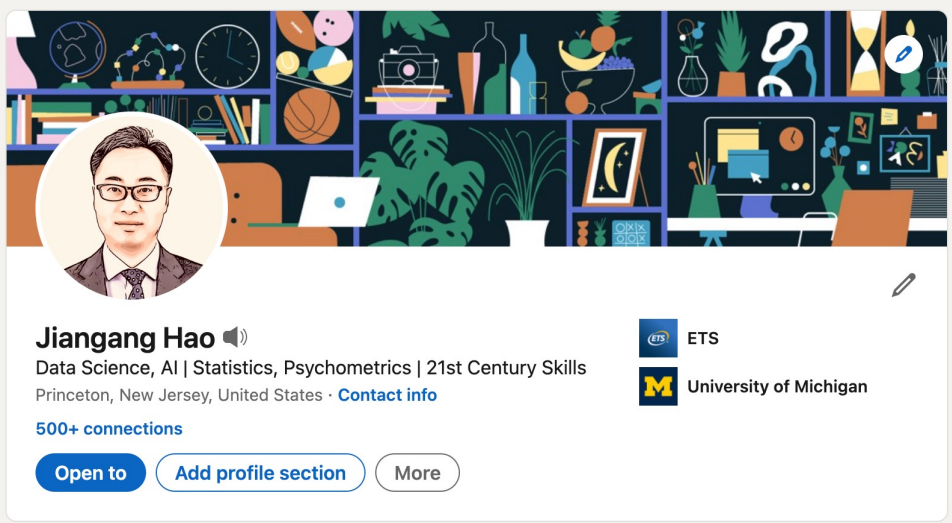
Machine Learning, Natural Language Processing, and their Application in Educational Assessment

Jiangang Hao
Psychometric and Data Science Modeling
Educational Testing Service

Training Workshop in 2022 Maryland Assessment Research Center Conference – Nov. 2nd, 2022

About me



https://www.linkedin.com/in/jiangang-hao/

- Data Science
- AI/Machine Learning/NLP/Statistics
- Collaborative Problem Solving
- 21st Century Skills
- Game/simulation-based Assessments
- Astrophysics & Cosmology

Psychometric and Data Science Modeling group @ ETS

Measuring critical intra-personal and inter-personal skills through EdTech and data-driven computational methods from data science, machine learning/AI, natural language processing, statistics/psychometrics and other quantitative disciplines

Learning Goals of this Workshop

- Learn the promises of data for digital assessments with examples
- Get a high-level understanding of
  - Machine learning
  - Natural language processing
- Know where to find resources for further learning
- Lecture + demo/hands-on exercise based on Python
- Workshop materials:

  https://github.com/jgbrainstorm/marc2022_trainingworkshop

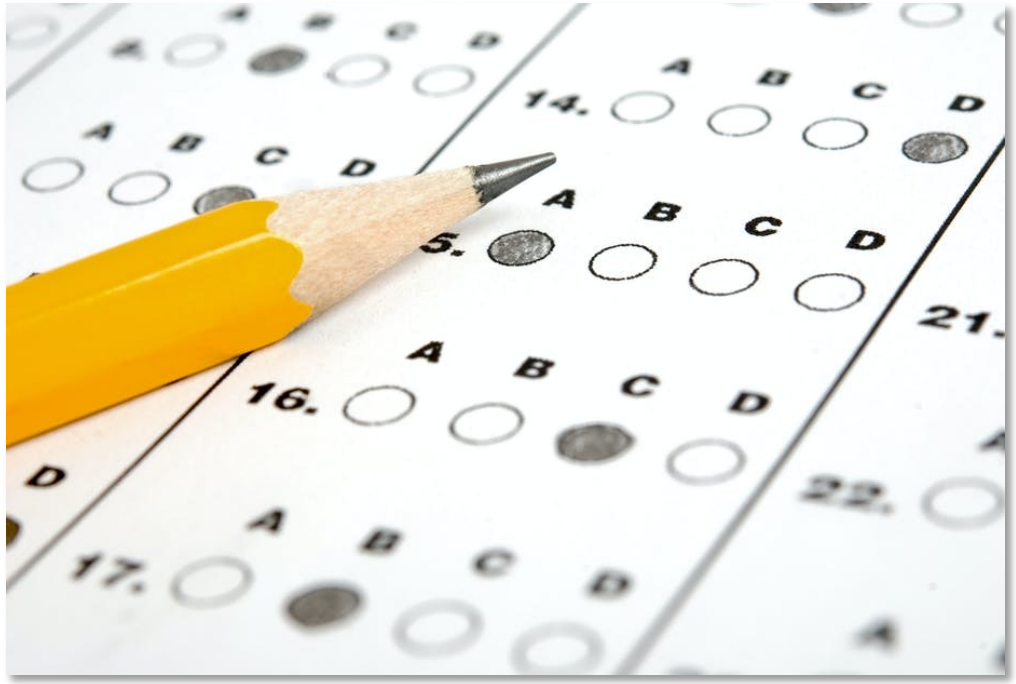
Session I: 9:00 AM – 12:00 PM
Session II: 1:00 PM – 4:00 PM

Description	Format	Duration
Overview of the data challenges from digital learning and assessment	Lecture	20 min
1.1 Introduction to machine learning basics • Supervised, unsupervised • SVM, Decision Tree, Random Forest, Gradient Boosting • Evaluation Metrics, ROC • Cluster analysis	Lecture	1 hour 30 min
1.2 Machine learning in Python: Scikit-learn	Demo & hands-on practice	30 min
1.3 Machine learning in GUI software: Orange	Demo & hands-on practice	30 min
Break		
2.1 Introduction to natural language processing basics • Text representation, N-gram, TF-IDF, word embedding • LSA/LDA • Automated Scoring • Deep learning-based language models	Lecture	1 hour 30 min
2.2 Natural language processing in Python: NLTK, spaCy	Demo and hands-on practice	1 hour
Wrap up/Q&A		30 min

