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…

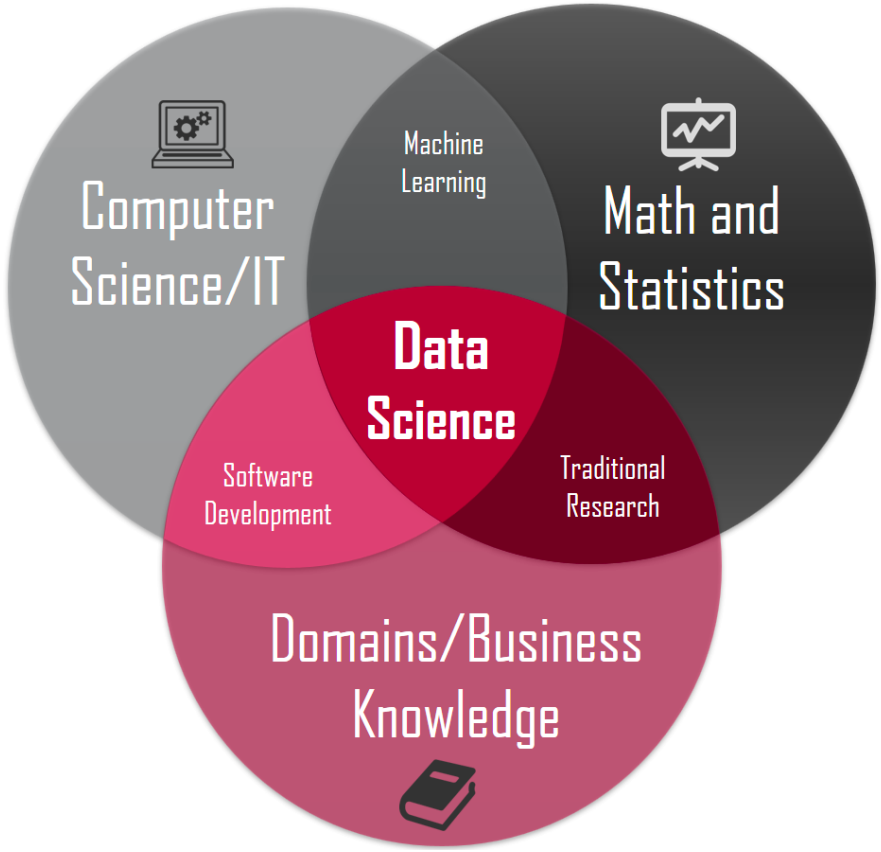
- Data science basics
  - Data science overview
  - Data storage – data lake, warehouse, mart
  - Data model – document, relational, graphic
  - Data processing
- Data wrangling
  - Parsing **XML and JSON structured formats**
  - Parsing **unstructured** files
- Interactive visualization
- Dashboard

What is Data Science

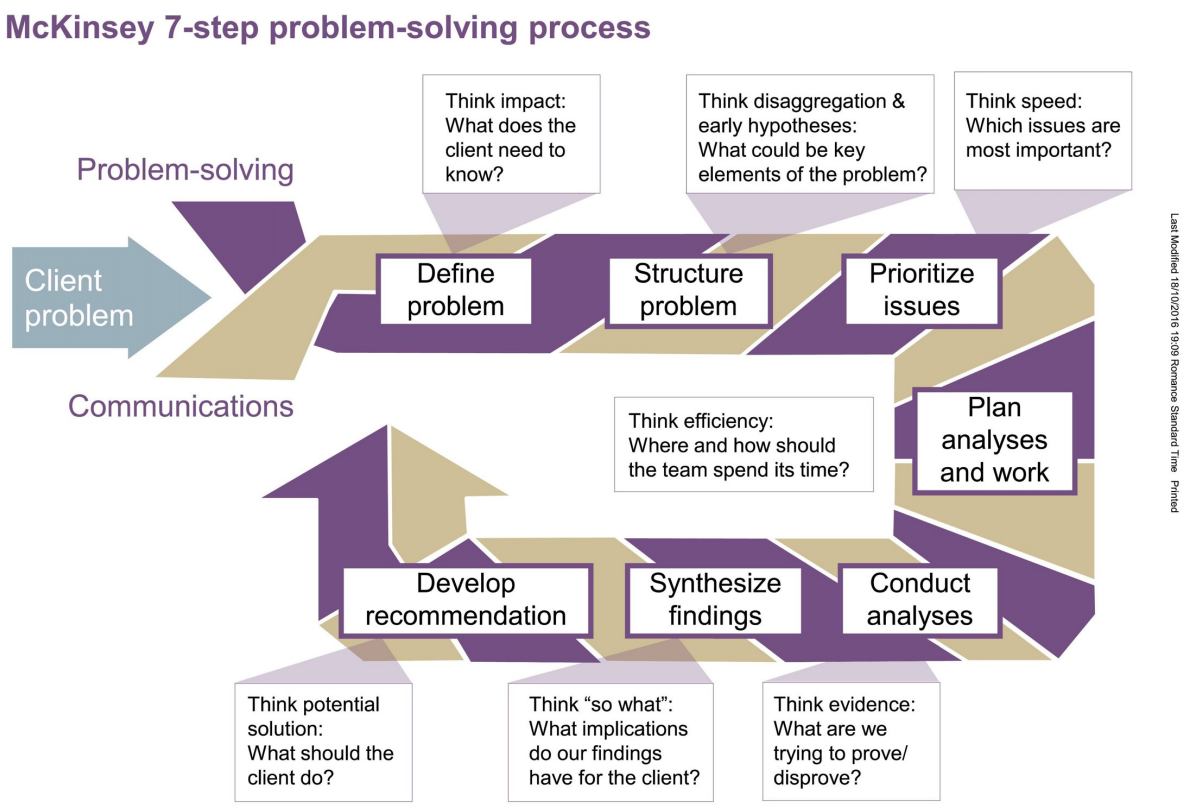
What is Data Science

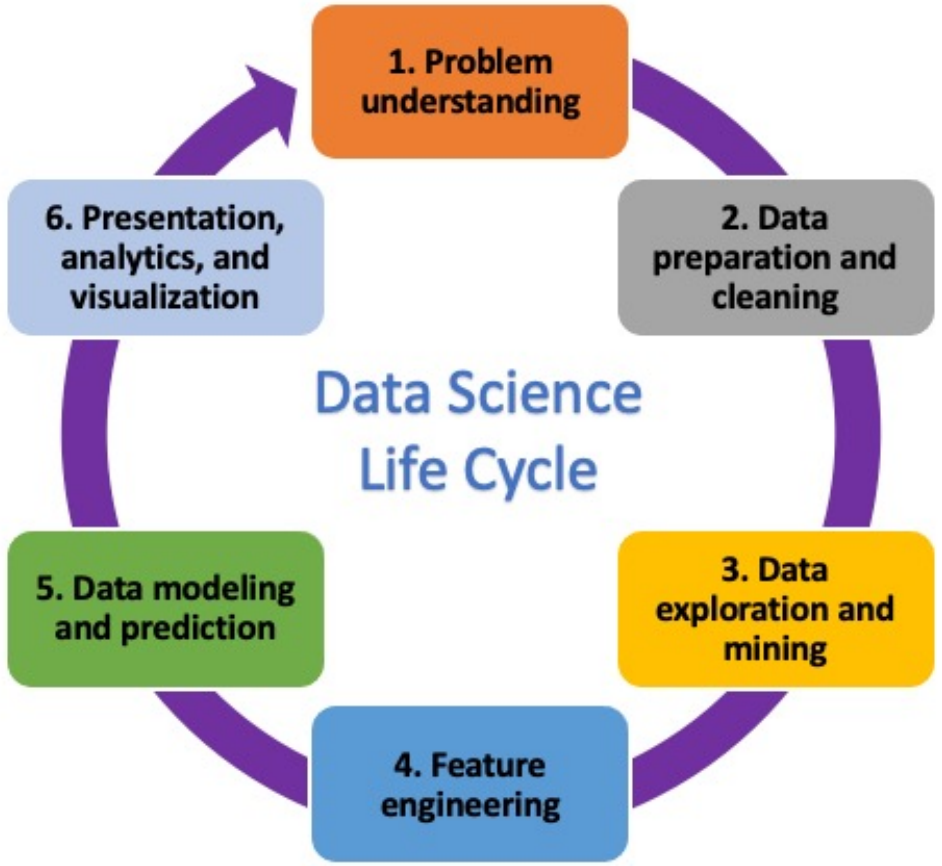
Data science is an inter-disciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from many structural and unstructured data

- Wikipedia

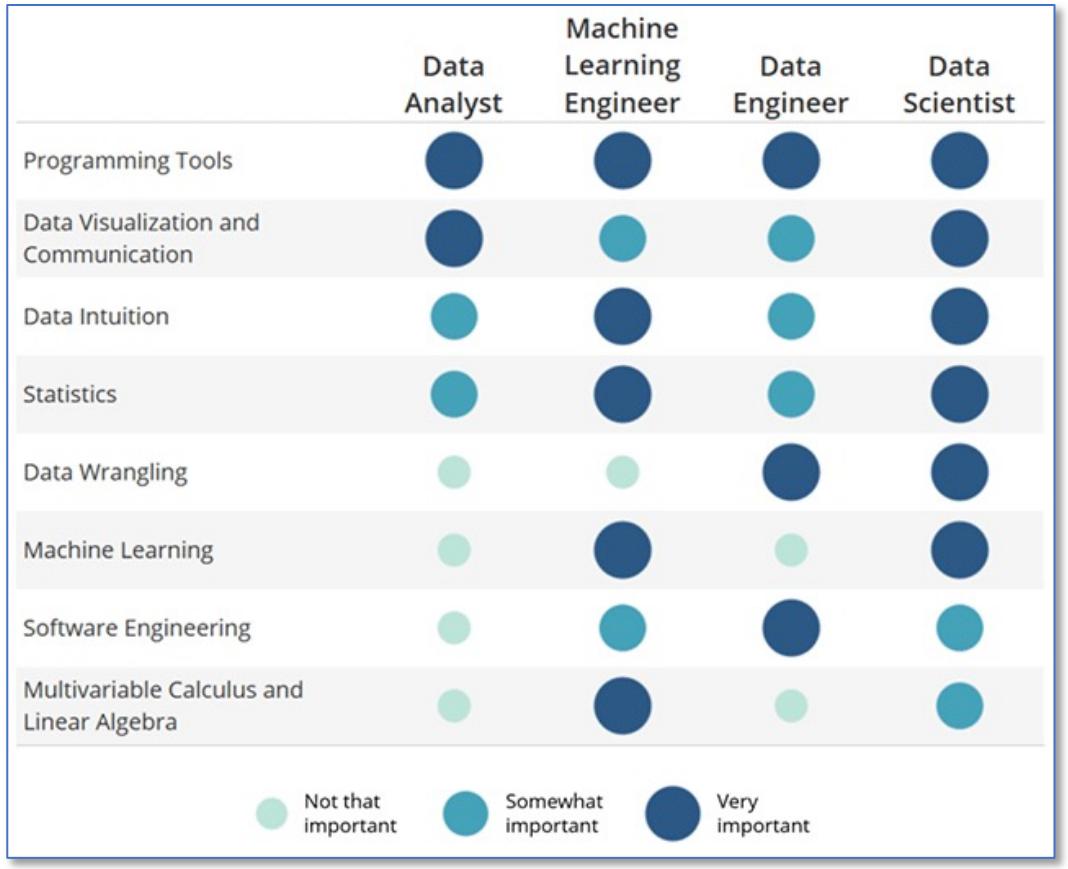


Life Cycle of Data Science Projects





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

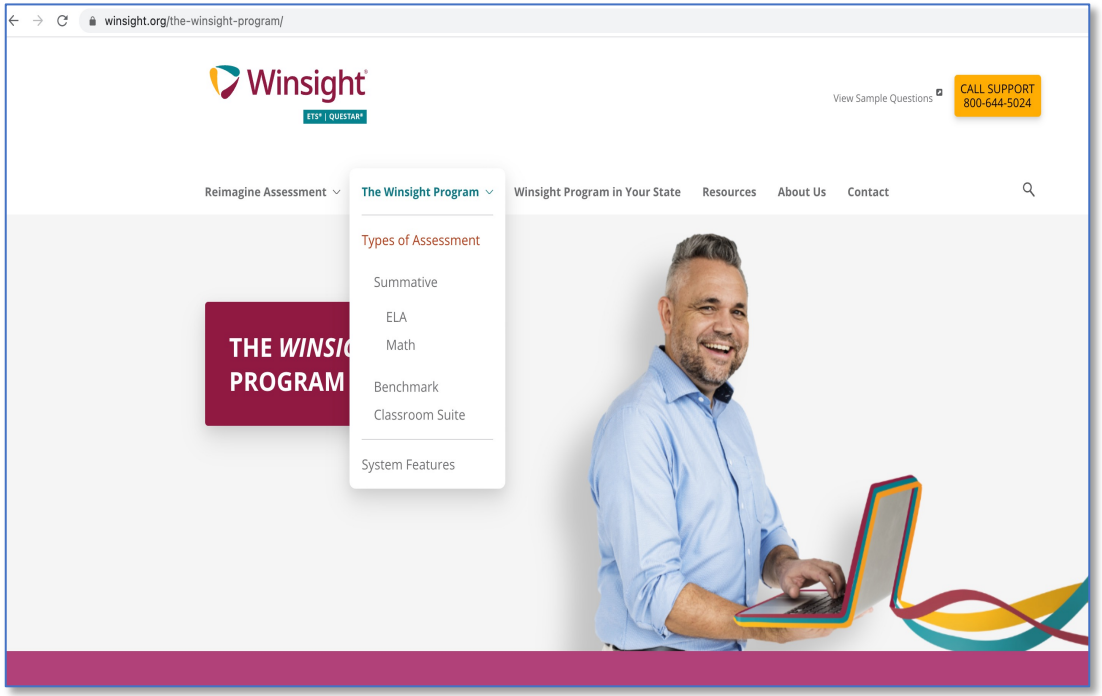
https://www.canr.msu.edu/news/what_makes_a_good_scientist

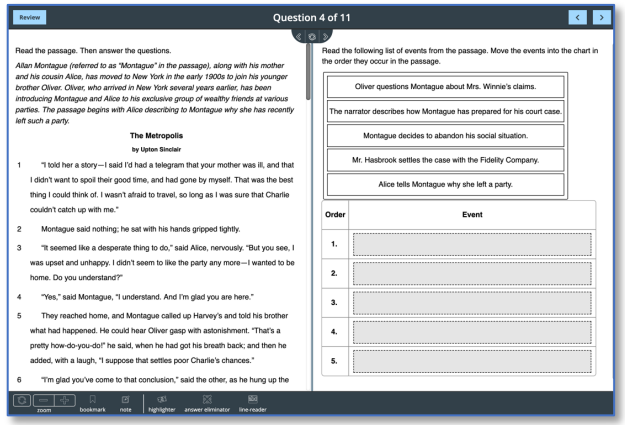
Why it matters in assessments?

