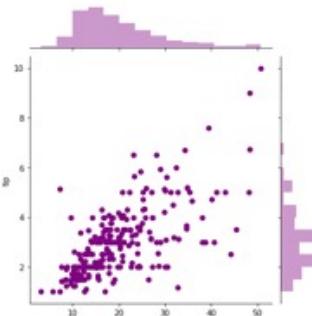


# Seaborn: built on top of matplotlib

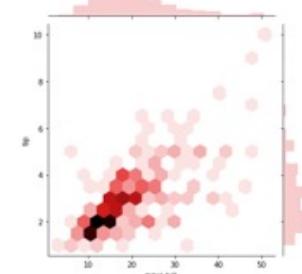
<https://seaborn.pydata.org/>



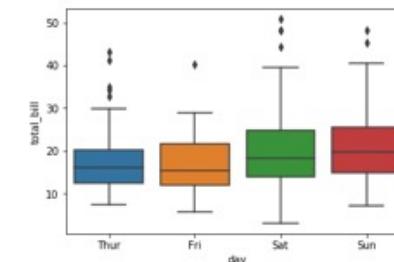
distplot



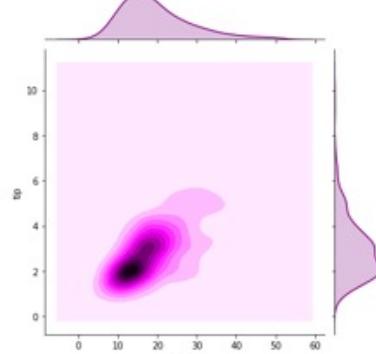
Jointplot



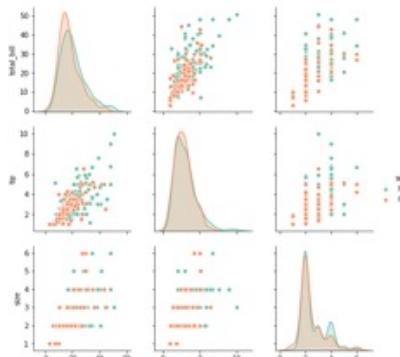
Hexplots



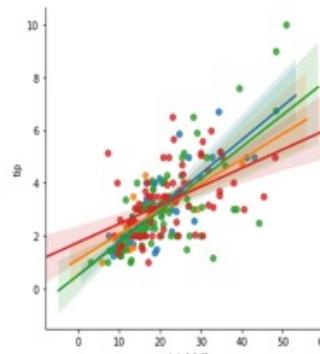
Boxplots



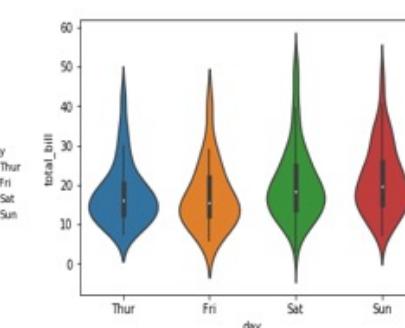
KDE Plot



Pair Plots



LM Plots



Violin Plots

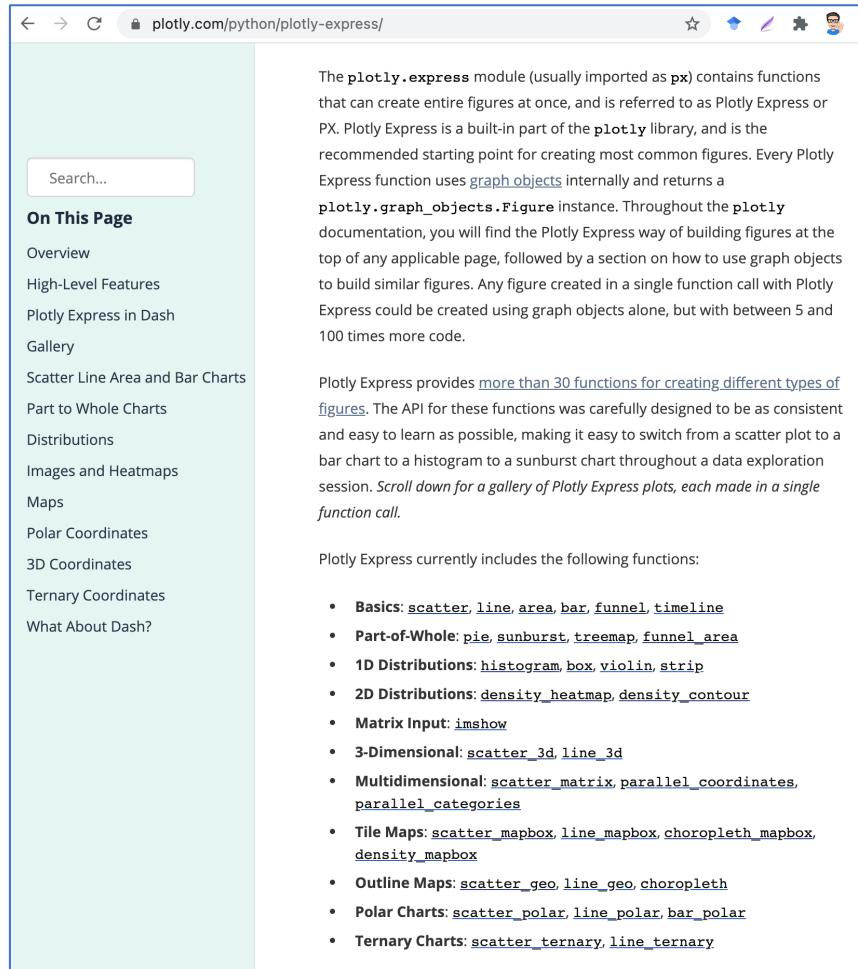




# Interactive Visualization



# Plotly Express and ipywidgets



The screenshot shows the official Plotly Express documentation page at [plotly.com/python/plotly-express/](https://plotly.com/python/plotly-express/). The page includes a search bar, a sidebar titled "On This Page" with links to various chart types and features, and the main content area which describes the `plotly.express` module and lists its functions.

**On This Page**

- Overview
- High-Level Features
- Plotly Express in Dash
- Gallery