Promises and Challenges of Digital Learning and Assessments

Be Prepared for A Mindset Shift

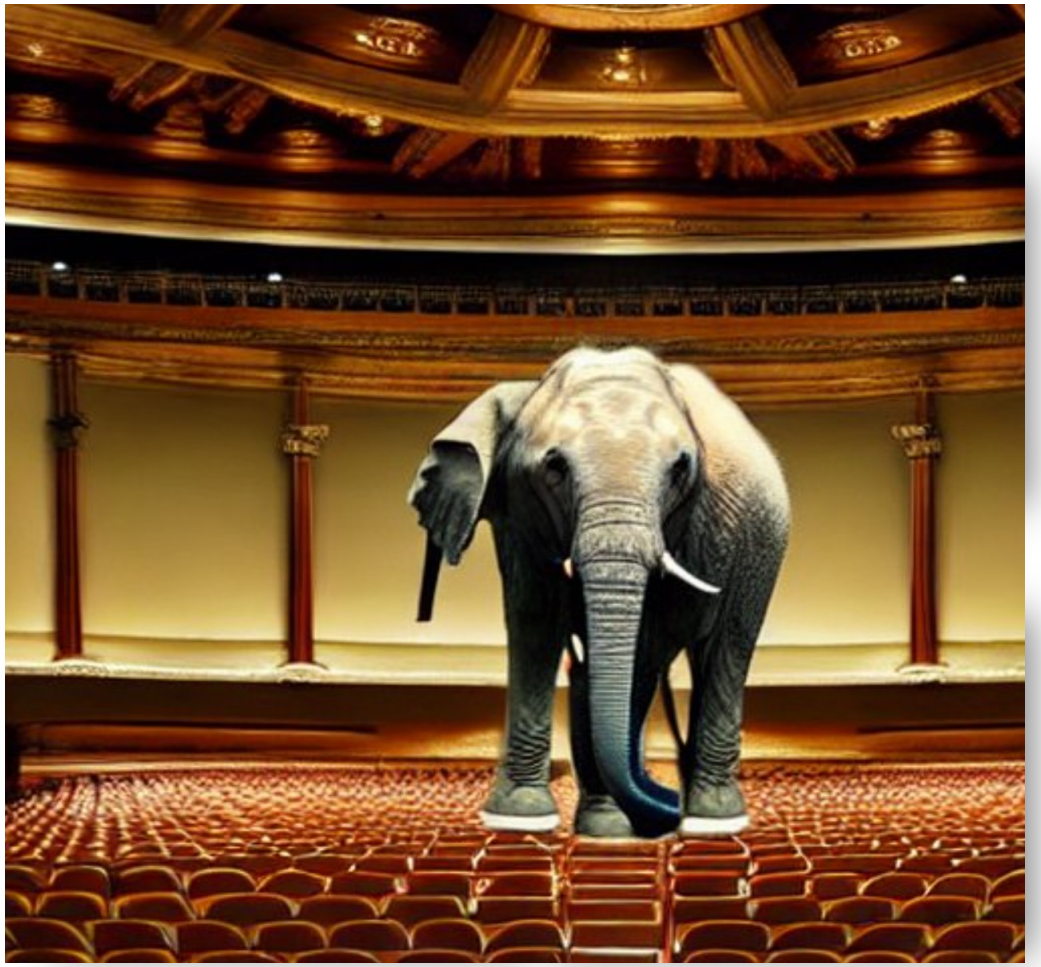
Assessments do not have to be always in this way



Digital technology affords much more

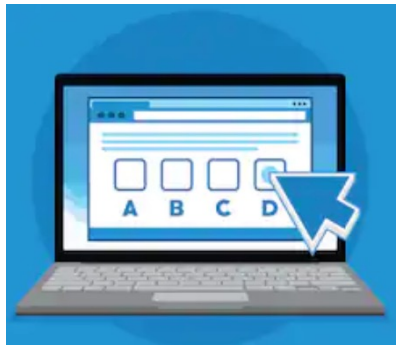
Impact of Digital Technology

- Digital technology completely changed the way of learning and assessment
  - Games, simulations, virtual reality, AI, online interaction, etc.
  - Performance/competency-based assessments for new skills become feasible
- Data are the key to materialize the promises of digital technology
  - Variety of new types of data from more authentic settings
  - Don’t just focus on exhausting ways to handle the 0/1 responses and miss the elephant in the room



Process Data in Assessments

Tasks



Interaction data

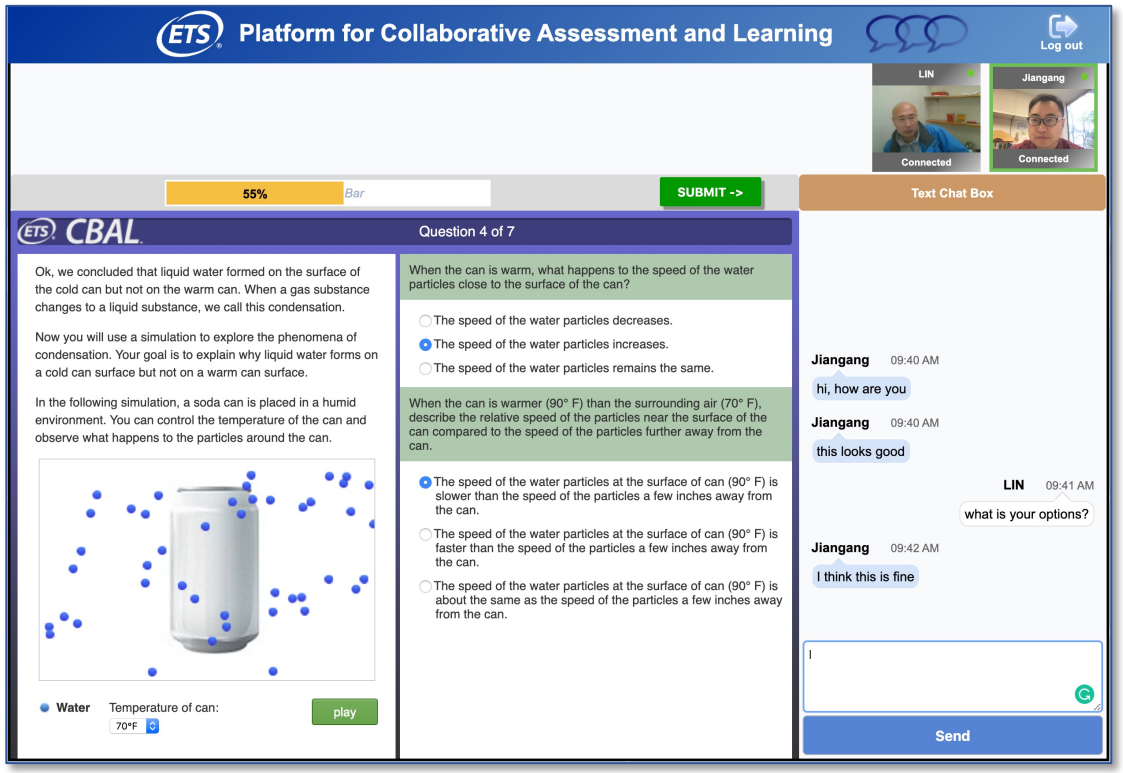
Learners

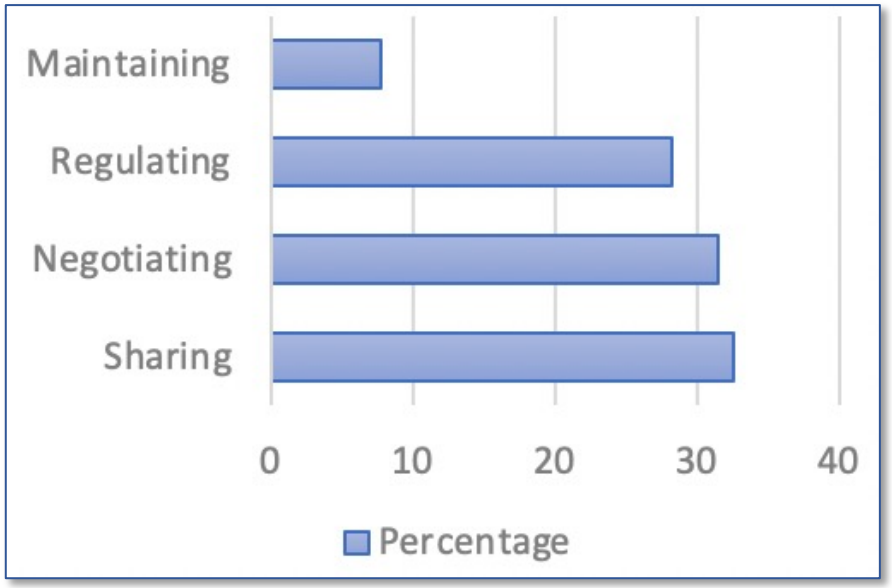
interaction information between learners and learning/assessment tasks, allowing better understanding of the learners and the tasks

Main uses:

1. Providing evidence for new constructs
2. Providing direct support to the psychometric operations
3. Informing designs of items and delivery platforms
4. Pushing the development of new psychometric and statistical models
5. Uncover new psychology and cognitive behaviors
6. Providing information to support test security and remote testing
7. Creating more informative reporting to feedback various stakeholders.
8. Providing information on group difference and shedding new light on fairness

Use Case 1 Example: Providing Evidence for New Constructs





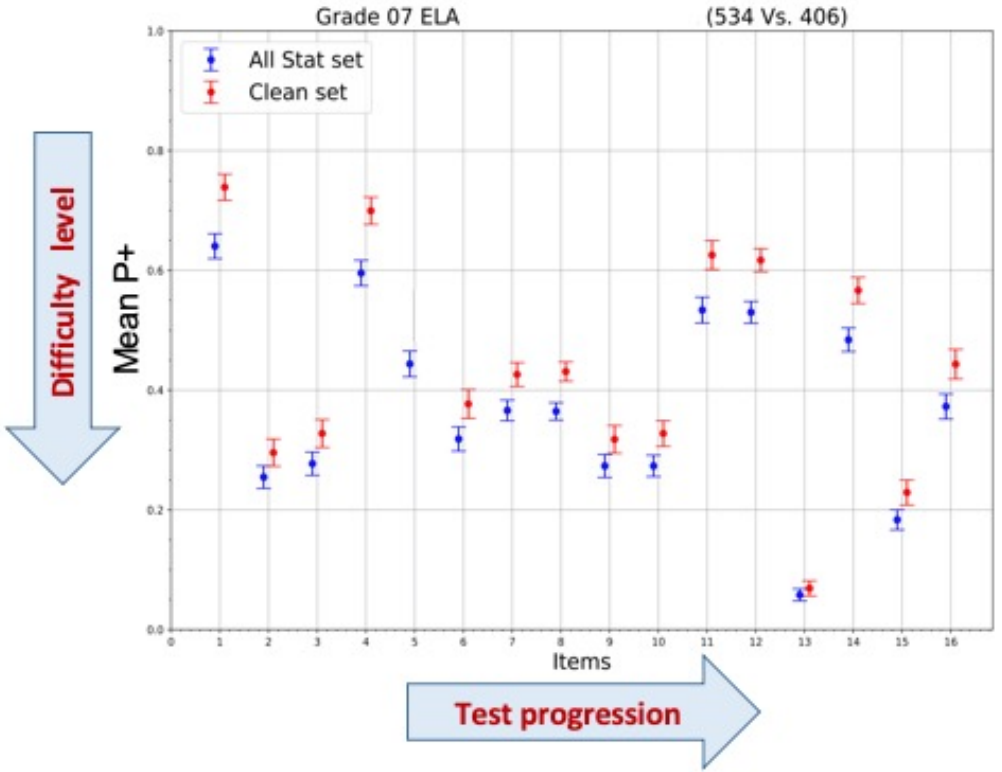
Hao, Liu, von Davier, Lederer, et al., 2017

Measuring collaborative problem solving based on the collaboration process data

Hao, NCME 2020

Use Case 2 Example: Support Psychometric Operations

Better item calibration



Definitions:

- Clean set: students who complete the assessment in a single session
- All Stat set: students who complete the assessment in a single session and in multiple sessions

Findings

- Different samples lead to different item parameter estimates
- The difference of item mean P+ is not affected much by the item location in the test
- Easier items are more affected

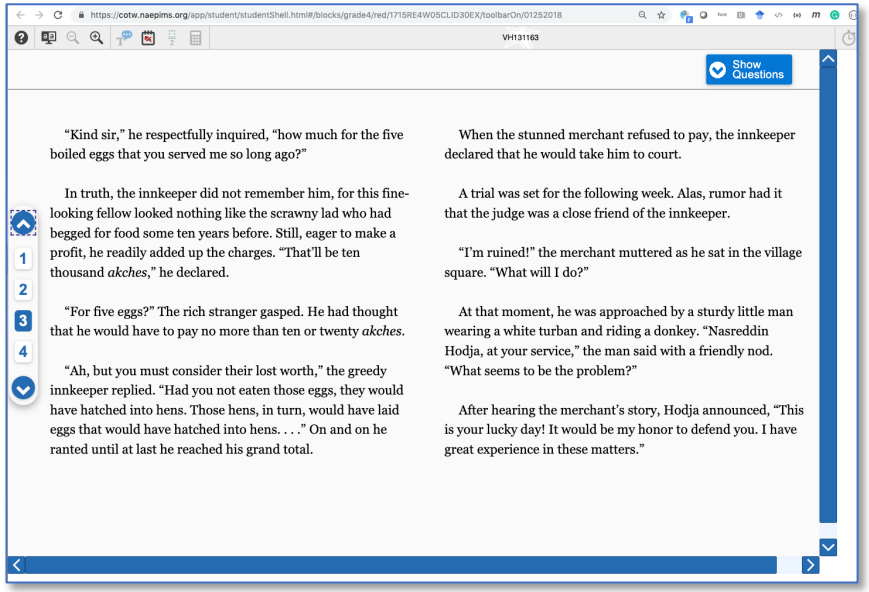
Based on data from ETS Winsight assessment

Hao, NCME 2019

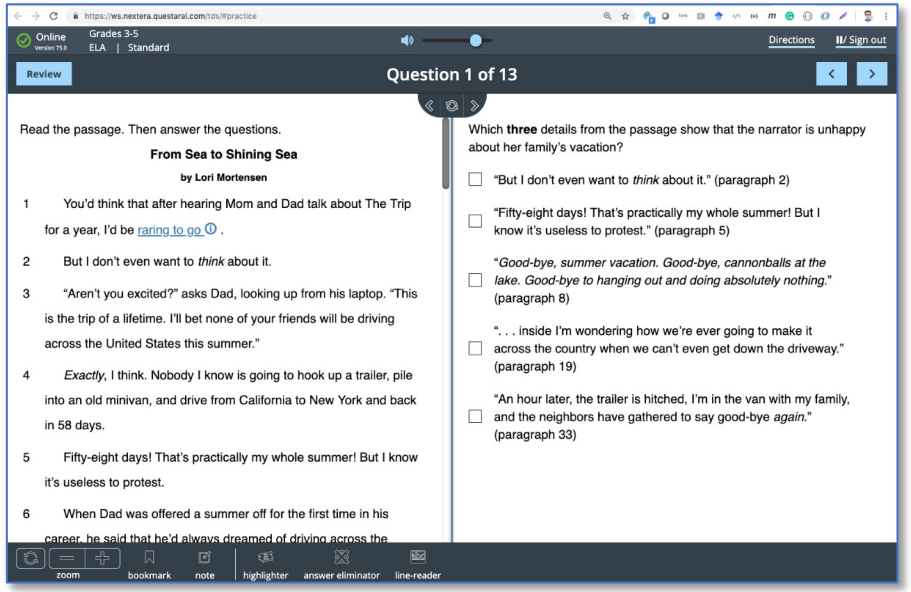
Use Case 3 Example: Informing Design

- For each item type, there should be an optimal design that allows us to get more information than others.
- Considering the reading item, the design on the left allows us to know the time student spend on reading passage and responding question

NAEP



Winsight

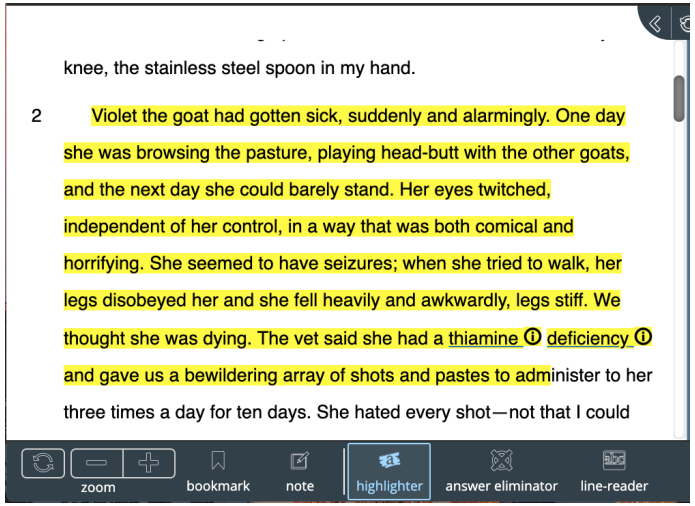


Process data provide an opportunity for us to summarize the optimal design for different item types to get the most information.