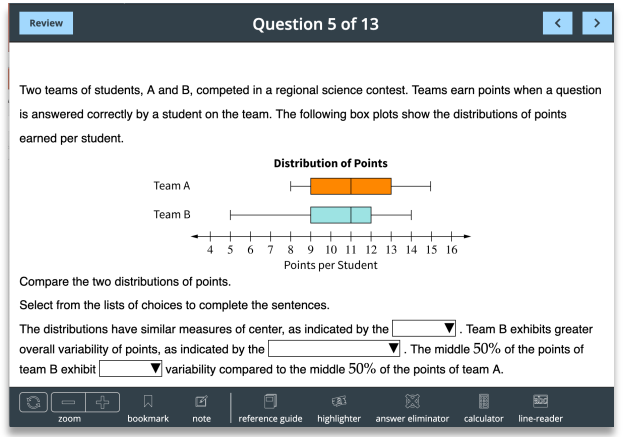
Digital Learning and Assessment

- 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





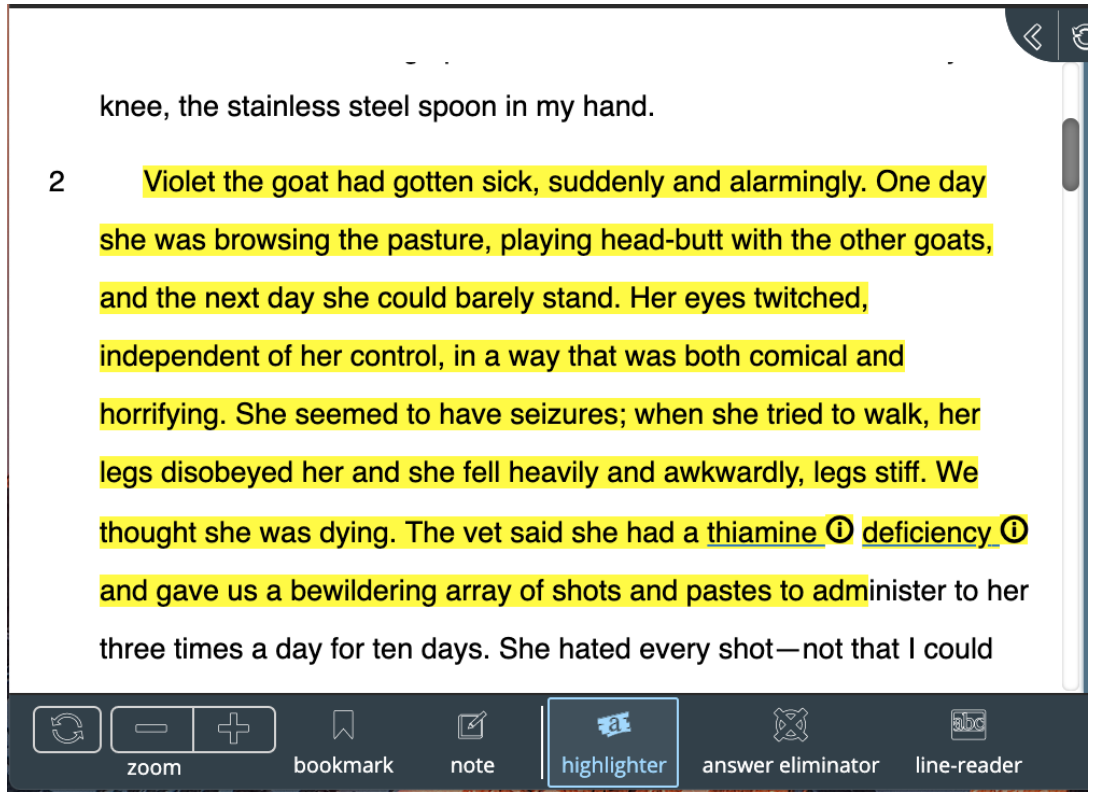


From ETS Winsight item sampler

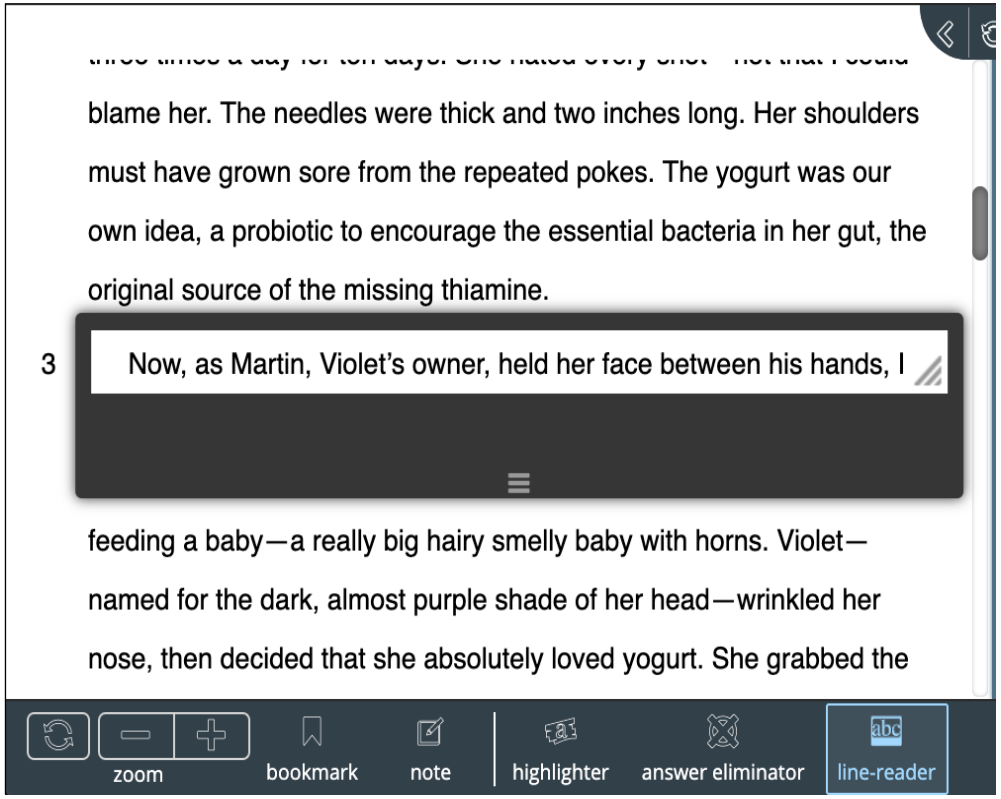
Acknowledgement: the findings presented in the following slides was conducted using field test data from the Winsight® summative assessment developed by ETS.

Tool Usage Difference

Highlighter tool

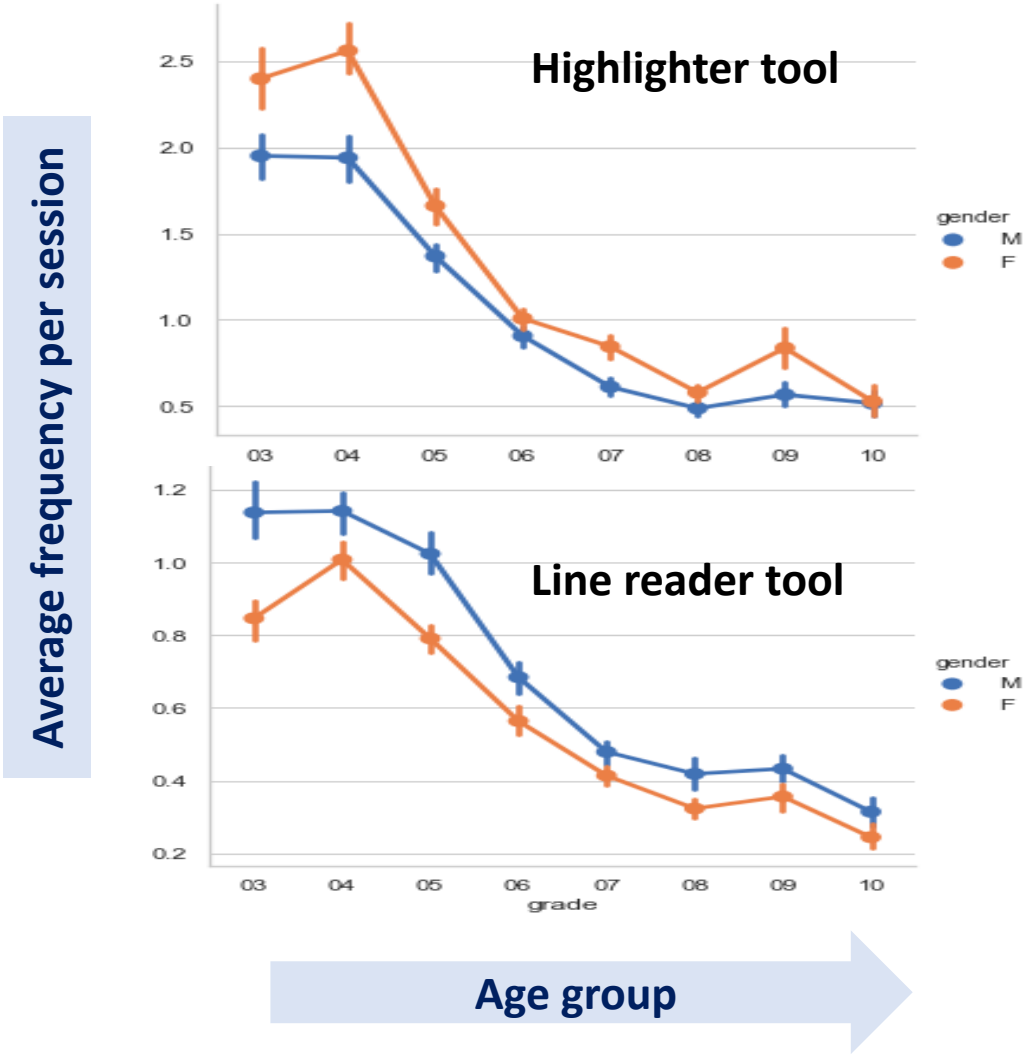


Line reader tool



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

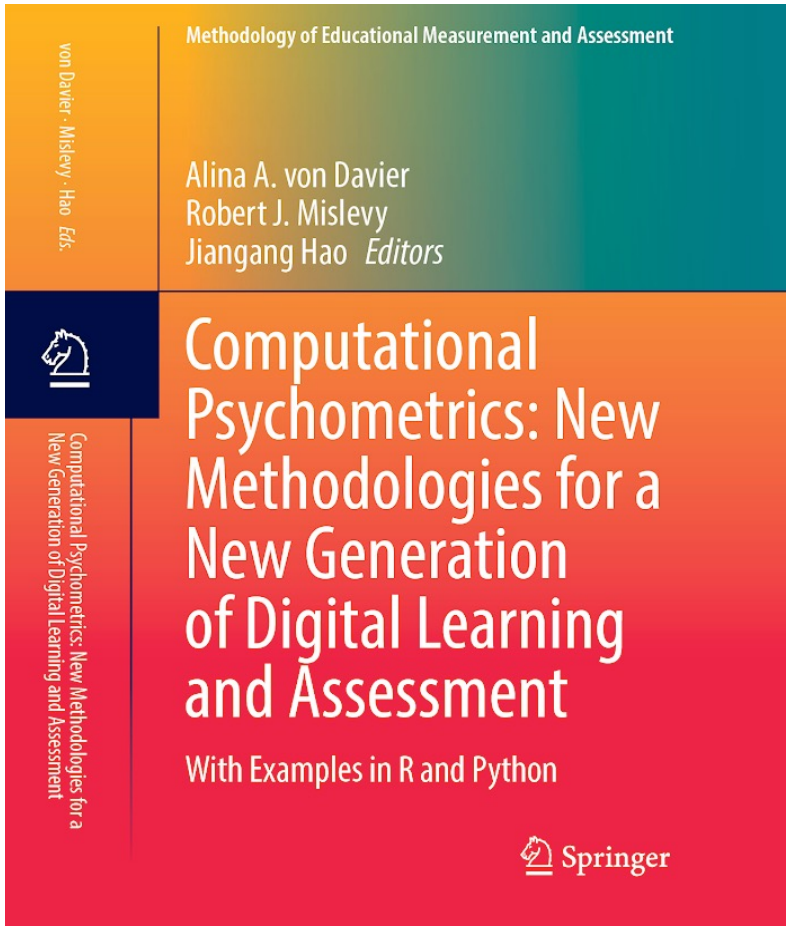
- How to make sense of the data
- Domain/application specific, and the solutions cannot be directly shopped from the “market”.