- Scatter Line Area and Bar Charts
- Part to Whole Charts
- Distributions
- Images and Heatmaps
- Maps
- Polar Coordinates
- 3D Coordinates
- Ternary Coordinates
- What About Dash?

**Search...**

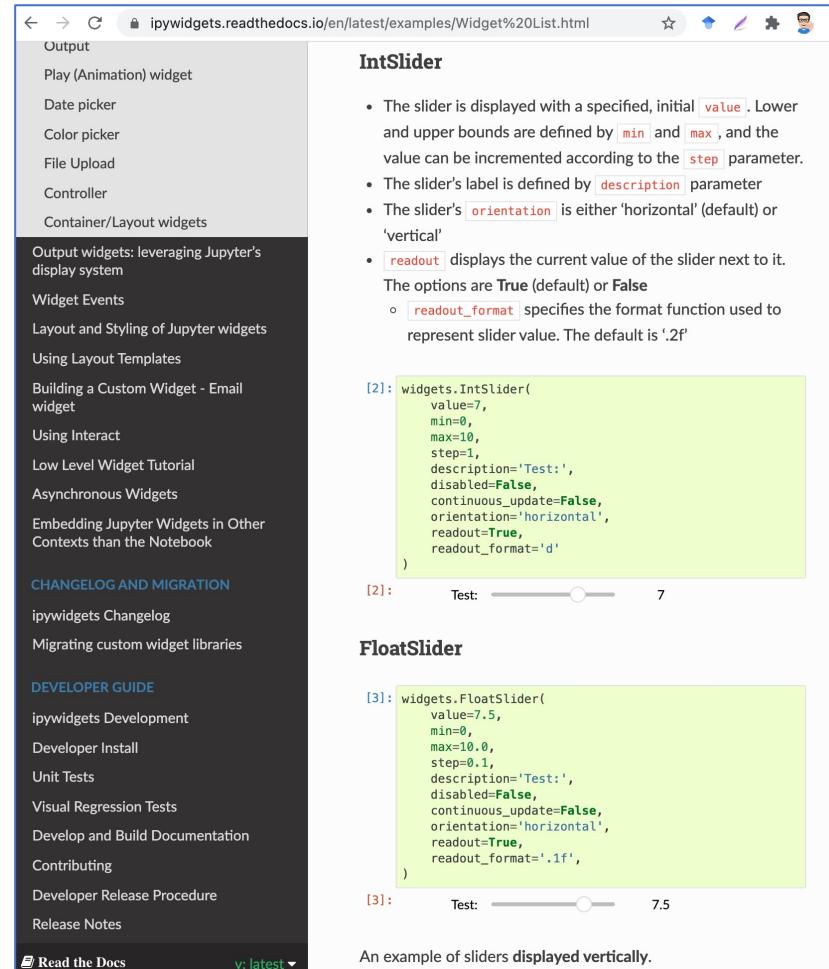
**Plotly Express**

The `plotly.express` module (usually imported as `px`) contains functions that can create entire figures at once, and is referred to as Plotly Express or PX. Plotly Express is a built-in part of the `plotly` library, and is the recommended starting point for creating most common figures. Every Plotly Express function uses `graph_objects` internally and returns a `plotly.graph_objects.Figure` instance. Throughout the `plotly` documentation, you will find the Plotly Express way of building figures at the top of any applicable page, followed by a section on how to use graph objects to build similar figures. Any figure created in a single function call with Plotly Express could be created using graph objects alone, but with between 5 and 100 times more code.