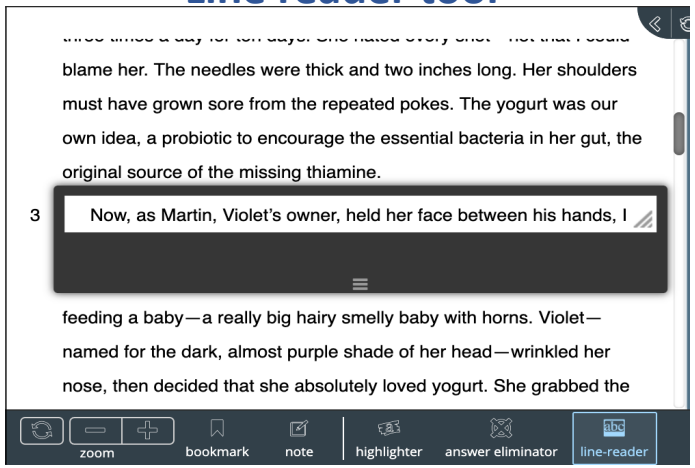
Hao, NCME 2019

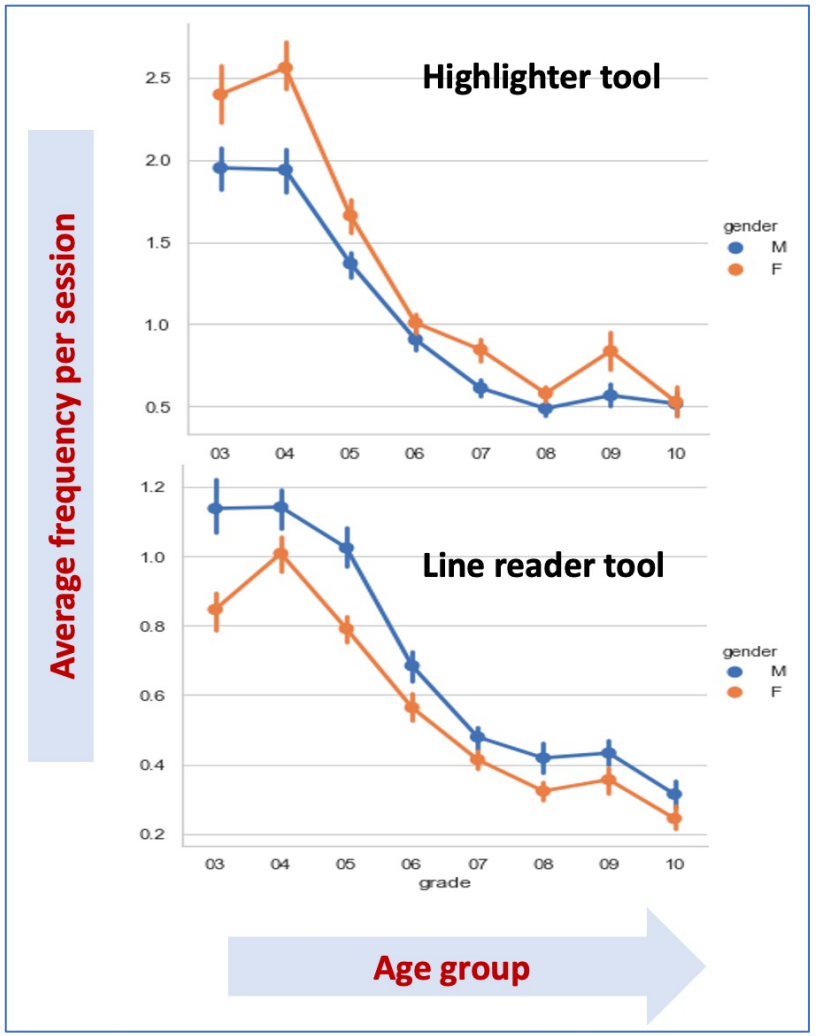
Use Case 5 Example: Uncover new psychological and cognitive behaviors

Highlighter tool



Line reader tool





Observations:

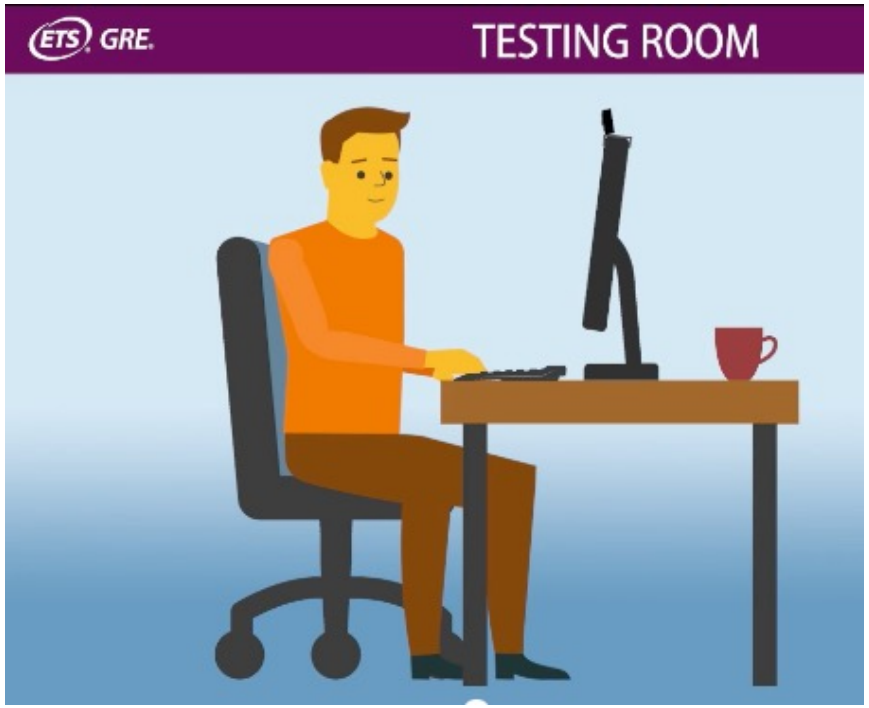
- Girls tend to use more highlighter tool
- Boys tend to use more line reader tool
- Decreasing trend of tool usage