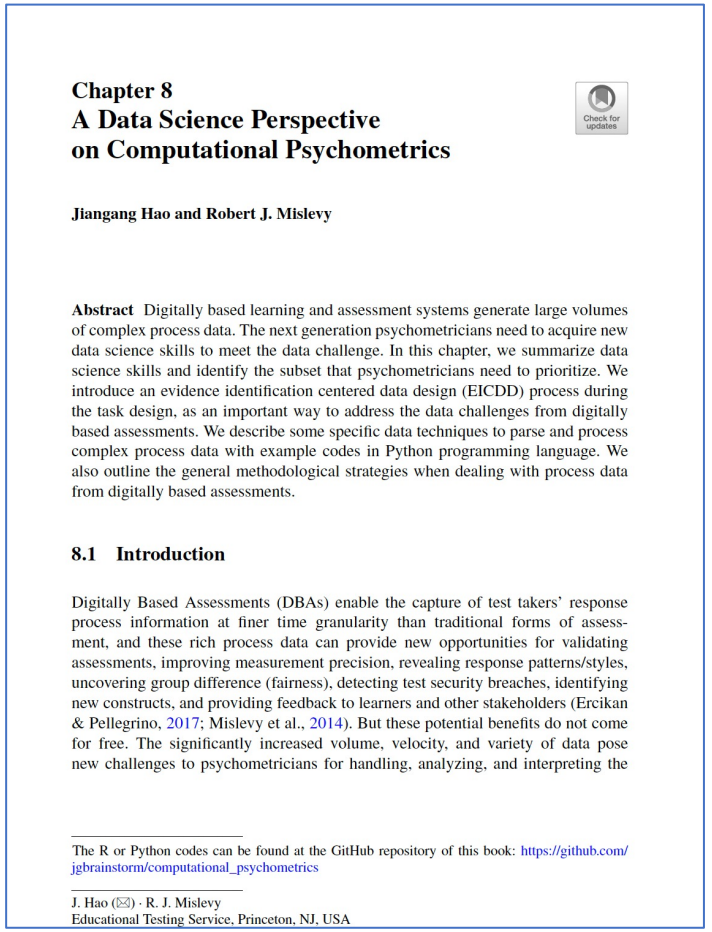
Organizational challenge

- 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

Check for updates

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 difference (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

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

https://github.com/jgbrainstorm/computational_psychometrics

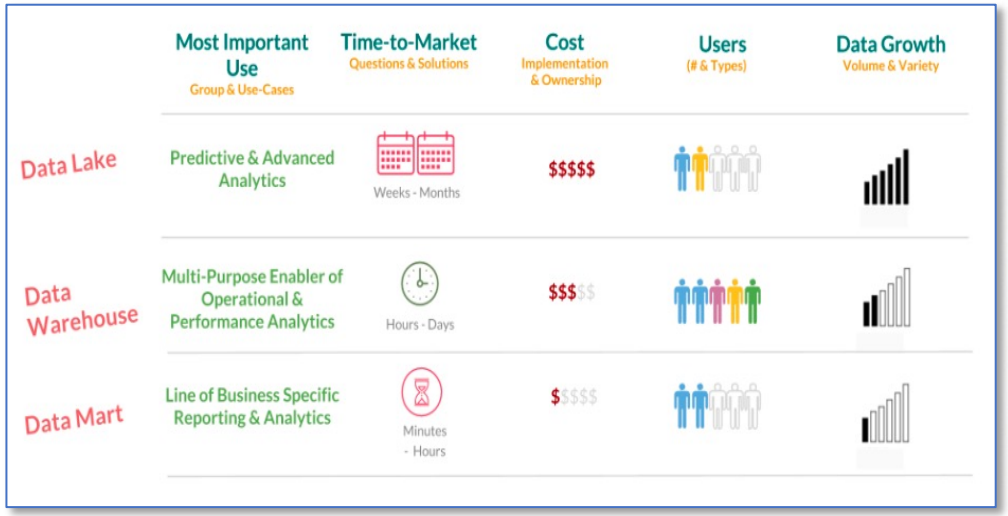
Data Type and Data Model

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.



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?

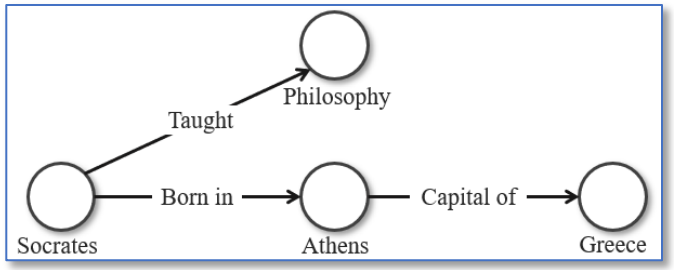
Data Model

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.

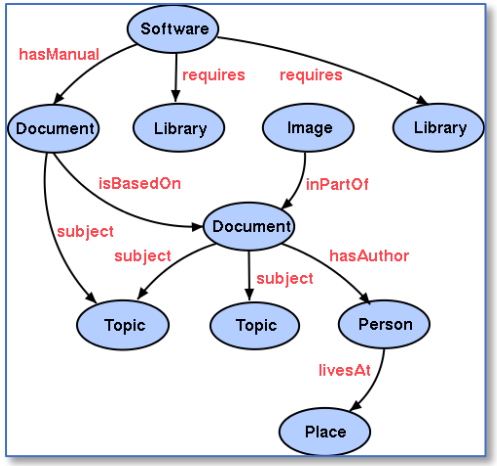
Triple Store and RDF

- **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)

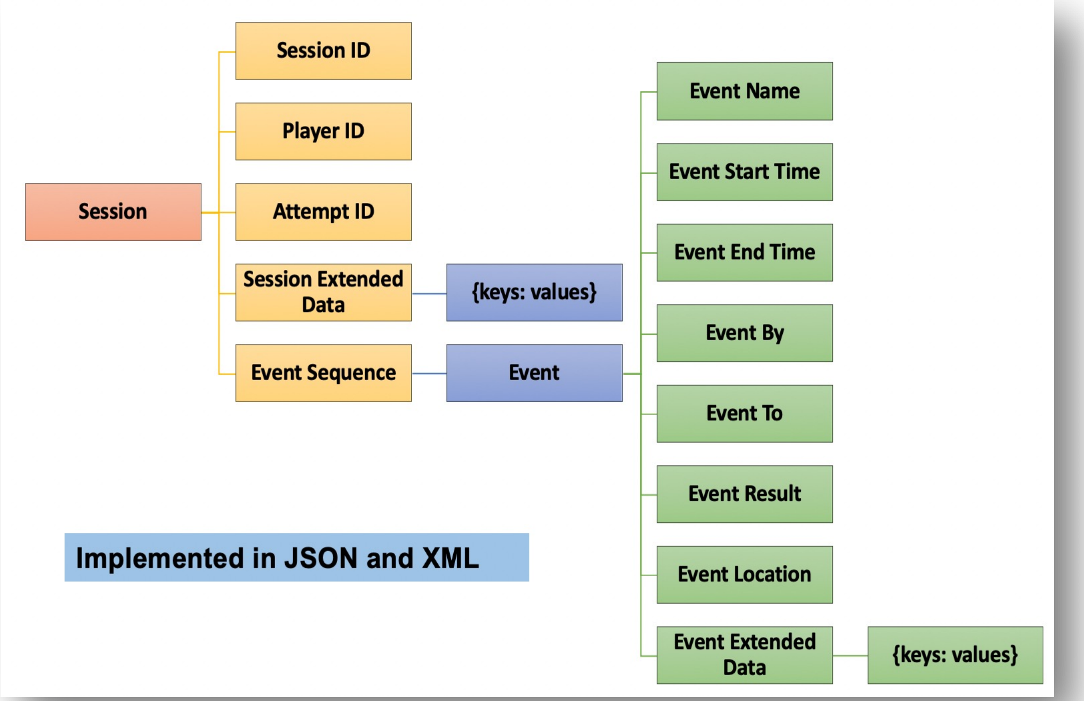


https://vodkhang.com/intelligent-system/semantic-technology-and-its-application

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


```
gameLog: [
  - {
        sessionID: "7402",
      + playerID: [2],
      - sessionExtData: {
            n_player: 2,
            team_assembly_mode: "random",
            P1: "jiangang",
            P2: "hao",
        },
      - eventSequence: [
          - {
                eventName: "chat",
                eventStartTime: "2020-05-17T00:28:48Z",
                eventEndTime: "2020-05-17T00:28:48Z",
                eventBy: "jiangang",
                eventTo: "others",
                eventResult: "hi",
                eventLocation: "slide1-step0",
                eventExtData: { },
            },
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
        ],
        teamID: "hao_jiangang",
        attemptID: 25,
    }
]
```


JSON Implementation

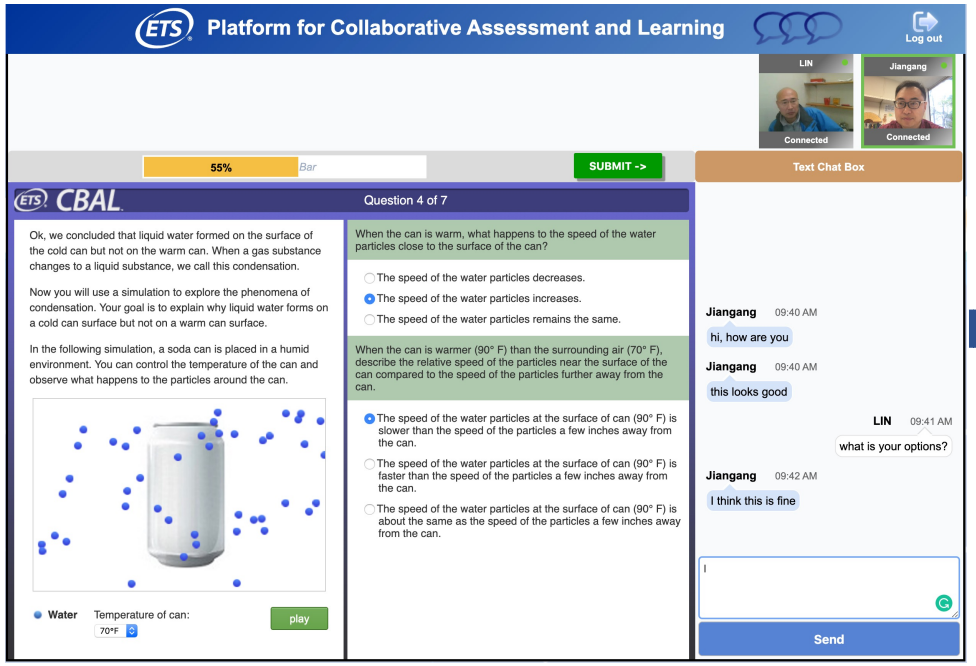

```
▼ <gameLog xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  ▼ <session>
      <sessionID> 7402 </sessionID>
      <teamID> hao_jiangang </teamID>
    ▶ <playerID>
      <attemptID> 25 </attemptID>
      <sessionExtData>
      ▶ <pair>
      ▶ <pair>
      ▶ <pair>
      ▶ <pair>
      ▶ <pair>
      </sessionExtData>
    ▼ <eventSequence>
        <event>
          <eventName> chat </eventName>
          <eventStartTime> 2020-05-17T00:28:48Z </eventStartTime>
          <eventEndTime> 2020-05-17T00:28:48Z </eventEndTime>
          <eventBy> jiangang </eventBy>
          <eventTo> others </eventTo>
          <eventResult> hi </eventResult>
          <eventLocation> slide1-step0 </eventLocation>
        </event>
      ▶ <event>
      ▶ <event>
      ▶ <event>
      ▶ <event>
      ▶ <event>
      ▶ <event>
      ▶ <event>
      ▶ <event>
      ▶ <event>
      ▶ <event>
      ▶ <event>
      ▶ <event>
      ▶ <event>
      ▶ <event>
      ▶ <event>
      ▶ <event>
      </eventSequence>
    </session>
</gameLog>
```

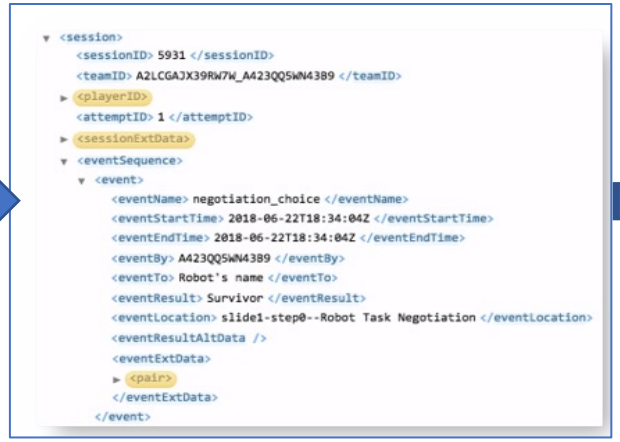

XML Implementation

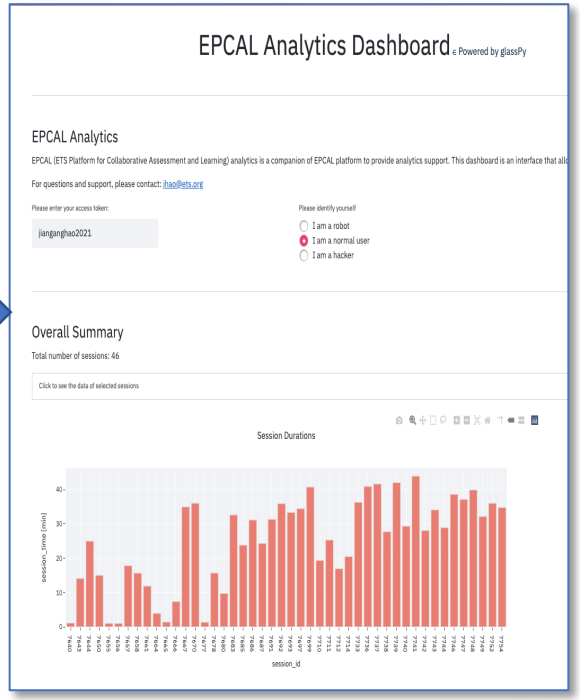
Hao et al., 2014; 2016; Hao & Mislevy, 2018