Plotly Express provides [more than 30 functions for creating different types of figures](#). The API for these functions was carefully designed to be as consistent and easy to learn as possible, making it easy to switch from a scatter plot to a bar chart to a histogram to a sunburst chart throughout a data exploration session. *Scroll down for a gallery of Plotly Express plots, each made in a single function call.*

Plotly Express currently includes the following functions:

- **Basics:** `scatter`, `line`, `area`, `bar`, `funnel`, `timeline`
- **Part-of-Whole:** `pie`, `sunburst`, `treemap`, `funnel_area`
- **1D Distributions:** `histogram`, `box`, `violin`, `strip`
- **2D Distributions:** `density_heatmap`, `density_contour`
- **Matrix Input:** `imshow`
- **3-Dimensional:** `scatter_3d`, `line_3d`
- **Multidimensional:** `scatter_matrix`, `parallel_coordinates`, `parallel_categories`
- **Tile Maps:** `scatter_mapbox`, `line_mapbox`, `choropleth_mapbox`, `density_mapbox`
- **Outline Maps:** `scatter_geo`, `line_geo`, `choropleth`
- **Polar Charts:** `scatter_polar`, `line_polar`, `bar_polar`
- **Ternary Charts:** `scatter_ternary`, `line_ternary`



The screenshot shows the ipywidgets documentation page at [ipywidgets.readthedocs.io/en/latest/examples/Widget%20List.html](https://ipywidgets.readthedocs.io/en/latest/examples/Widget%20List.html). It includes a sidebar with links to various widget examples and developer guides, and two examples of sliders.

**Output**

- Play (Animation) widget
- Date picker
- Color picker
- File Upload
- Controller
- Container/Layout widgets

**IntSlider**

- The slider is displayed with a specified, initial `value`. Lower and upper bounds are defined by `min` and `max`, and the value can be incremented according to the `step` parameter.
- The slider's label is defined by `description` parameter
- The slider's `orientation` is either 'horizontal' (default) or 'vertical'
- `readout` displays the current value of the slider next to it. The options are `True` (default) or `False`
  - `readout_format` specifies the format function used to represent slider value. The default is `'.2f'`

```
[2]: widgets.IntSlider(
    value=7,
    min=0,
    max=10,
    step=1,
    description='Test:',
    disabled=False,
    continuous_update=False,
    orientation='horizontal',
    readout=True,
    readout_format='d'
)
```

[2]: Test:  7

**FloatSlider**

```
[3]: widgets.FloatSlider(
    value=7.5,
    min=0,
    max=10.0,
    step=0.1,
    description='Test:',
    disabled=False,
    continuous_update=False,
    orientation='horizontal',
    readout=True,
    readout_format='.1f',
)
```

[3]: Test:  7.5

An example of sliders displayed vertically.