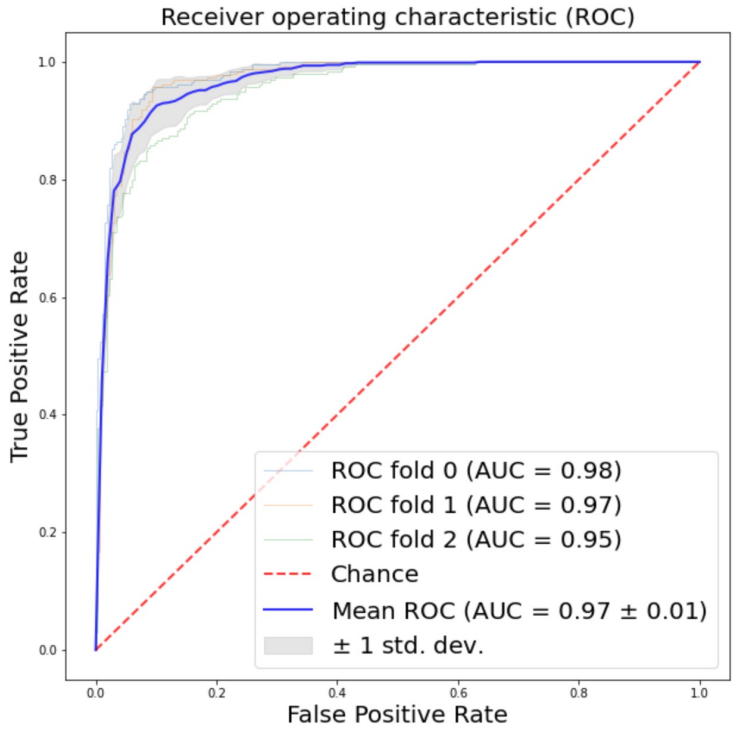
Key message:

- Analytics helps to identify subgroup difference to allow tailored strategy
- In business settings, marketing and sales strategy

Use Case 6 Example: Providing Information to Support Test Security

Detecting remote desktop access in remote testing based on clickstream data





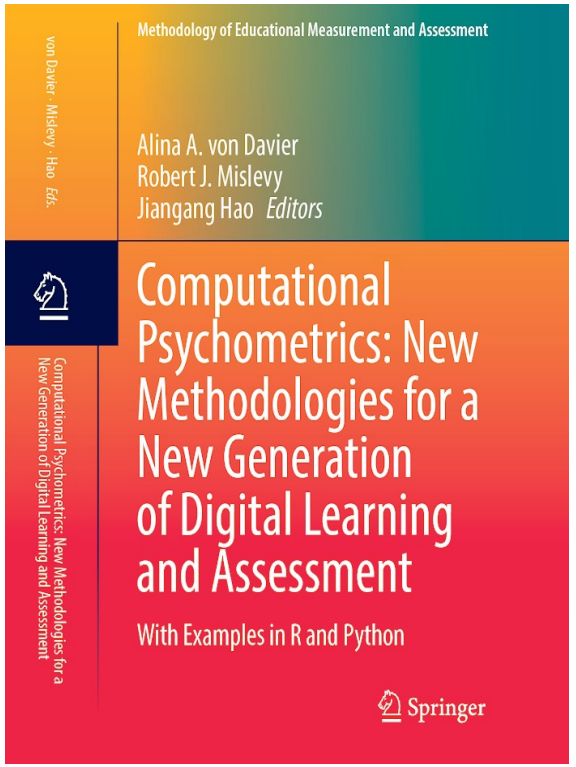
Hao & Li, 2021

Practical Challenges

- Who is going to do the work and where are we going to find them?
- New skills are needed
  - Design systems by leveraging the new digital affordances
  - Data science
  - Machine learning
  - Natural language processing
- How much should a psychometric researcher learn?
- Where to learn these skills?

A New Book

Computational Psychometrics is a term introduced by Alina von Davier in 2015 to describe an interdisciplinary field that fuses theory-based psychometric principles and data-driven computational methods from data science, machine learning, natural language processing, and other quantitative disciplines to handle the large-scale and high-dimensional data from digital learning and assessment.



https://bookauthority.org/books/new-educational-assessment-books?t=14nb4c&s=award&book=3030743934

Coverage

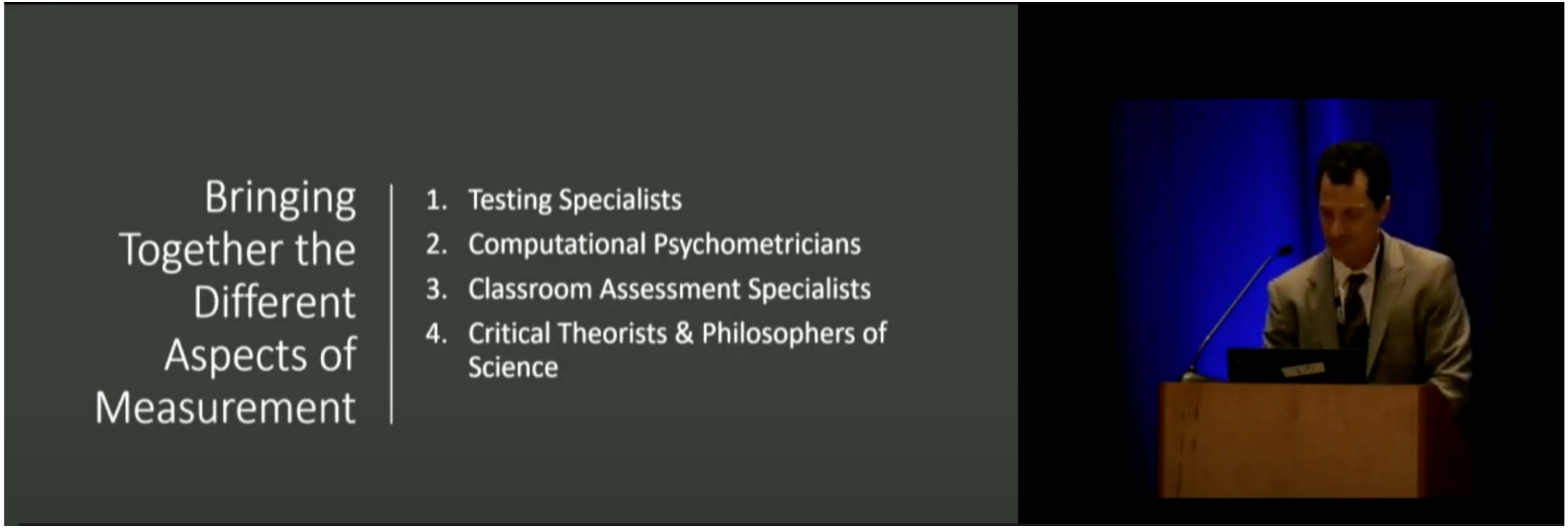
- Conceptualization
  - Next generation learning and assessment
  - Computational psychometrics
  - Virtual performance-based assessment
  - Adaptive learning
- Methodology
  - Concepts and models from psychometrics
  - Bayesian inference
  - Data science
  - Machine learning
  - Time series and stochastic process
  - Social network analysis
  - NLP - text mining and automated scoring
- Code examples

  https://github.com/jgbrainstorm/computational_psychometrics

Why We Need It?