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

Hao & Mislevy, 2021

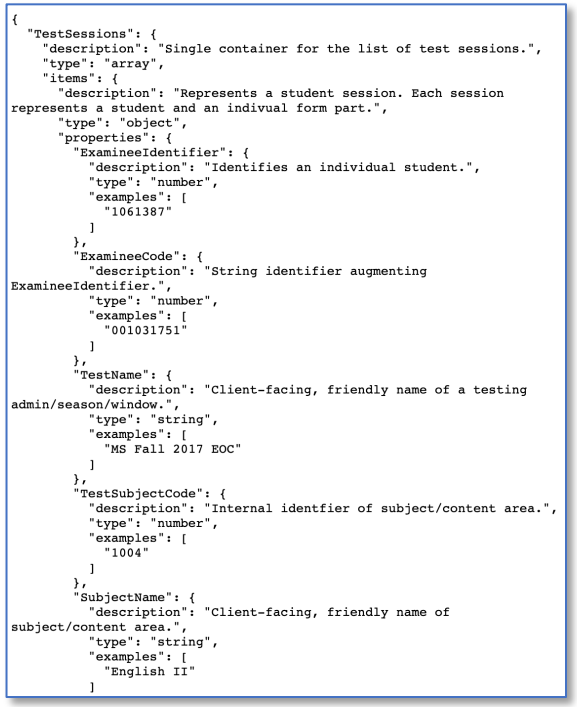
Types of Data Processing

- Batch processing
  - Process data after the data are collected
  - E.g., Apache Hadoop, …
- Stream processing
  - Process live data
  - E.g., Apache Kfaka, 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 indivual 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 identfier of subject/content area.",
          "type": "number",
          "examples": [
            "1004"
          ]
        },
        "SubjectName": {
          "description": "Client-facing, friendly name of
subject/content area.",
          "type": "string",
          "examples": [
            "English II"
          ]
```

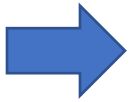


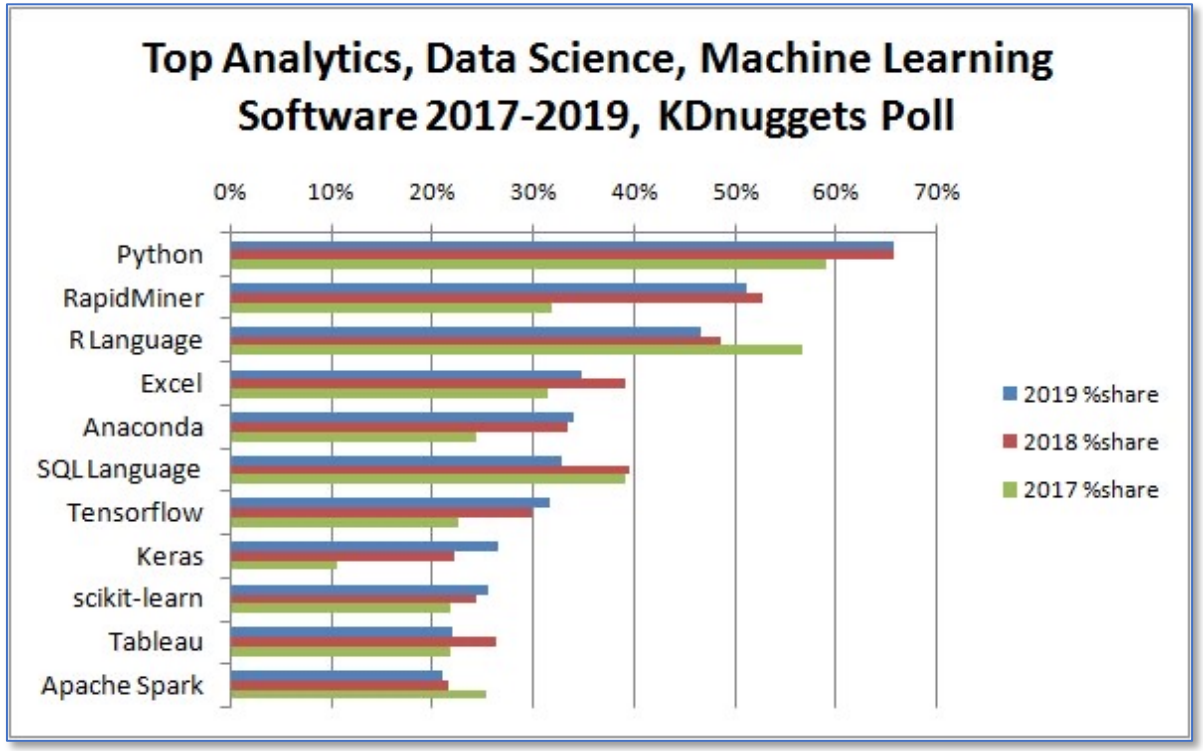
Table 3. *A schematic of the data frame converted from sequential data.*

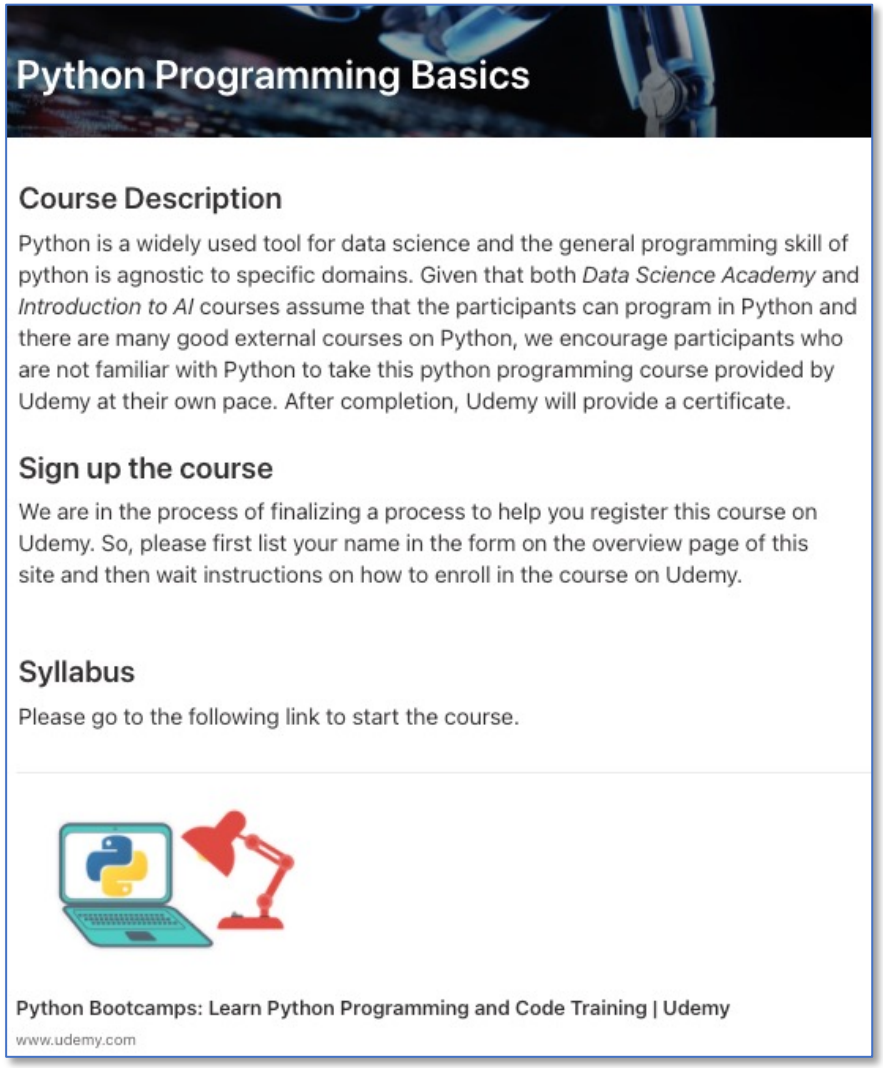
Sequence Number	*Event Time*	*Test Taker ID*	*Session ID*	*Event Name*	*Attribute 1*	*Attribute 2*	...
1	Timestamp 1	A	1	...	...	...	...
2	Timestamp 2	A	1	...	...	...	
3	…	A	1	...	...	...	...
4	…	A	1	...	...		
5	…	B	2	...	...	...	...
6	…	B	2	...	...	...	...
7	…	B	2	...	...	...	...

Hao & Mislevy, 2021

Data Science Tools

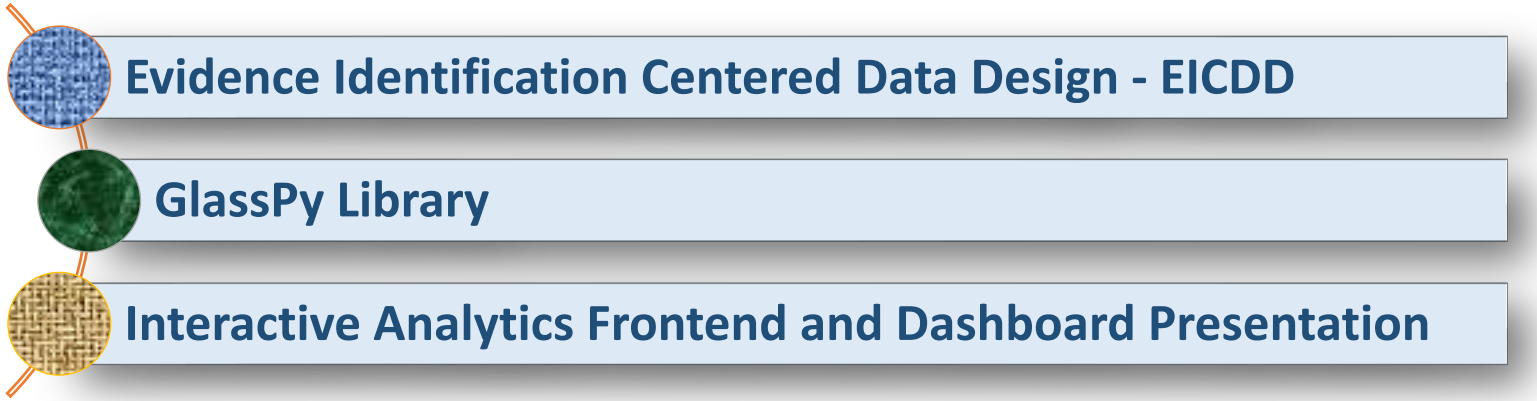
Survey from KDnuggets





GlassPy Analytics Solution at ETS

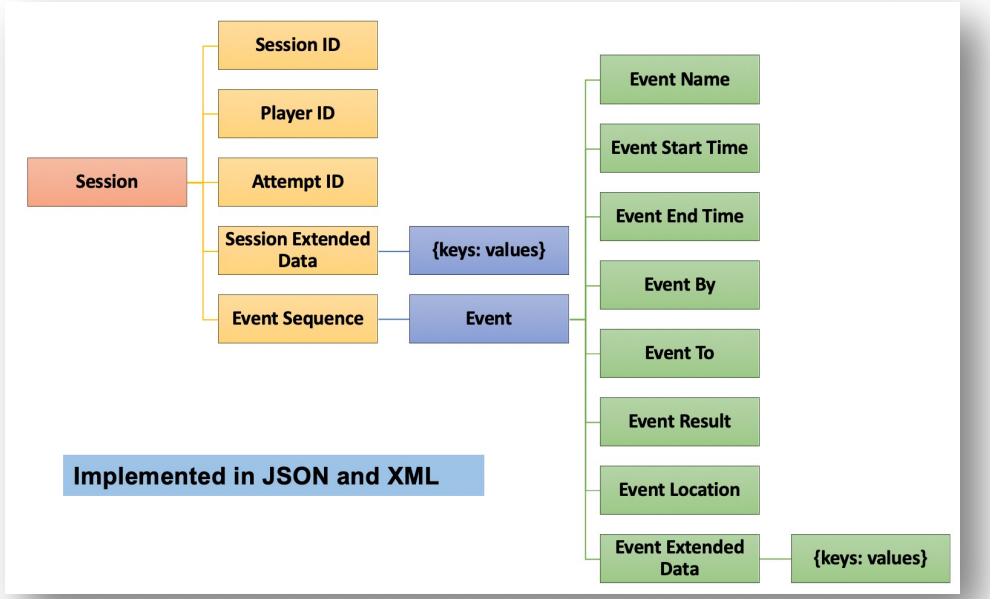
- GlassPy: Game Log Analysis in Python, started in 2014 at ETS
- We envision glassPy more as a generic framework for handling process data from digital-based assessments (DBAs)
- It is an "adaptor" to connect the Evidence Centered Design (ECD) to data in practice