Demos



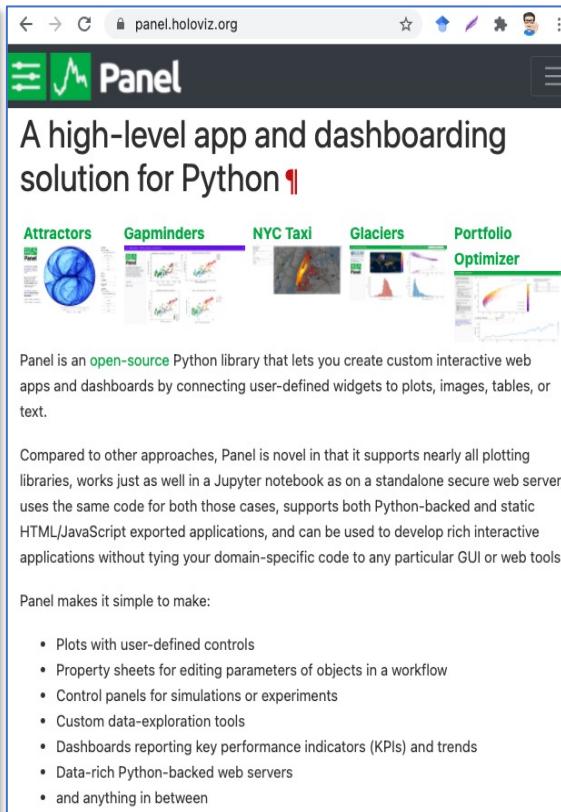


# Dashboarding

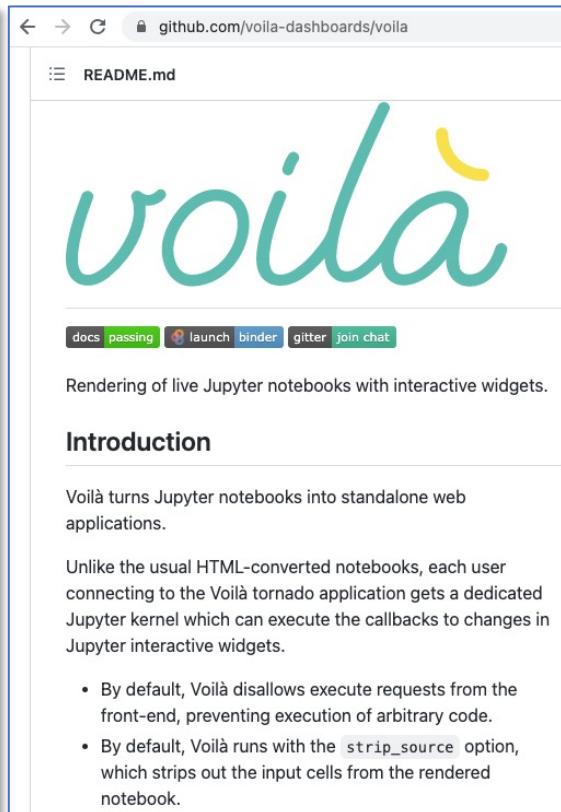
# Major Dashboard Tools in Python



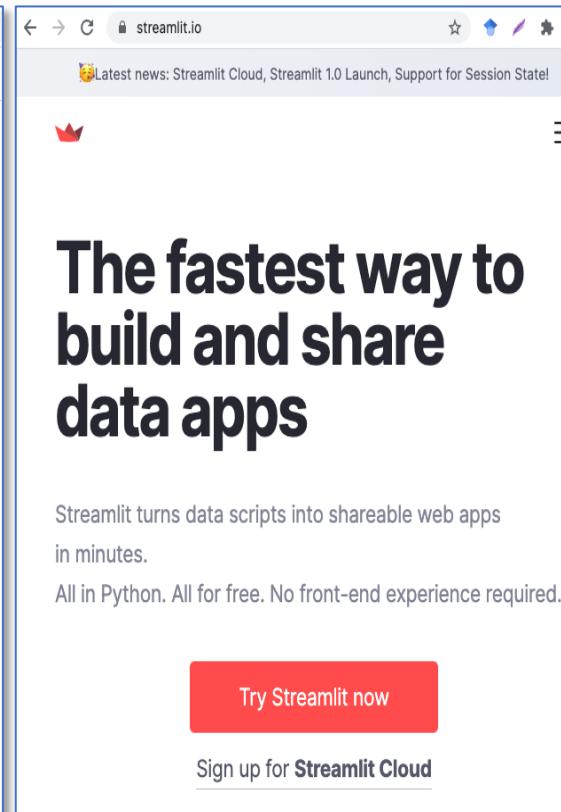
The screenshot shows the official Plotly Dash website. At the top, there's a navigation bar with links for 'Dash', 'Low-Code Development', and 'Deployment & Scaling'. Below this, a section titled 'Overview of Dash & Dash Apps' is displayed. It explains that Dash apps provide a point-and-click interface for Python, R, and Julia. A paragraph describes how Dash Enterprise helps deploy and scale these apps. At the bottom, there's a testimonial from a company named 'Futura'.



The screenshot shows the Panel website. It features a large heading 'A high-level app and dashboarding solution for Python' with a small icon. Below it are five thumbnail images of different dashboard examples: 'Attractors', 'Gapminders', 'NYC Taxi', 'Glaciers', and 'Portfolio Optimizer'. A descriptive paragraph follows, explaining that Panel is an open-source Python library for creating custom interactive web apps and dashboards. It compares Panel to other approaches like Jupyter notebooks, highlighting its support for nearly all plotting libraries and its ability to work as a standalone web server. A list of features is provided at the bottom.



The screenshot shows the Voila website. It has a large teal 'voila' logo with a yellow exclamation mark. Below the logo are links for 'docs', 'passing', 'launch', 'binder', 'gitter', and 'join chat'. A section titled 'Introduction' explains that Voila turns Jupyter notebooks into standalone web applications. Another section details how Voila executes callbacks from Jupyter interactive widgets. A bulleted list at the bottom provides more information about its execution behavior.