- Intrinsic need to expand the existing psychometric methodologies to include new methods from, e.g., data science, machine learning, natural language processing and other quantitative disciplines, to address the new challenges of learning and assessment in the digital age. When many new methods are included, introducing a new term to encompass these new features will be more convenient and effective for communication.
- Practical challenges of preparing the workforce.
  - Applicants from psychometrics programs do not have the needed data science/machine learning skills (and mindsets) to process and model complex data from digital tasks
  - Applicants with data science/machine learning skills from other disciplines, such as computer science, generally know very little about the core values of psychometrics.
  - Hiring people who do not know the core values of the substantive area poses a big retention challenge for organizations, as they may quickly move on if they find they are not interested in the area at all after a few months.
  - It is imperative to prioritize a set of new methodologies and integrate them with the core values of psychometrics in a principled manner to help prepare a stable workforce for digital learning and assessment in the future.
- Bridge people from other quantitative disciplines (such as computer science, applied mathematics, physics, and others) to digital learning and assessment.
  - Providing a concise coverage of psychometrics' established values and methods could help them better understand how to apply their skills to join forces to promote learning and assessment in a digital age.

Computational Psychometrics in Measurement



Briggs, NCME Presidential Address, 2022

Prioritized Areas

- Machine learning/AI
  - Supervised and unsupervised learning
  - Some use cases
  - Software packages
- Natural Language Processing
  - Language models
  - Text representation and mining
  - Automated scoring
  - Deep learning-based models

Workshop

- Data science
  - Data science basics
  - Data wrangling and processing
  - Visualization and dashboarding

Homework

Machine Learning

- Machine learning basics
- Supervised learning
- Unsupervised learning
- Deep learning
- Software tools and packages

Chapter 9
Supervised Machine Learning

Jiangang Hao

Abstract Machine learning refers to a set of methodologies that allow computers to "learn" the relationship among numerical representations of data. In this Chapter, we focus on an important branch of machine learning, supervised machine learning, and introduce three widely used supervised learning methods, the Support Vector Machine, Random forest, and Gradient Boosting Machine. Python codes examples are included to show how to use these methods in practice.

9.1 Introduction

As highlighted in Chap. 6, one of the core missions of psychometrics is to ensure the observed evidence from assessment or learning tasks can support the claims on the targeted constructs in a valid, reliable, fair and comparable way. In the area of educational assessment, traditional psychometrics has given most attention to developing systematic methodologies to accomplish this mission when the evidence can be easily mapped into some simple forms, such as dichomotous or polymotous scores. The regularity and simplicity of the data make statistical inference well suited for modeling the data. However, digitally based learning and assessment tasks generate much more complex data. The complexity of these data makes it difficult (or sometimes impossible) to represent the information in simple scores in order to apply well-established statistical modeling frameworks, such as the familiar Item Response Theory (IRT). So, there is an intrinsic need for additional methodologies to harness the more complex data from digitally based

The R or Python codes can be found at the GitHub repository of this book: https://github.com/jgbrainstorm/computational_psychometrics

J. Hao (✉)
Educational Testing Service, Princeton, NJ, USA
e-mail: jhao@ets.org

Chapter 10
Unsupervised Machine Learning

Pak Chung Wong

Abstract The chapter introduces the concept of machine learning with an emphasis on unsupervised learning algorithms and applications. The discussion starts with a brief background on machine learning and then a high-level discussion on the differences between supervised and unsupervised learning algorithms. We present three categories of unsupervised machine learning techniques that include clustering, outlier detection, and dimension reduction; five prevailing unsupervised learning algorithms that include K-means, agglomerative clustering, DBSCAN, principal component analysis, and multidimensional scaling; and five Python programming examples that demonstrate the learning concepts and results using psychometric assessment data collected from an online collaborative problem-solving environment. This chapter demonstrates the potential of machine learning and highlights the opportunities it presents in psychometric research and development.

10.1 Introduction

Arthur Lee Samuel (Samuel, 1959), who coined the term "machine learning," defined the then new discipline as a sub-field of Computer Science that gives "computers the ability to learn without being explicitly programmed." Machine learning (Machine Learning, 2017) has since found its way into different data science communities, which include, among others, exploratory data analysis (Tukey, 1997), data mining (Fayyade t al., 1996), and visual analytics (Wong & Thomas, 2004). More recently, (Pedro Domingos, 2015) broke down the five tribes (or paradigms) of machine learning as symbolists, connectionists, evolutionaries,

The R or Python codes can be found at the GitHub repository of this book: https://github.com/jgbrainstorm/computational_psychometrics

P. C. Wong (✉)
University of Iowa, Iowa City, IA, USA

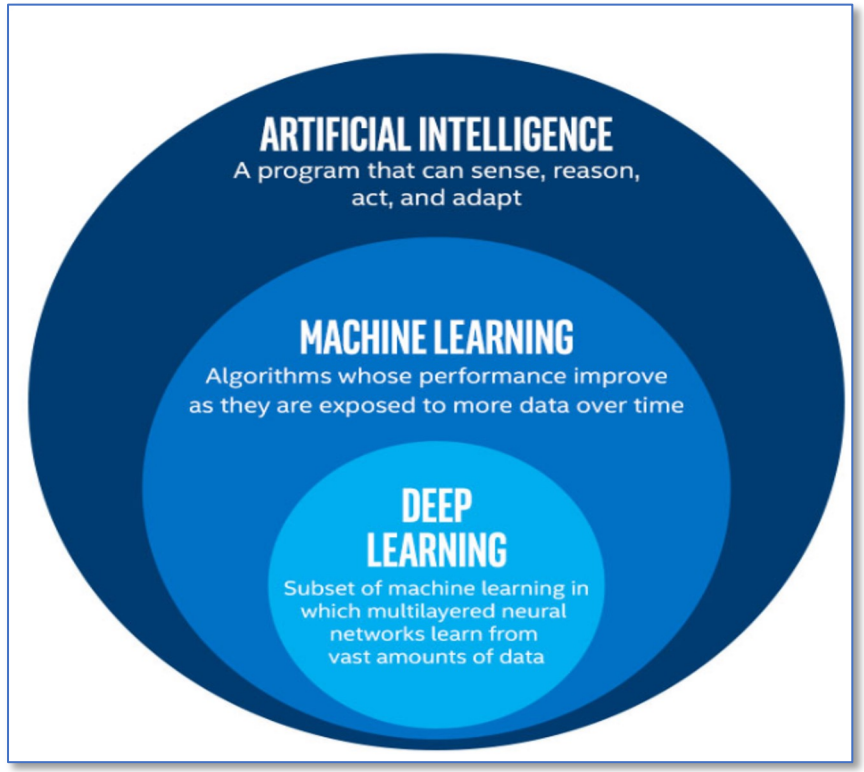
Chapter 11
Advances in AI and Machine Learning for Education Research