- GlassPy analytics solution Includes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



Hao & Mislevy, 2021

Evidence Identification Centered Data Design

Data Model



Standard Operational Procedure

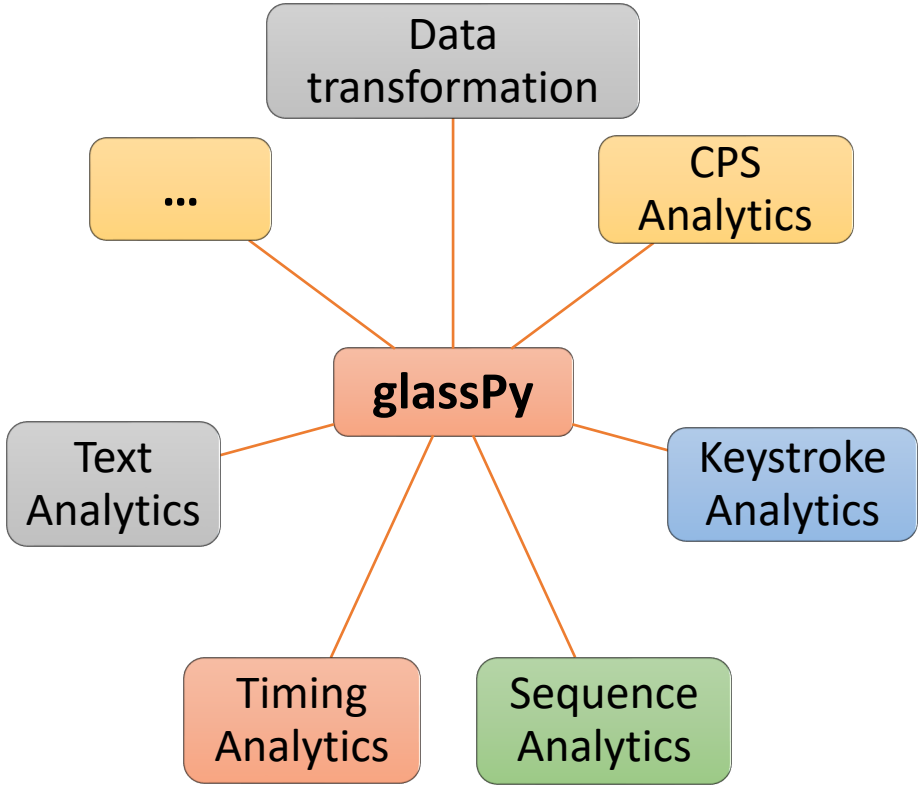
X: participant XX: coordinator

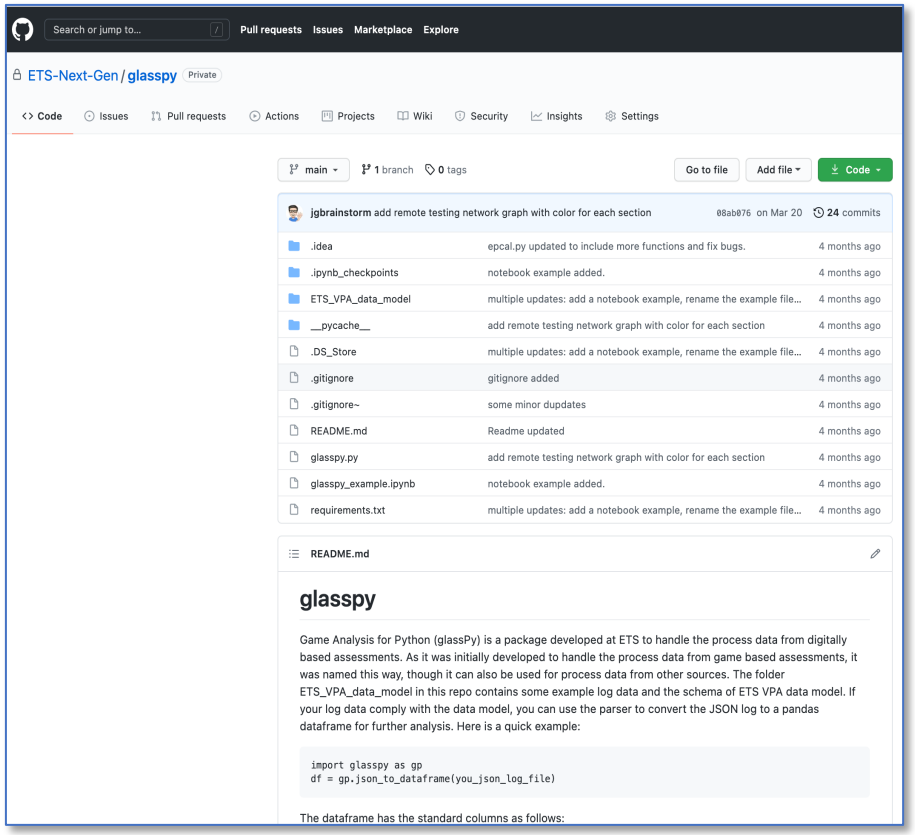
		Roles				
		Learning Scientist/ Assessment Developer	Psychometrician	Programmer	Data Scientist	Data analyst
Timeline	**Step 1**: All-party meeting to brief the procedures	X	X	X	XX	X
	Step 2: Specification of "Q matrix" for constructs and evidences	X				
	Step 3: Specification of additional evidence for 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

Hao et al., 2016; Hao & Mislevy, 2018; 2021

GlassPy Library

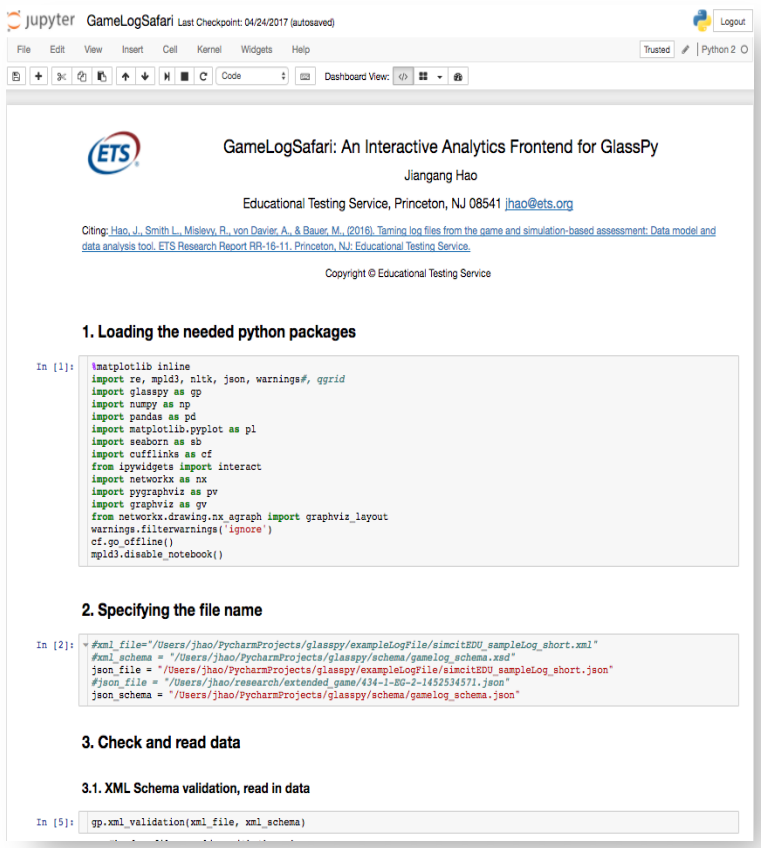


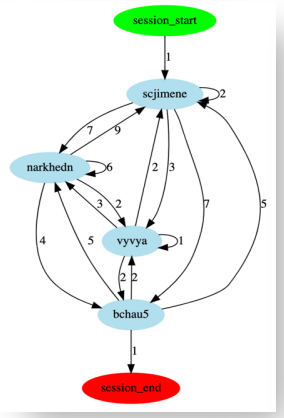


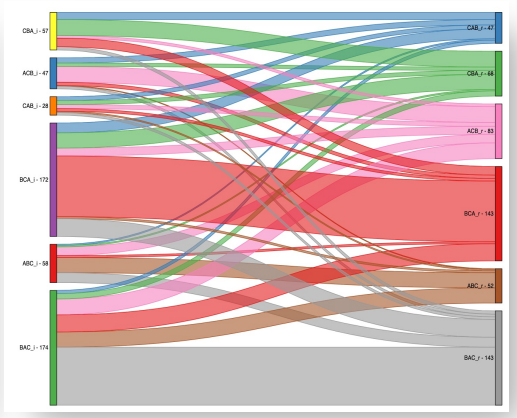
Not open sourced yet and under active expanding

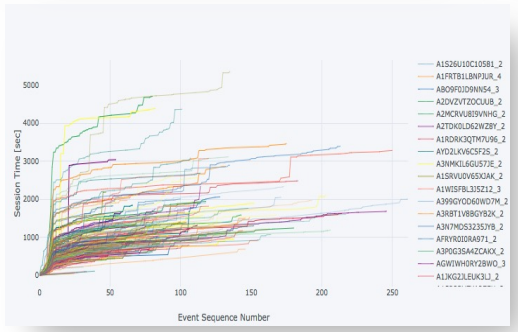
Interactive Analytics Frontend

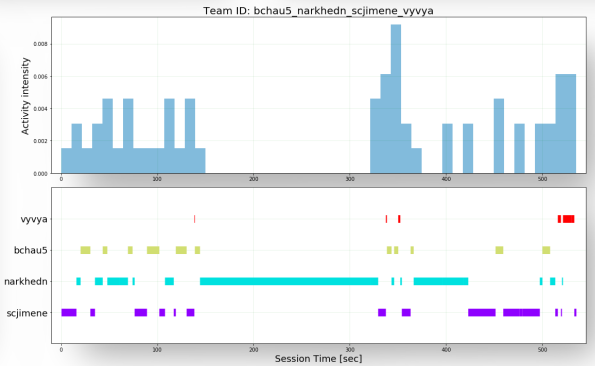
Jupyter/Zeppelin Notebook/Lab + Plotly Dash/Shiny /Streamlit+ FastAPI/Flask+ others







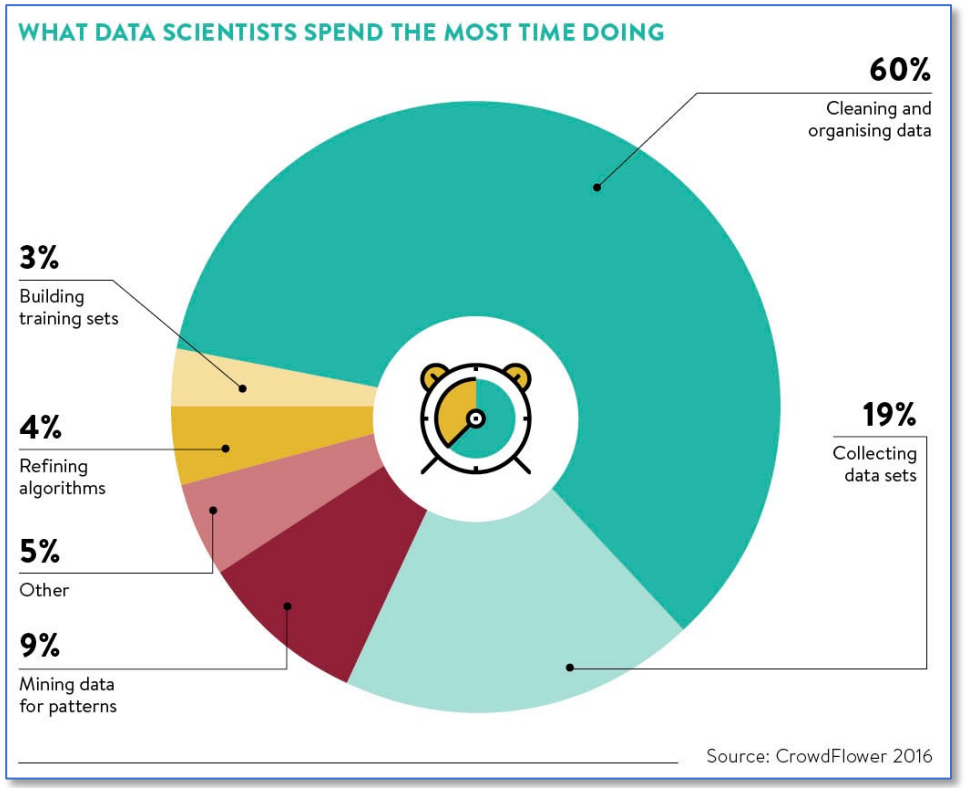




Summary

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!



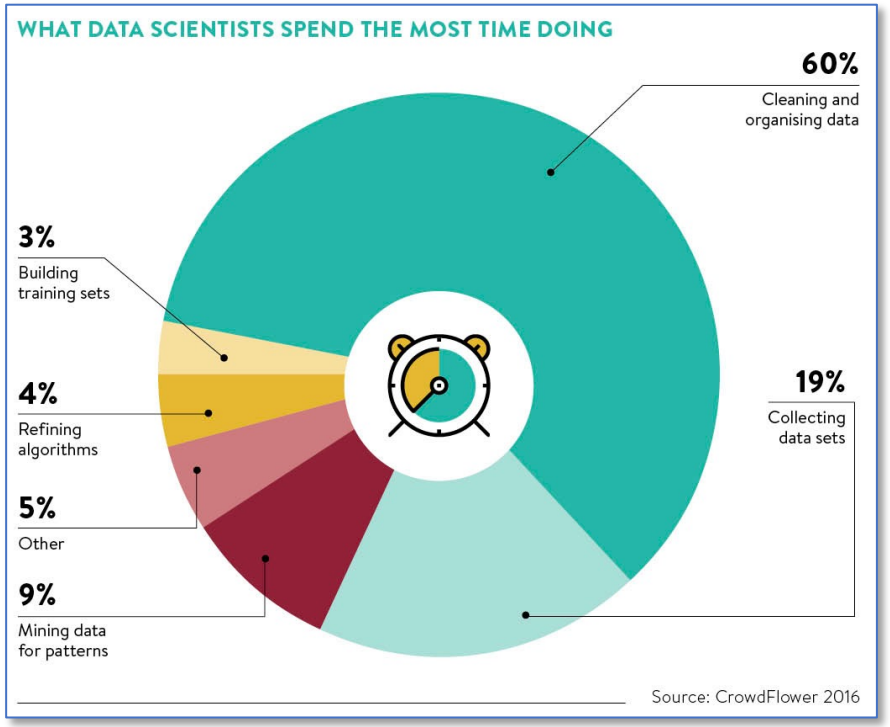
Data Wrangling

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



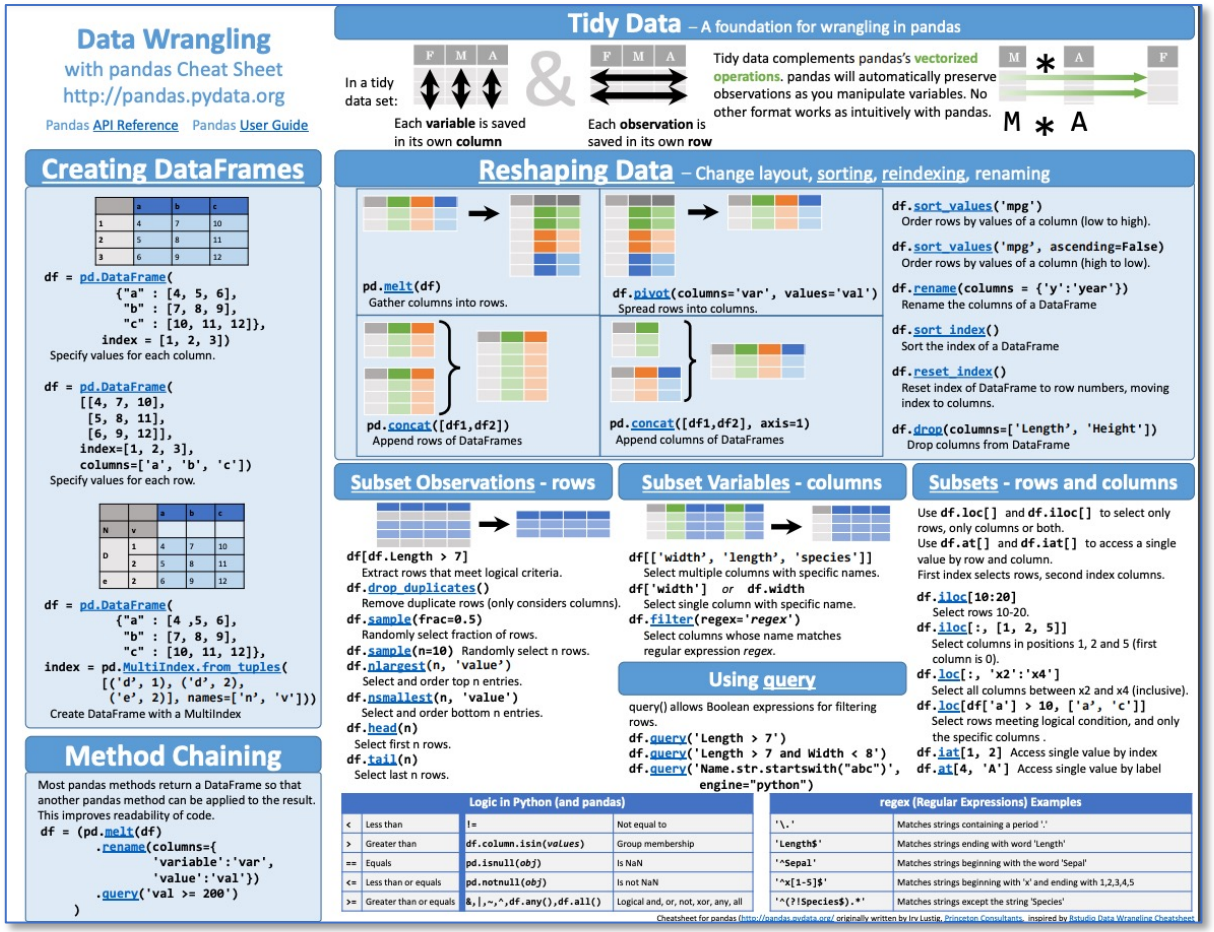
Types of Raw Data

- **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

pandas

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



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.
```