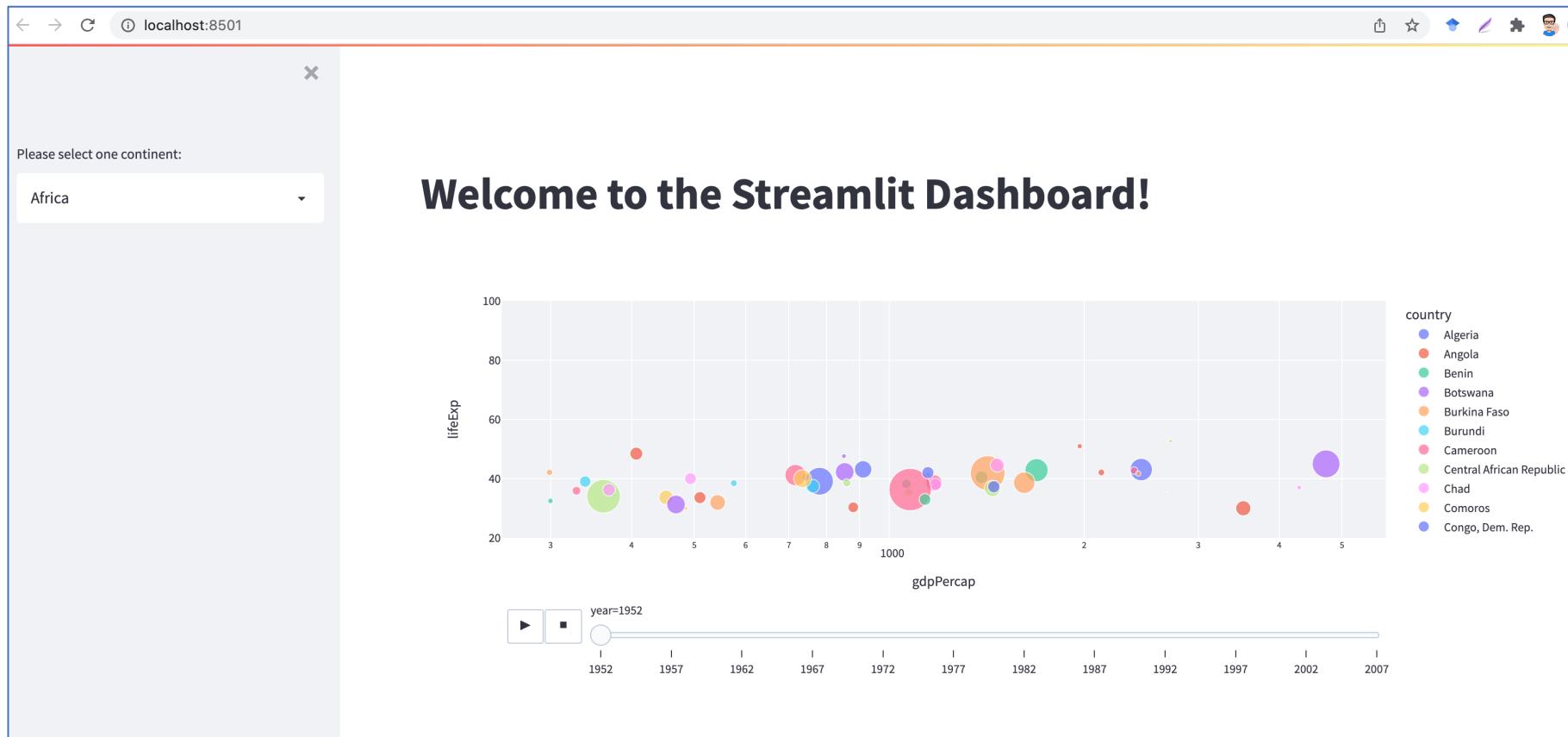


The screenshot shows the Streamlit website. It features a large heading 'The fastest way to build and share data apps'. Below this is a testimonial from a user named 'David' with a small profile picture. A paragraph explains that Streamlit turns data scripts into shareable web apps in minutes. Another paragraph states that Streamlit is built entirely in Python and requires no front-end experience. At the bottom, there are two calls-to-action: a red button for 'Try Streamlit now' and a link for 'Sign up for Streamlit Cloud'.

They are kind of equivalent to the Shiny package in R



# Examples



Demo





# Hands-on Lab

# Lab

- Create visualizations using matplotlib, seaborn, plotly and ipywidgets using the dataset
  - Histogram, scatter plot, pie, others
- Create one streamlit dashboard