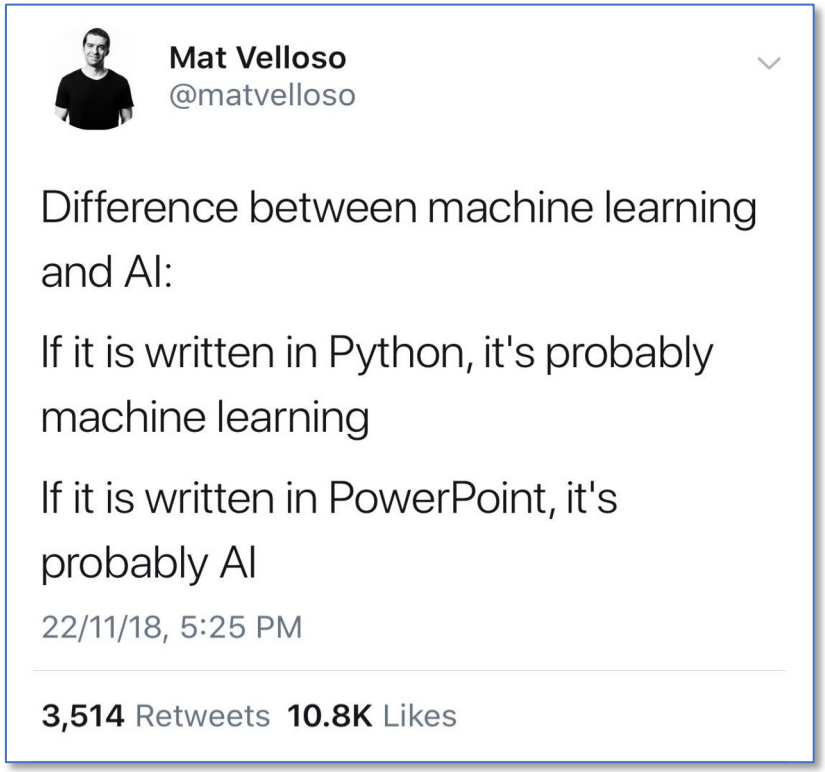
Yuchi Huang and Saad M. Khan

Machine Learning Basics

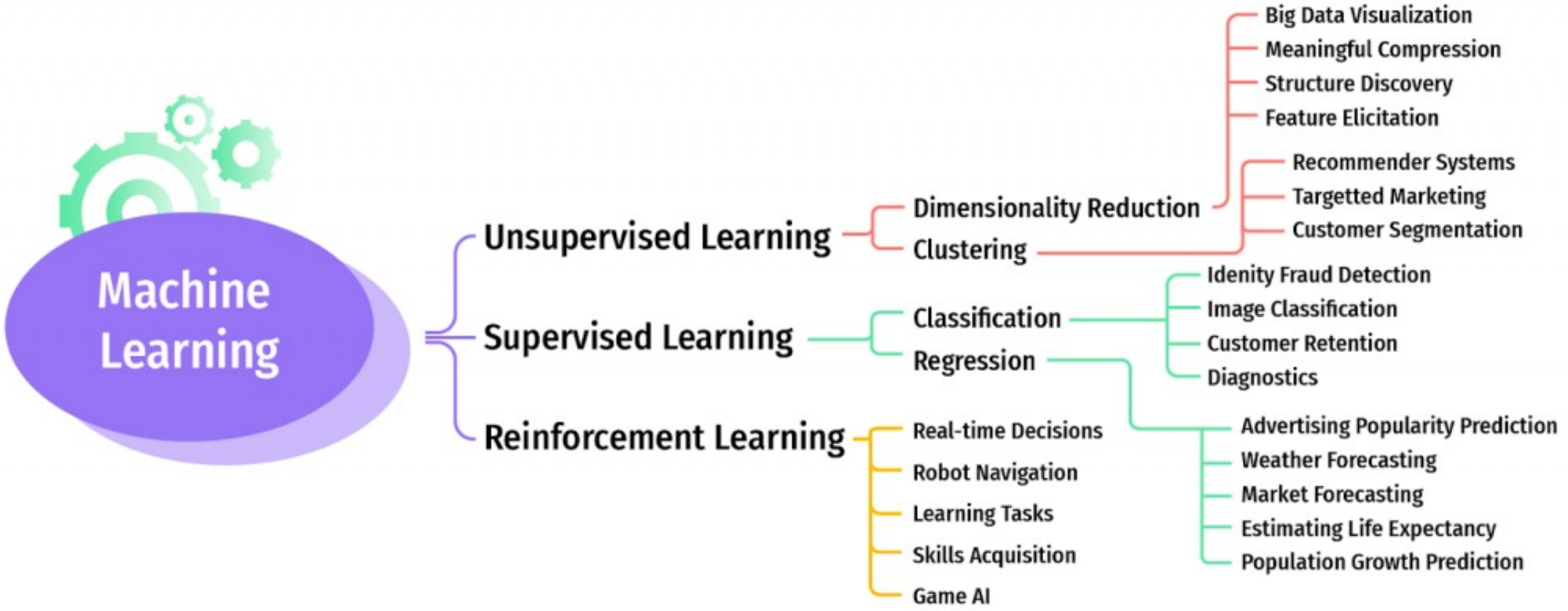
Machine Learning

A term coined by Auther Samuel in 1959, refers to a set of methodologies that allow computers to “learn” the relationship among numerical representations of data without explicit instructions by human experts



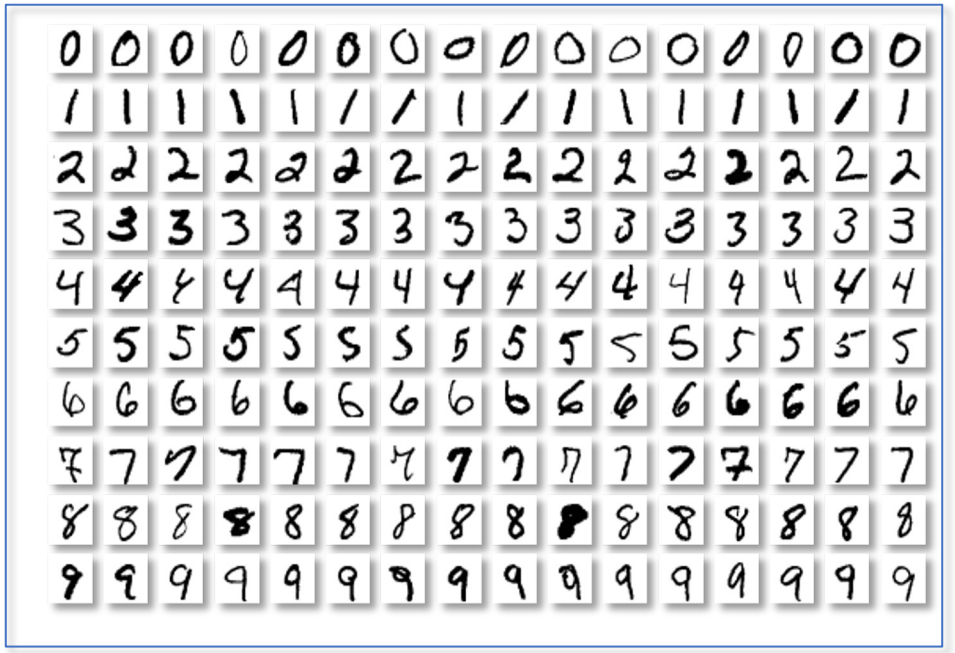


Main Paradigms of Machine Learning



Supervised learning

Machine learns the relationship (mapping) between independent variables (features) and dependent variables (labels)