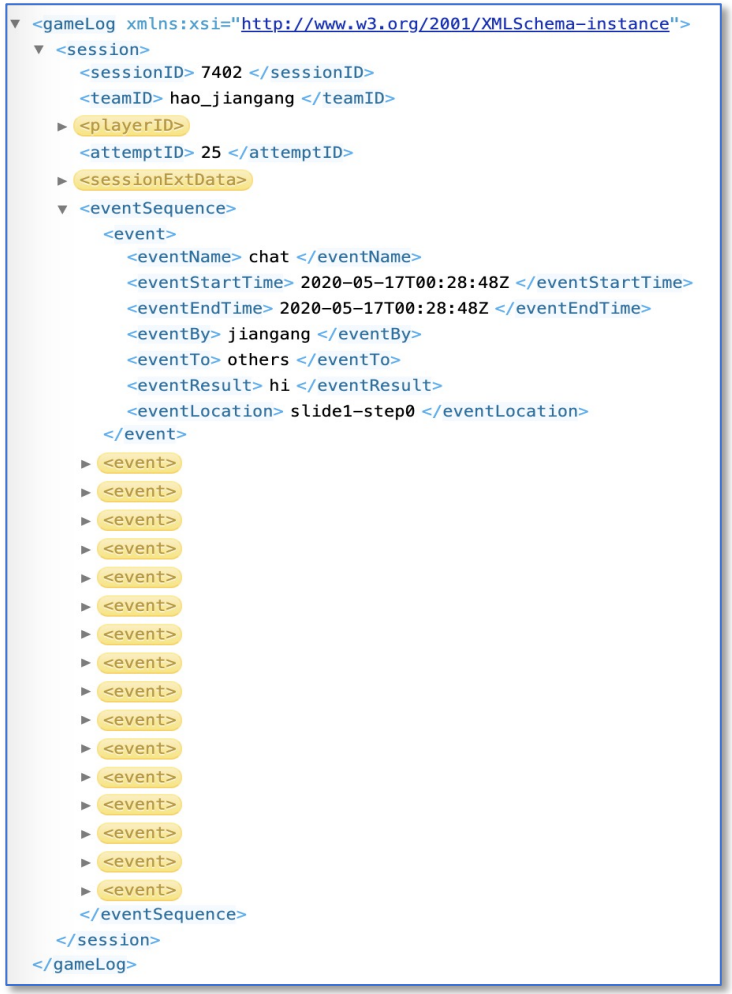
Demo of parsing unstructured file using Python

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"?>

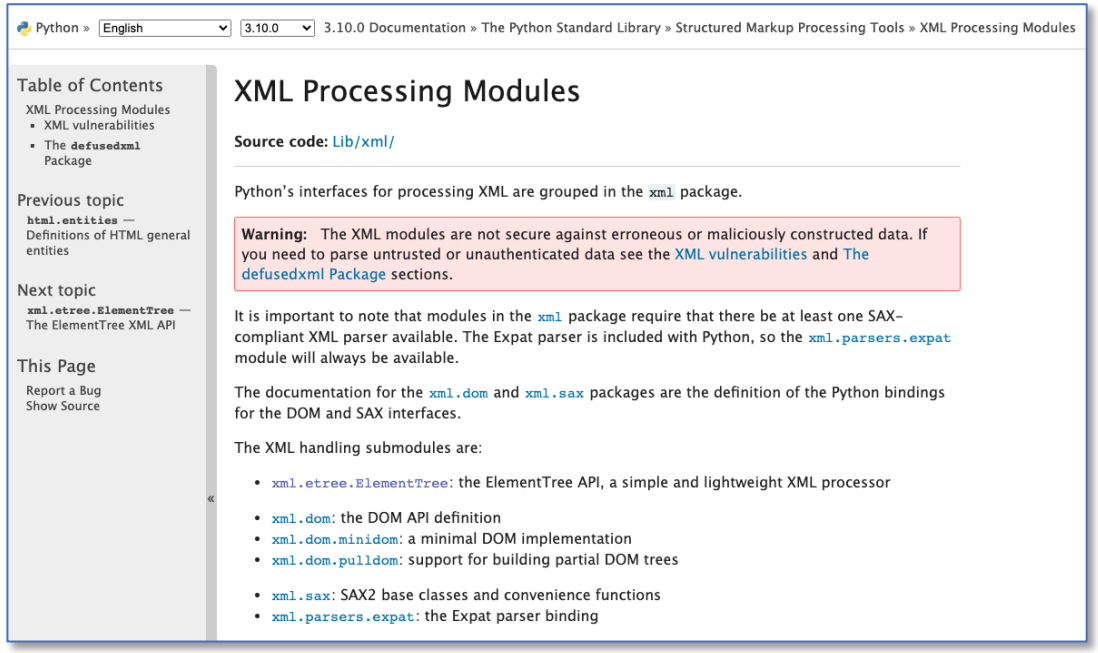


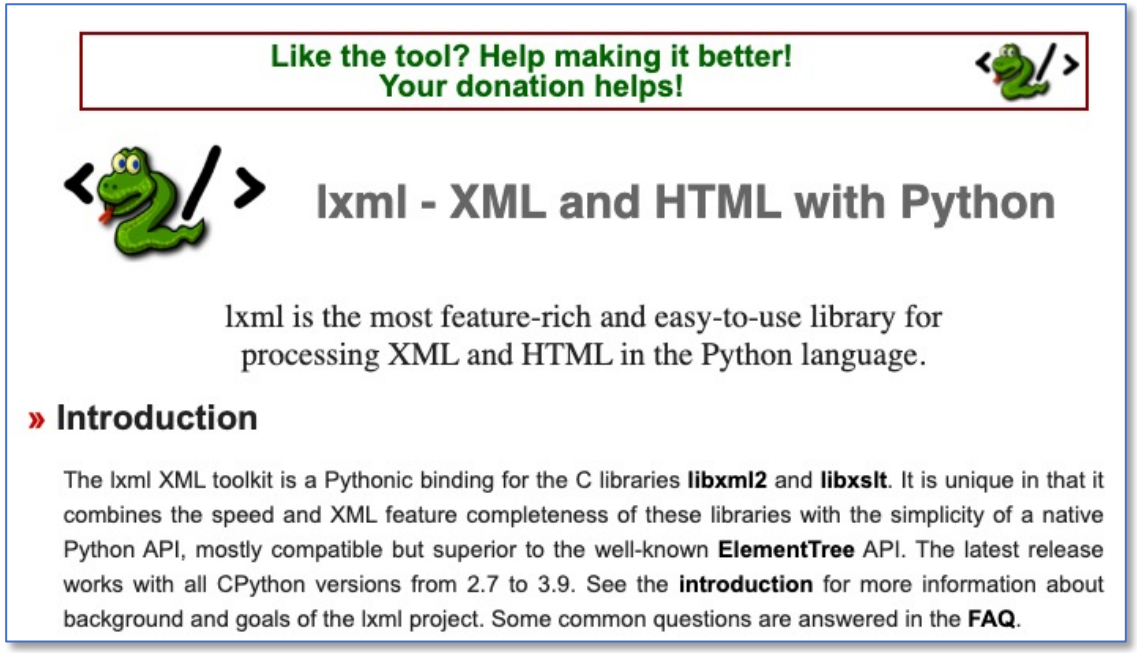
Data from EPCAL

https://www.w3schools.com/xml/

Python Tool for XML

- "Official" XML library: https://docs.python.org/3/library/xml.html
- LXML: https://lxml.de/





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), ECMA-404, ISO/IEC 21778:2017
Open format?	Yes
Website	json.org

Introduced in 2000


```
{
 - gameLog: [
    - {
         sessionID: "7402",
       + playerID: [2],
       + sessionExtData: {4},
       - eventSequence: [
          - {
               eventName: "chat",
               eventStartTime: "2020-05-17T00:28:48Z",
               eventEndTime: "2020-05-17T00:28:48Z",
               eventBy: "jiangang",
               eventTo: "others",
               eventResult: "hi",
               eventLocation: "slide1-step0",
               eventExtData: { },
            },
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
          + {8},
         ],
         teamID: "hao_jiangang",
         attemptID: 25,
      }
   ]
}
```


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/

ETS®

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
-->

  <!-- Root element and children -->

  <xs:element name="gameLog">
    <xs:complexType>
      <xs:sequence>
              <xs:element name="session" minOccurs="1" maxOccurs="unbounded">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="sessionID" type="idType" minOccurs="1" maxOccurs="1"/>
                    <xs:element name="teamID" type="idType" minOccurs="1" maxOccurs="1"/>
                    <xs:element name="playerID" type="dictType" minOccurs="1" maxOccurs="1"/>
                    <xs:element name="attemptID" type="idType" minOccurs="1" maxOccurs="1"/>
                    <xs:element name="sessionExtData" type="dictType" minOccurs="0" maxOccurs="1"/>
                    <xs:element name="eventSequence" minOccurs="1" maxOccurs="1">
                      <xs:complexType>
                        <xs:sequence>
                          <xs:element name="event" type="eventType" minOccurs="1" maxOccurs="unbounded"/>
                        </xs:sequence>
                      </xs:complexType>
                    </xs:element>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>


  <!-- Data type definitions -->


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


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

Example schema from ETS VPA data model

JSON Schema

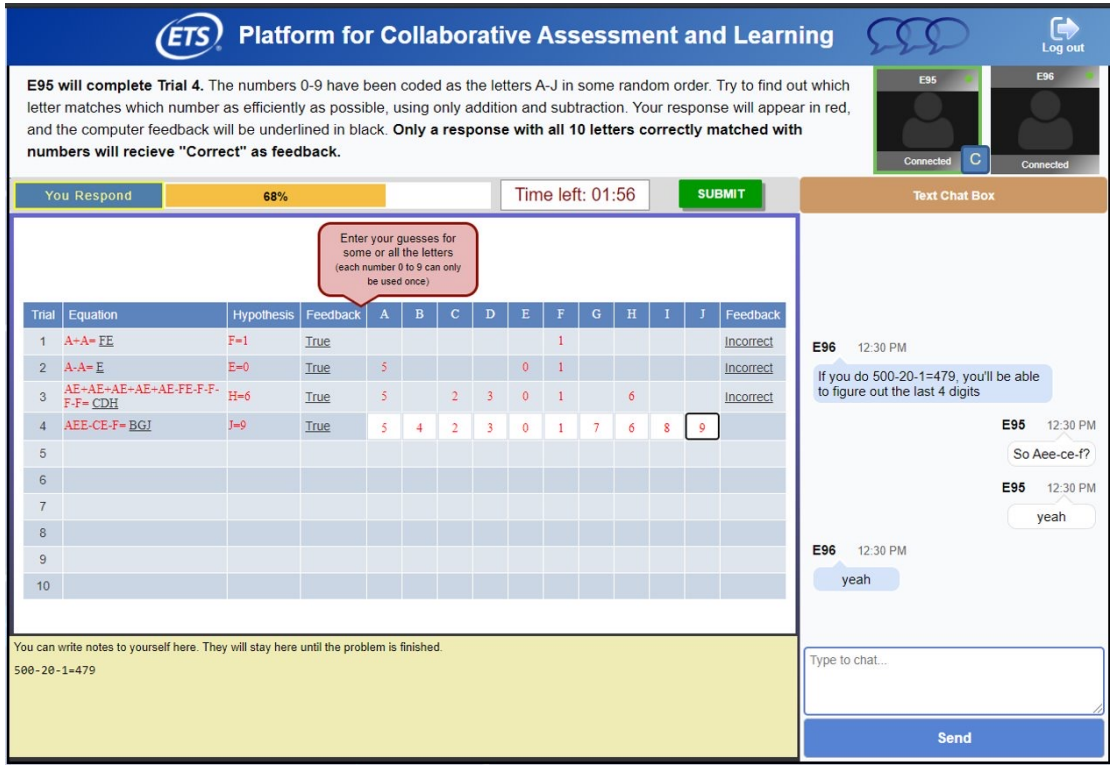
- Define JSON file/string
- Can be used to validate JSON file

```
{
  "$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




```
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

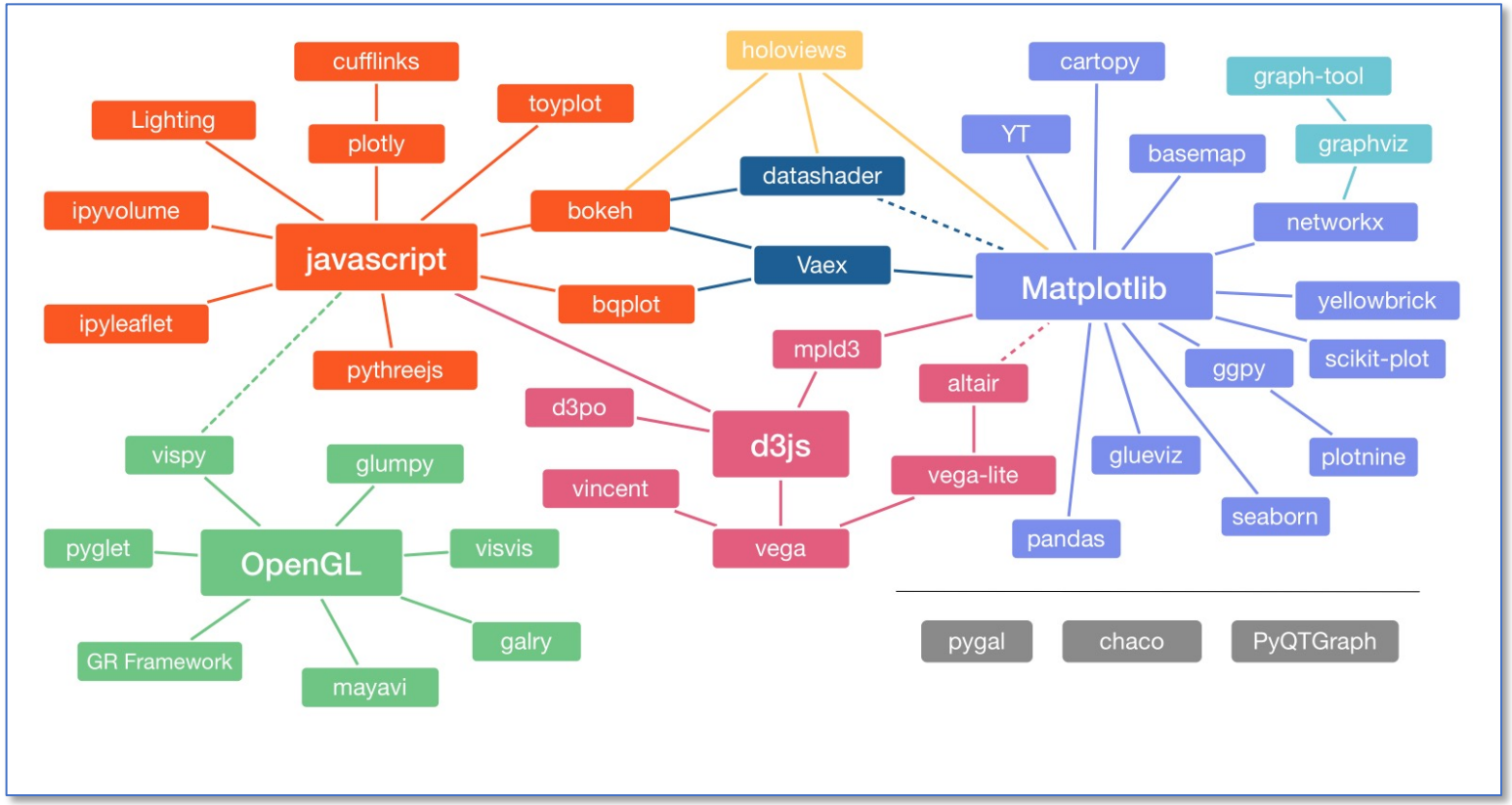
Steps to Visualization

- 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/

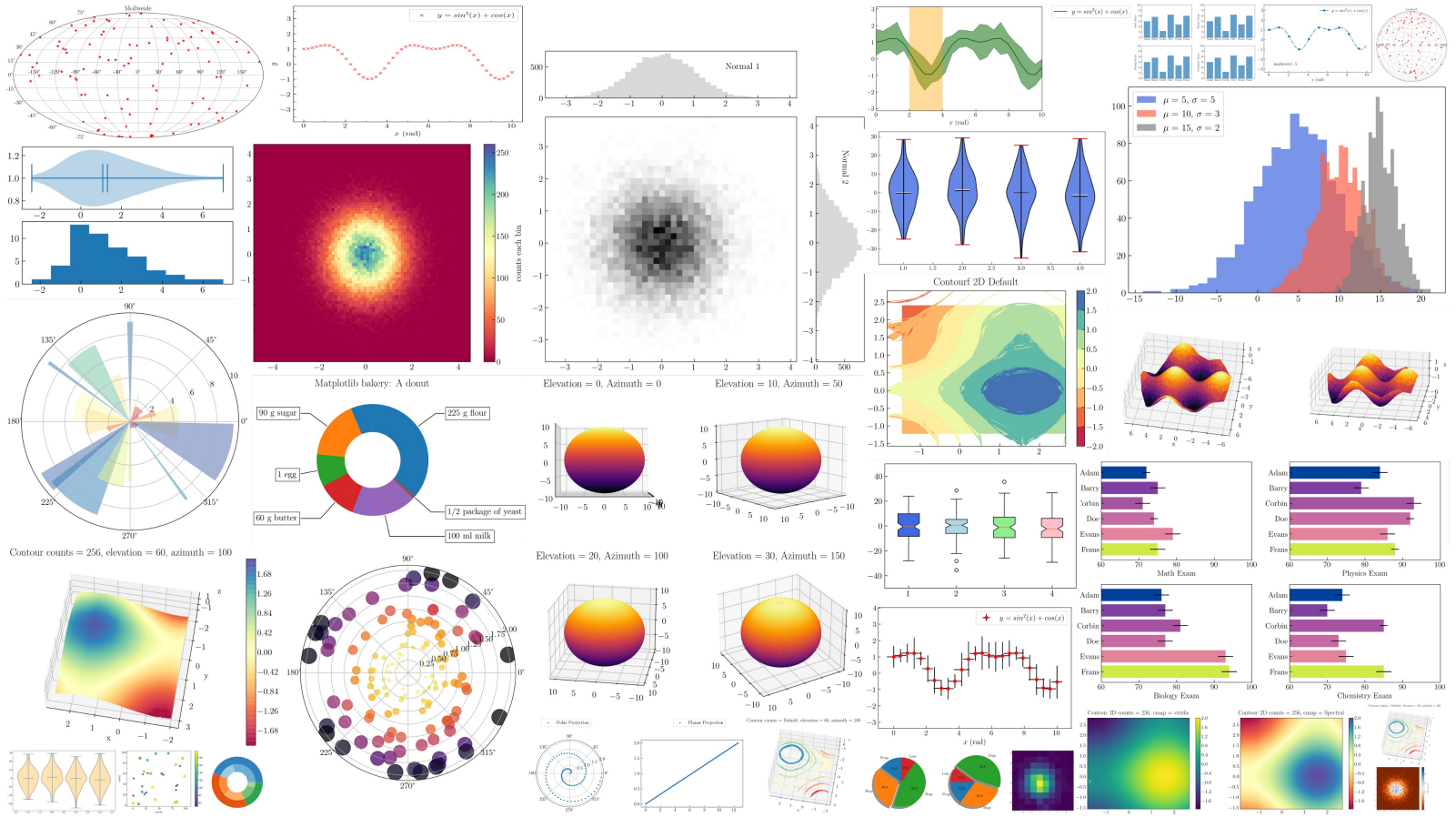
Visualization in Python

- PyViz is your gateway: www.pyviz.org
- Visualization paradigms



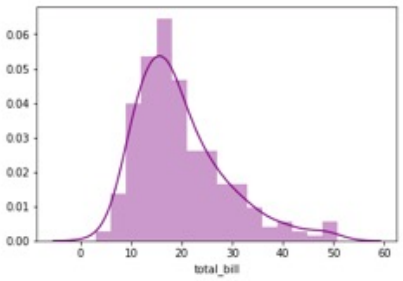
Matplotlib

https://matplotlib.org/

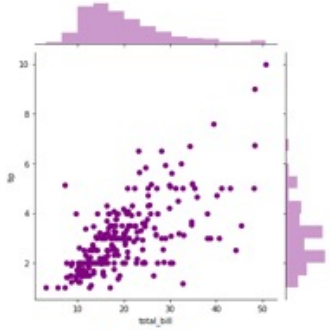


Seaborn: built on top of matplotlib

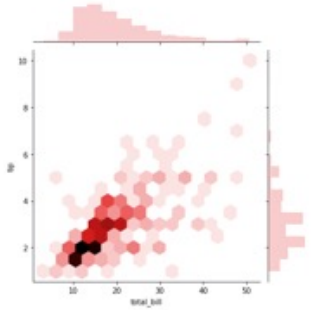
https://seaborn.pydata.org/



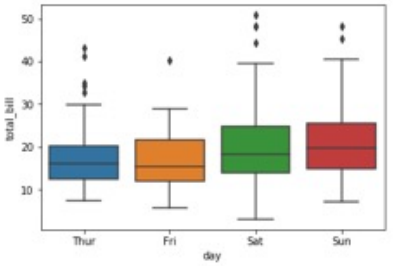
distplot



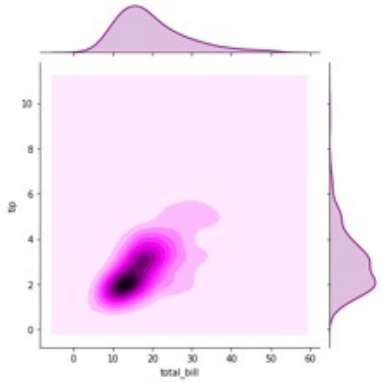
Jointplot



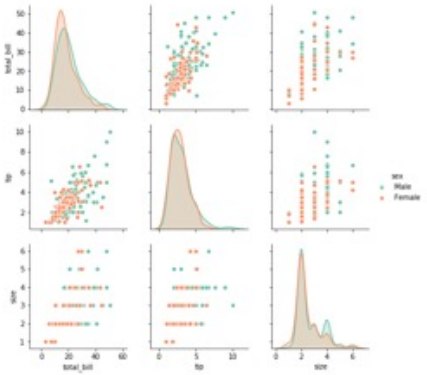
Hexplots



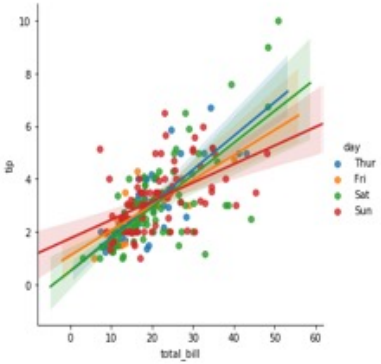
Boxplots



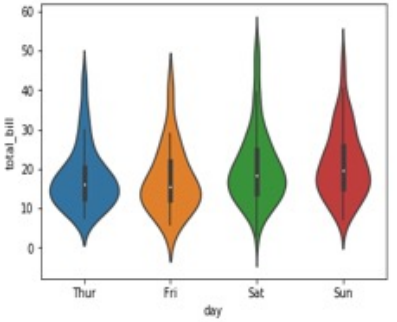
KDE Plot



Pair Plots



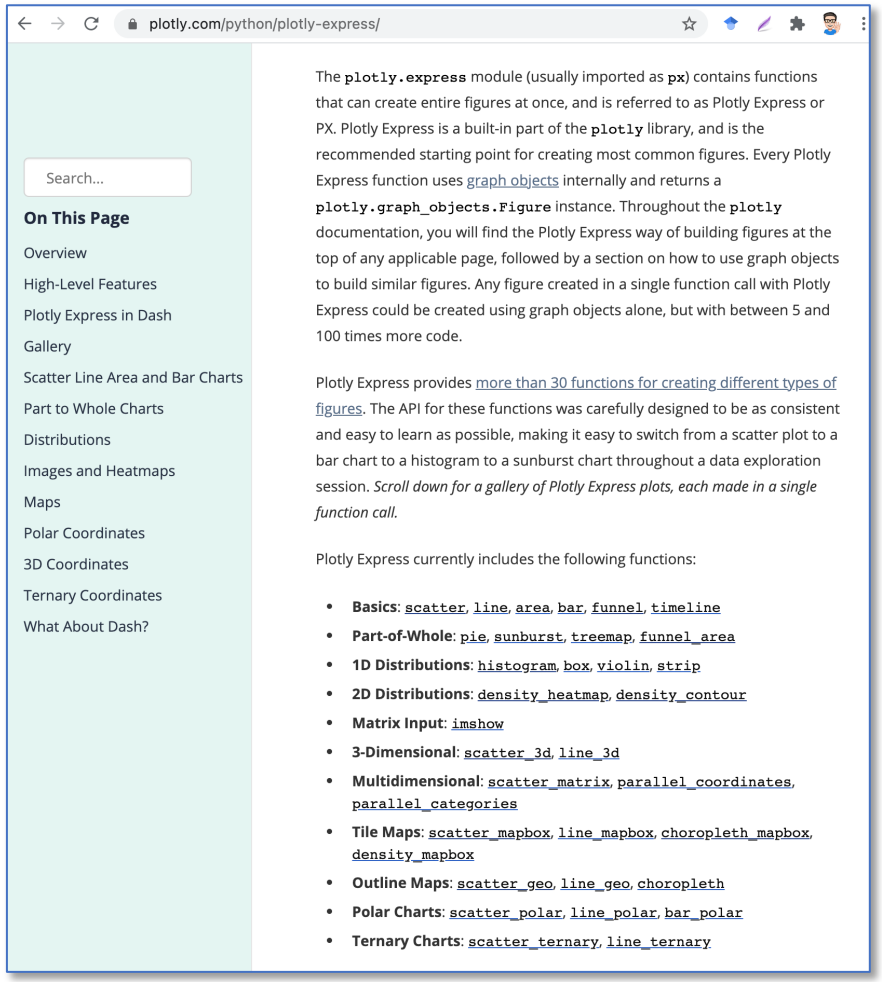
LM Plots

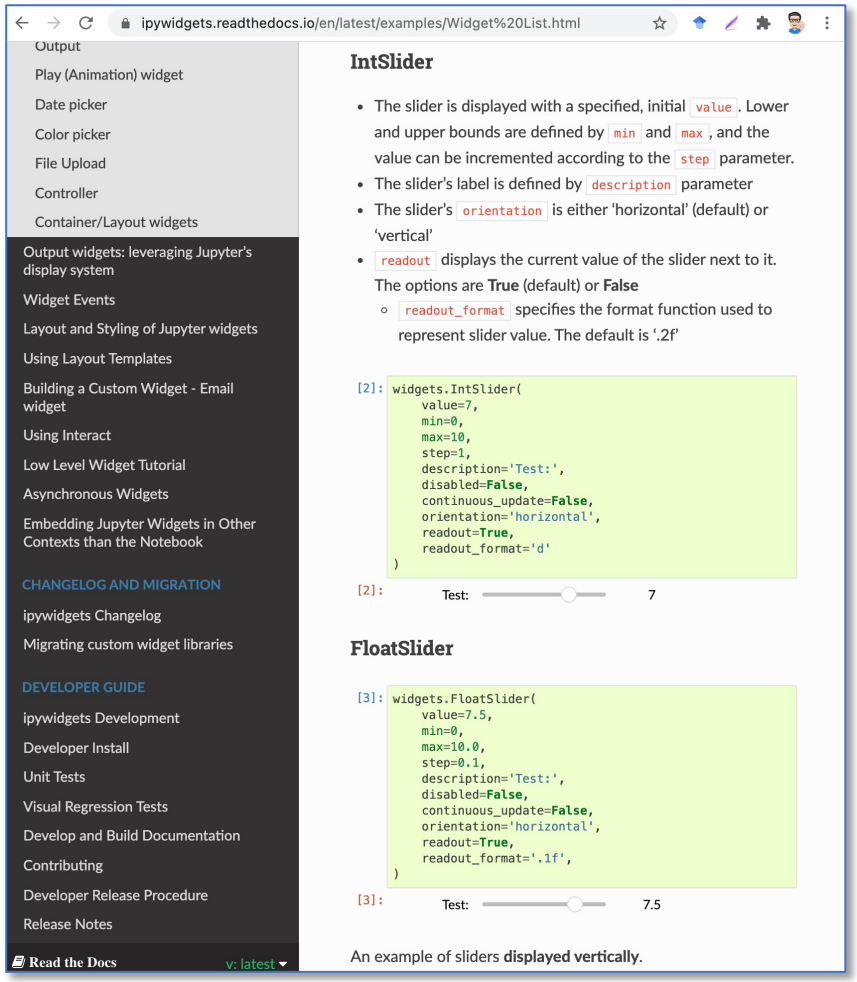


Violin Plots

Interactive Visualization

Plotly Express and ipywidgets

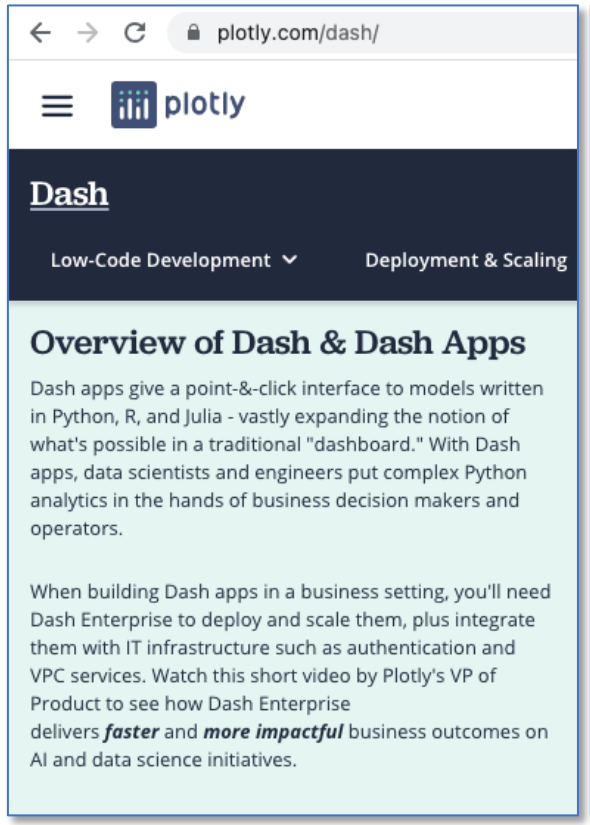


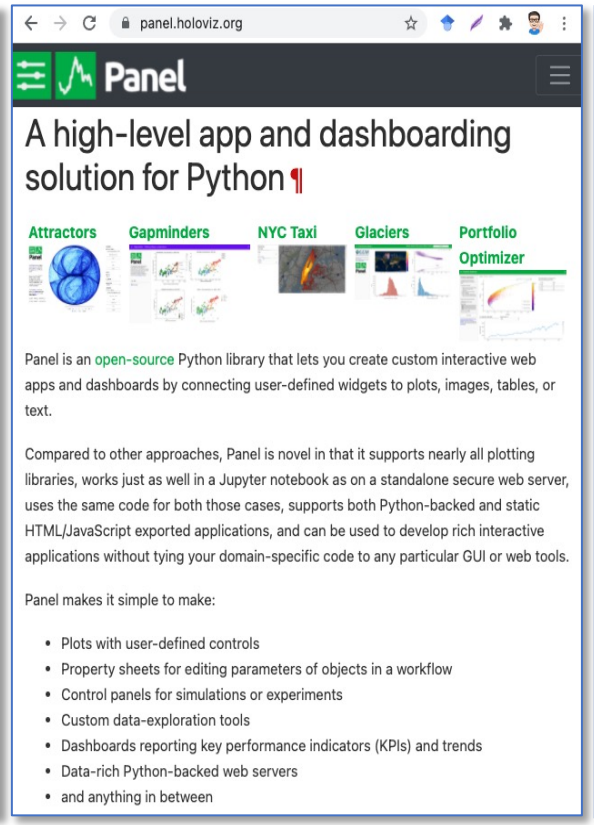


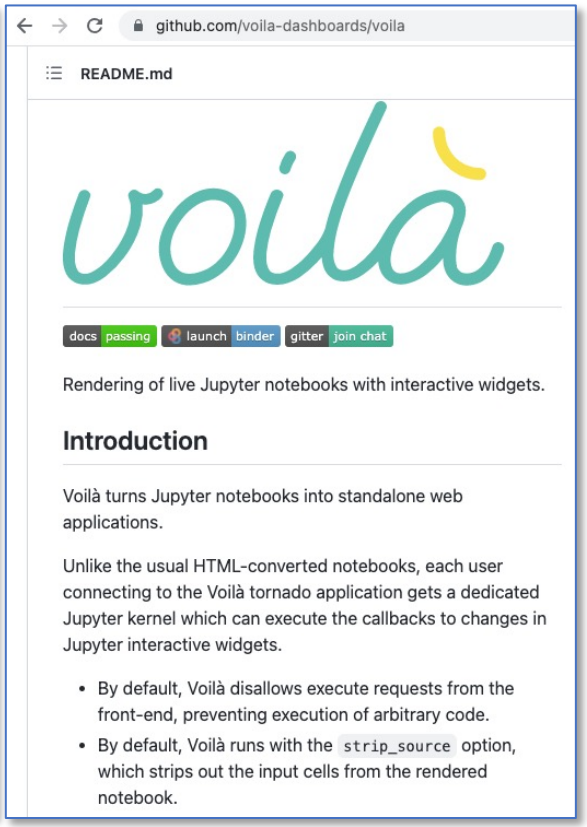
Demos

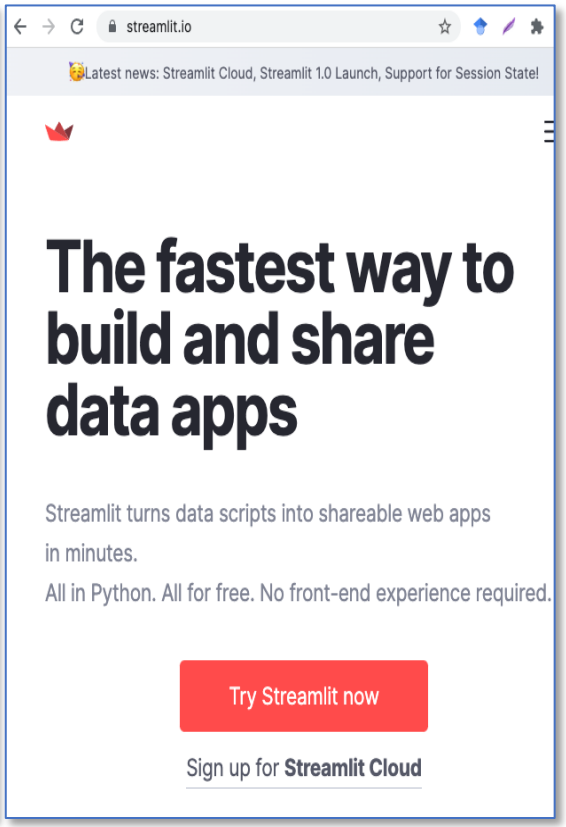
Dashboarding

Major Dashboard Tools in Python



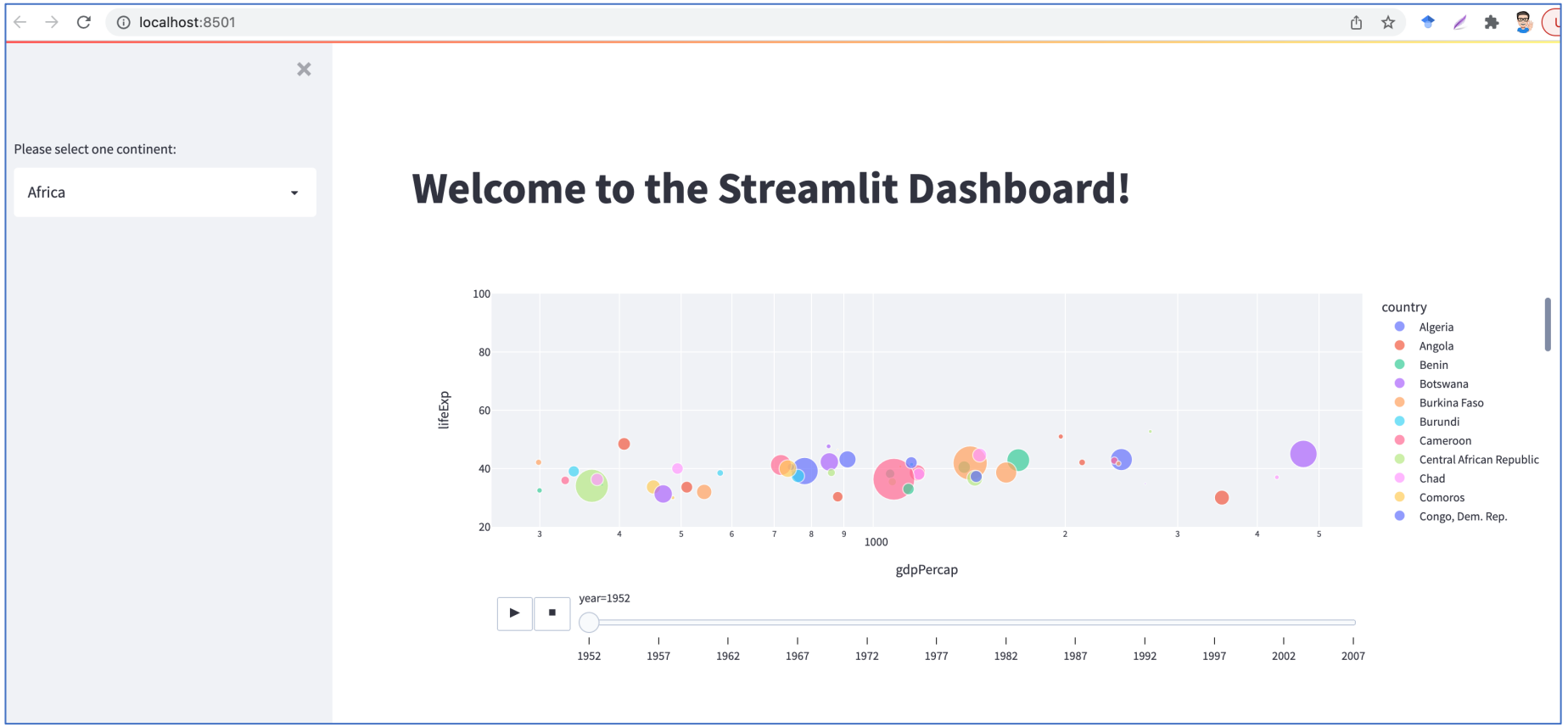






They are kind of equivalent to the Shiny package in R

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