data

Supervised learning Algorithms

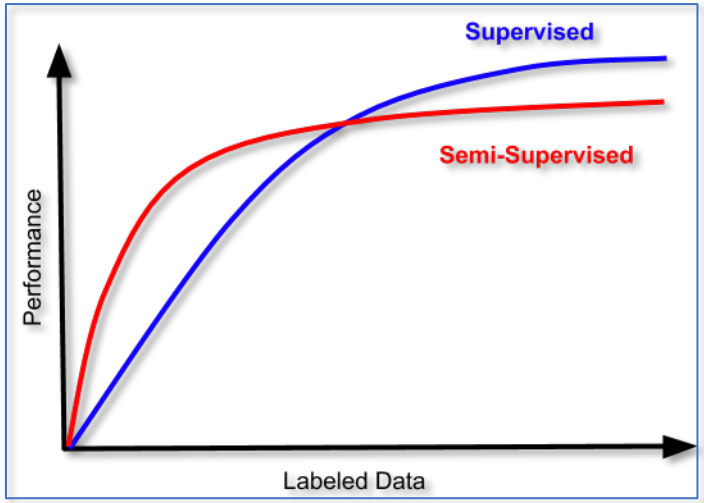
0
1
2
3
4
5
6
7
8
9

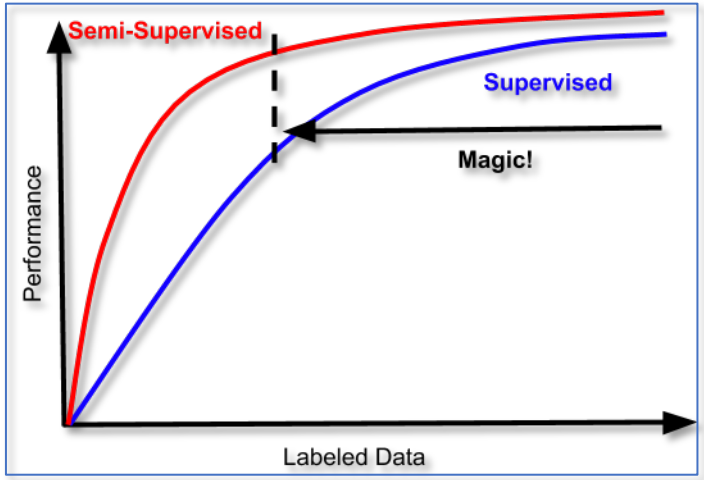
labels

Semi-supervised learning

The task of learning the mapping between features and labels when only part of the training data has labels

- Continuity assumption: Points that are close to each other are more likely to share a label
- Cluster assumption: The data tend to form discrete clusters, and points in the same cluster are more likely to share a label
- Manifold assumption: The data lie approximately on a manifold of much lower dimension than the input space.



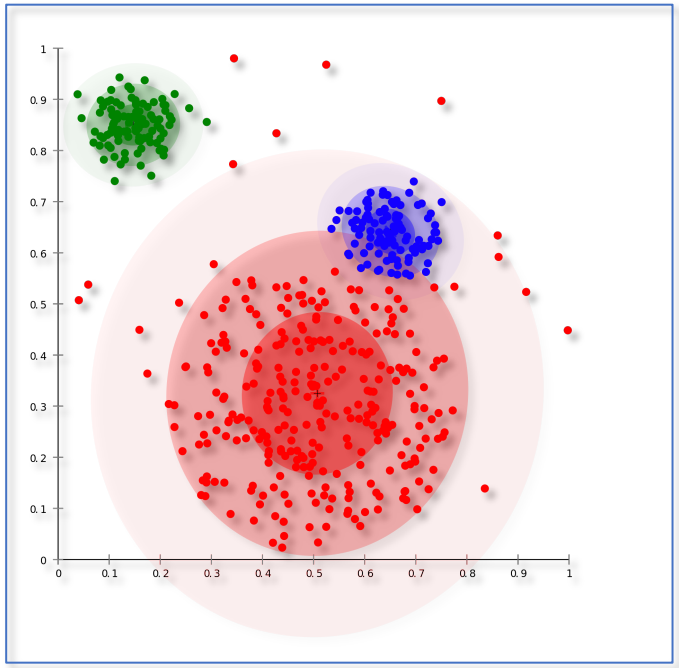


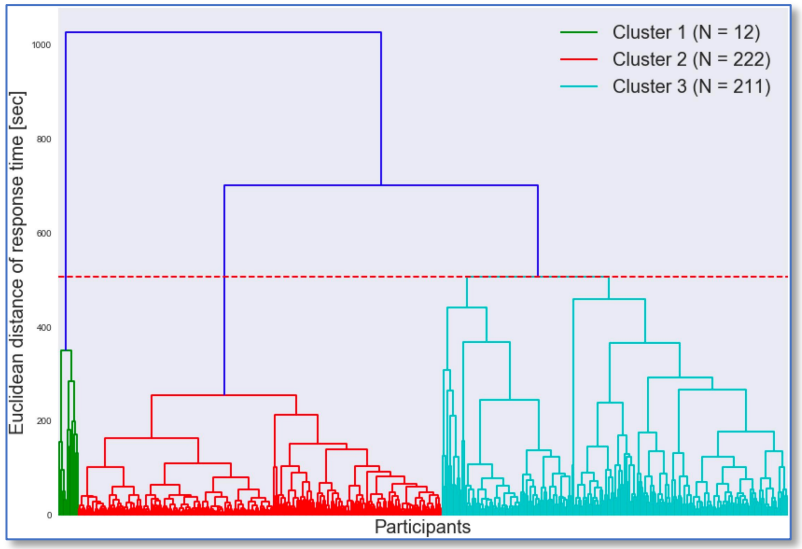
https://towardsdatascience.com/the-quiet-semi-supervised-revolution-edec1e9ad8c

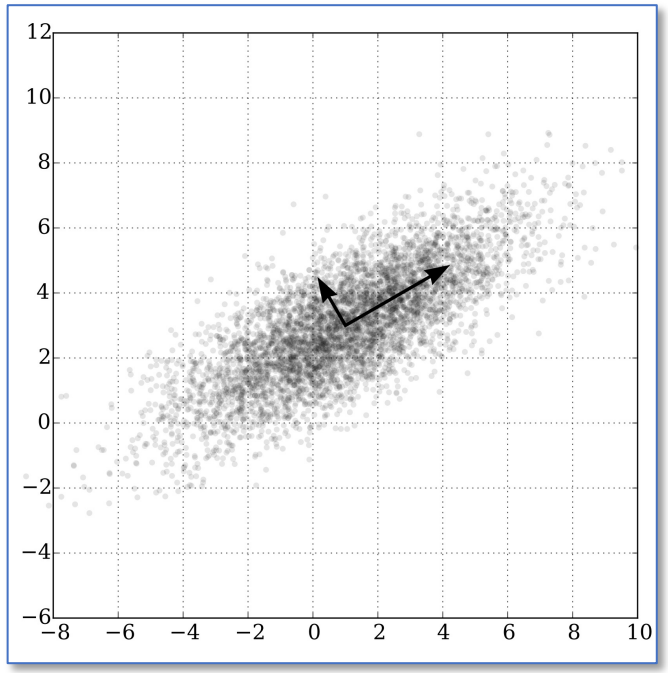
Unsupervised learning

Methodology that discovers concentrations, associations, or correlations in data.

- Cluster analysis
- Dimensionality reduction

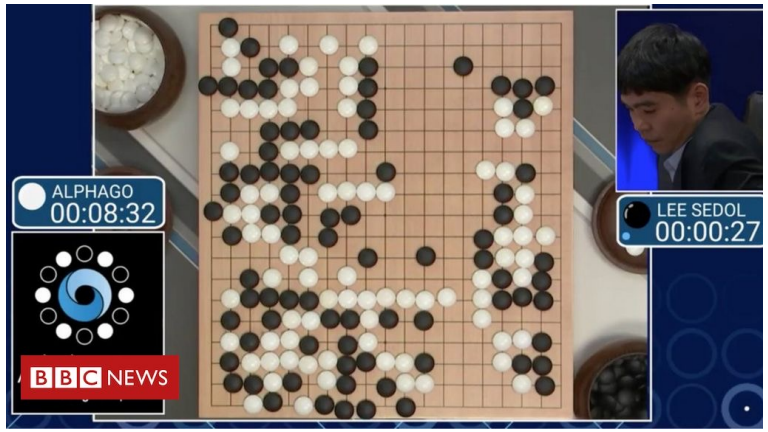


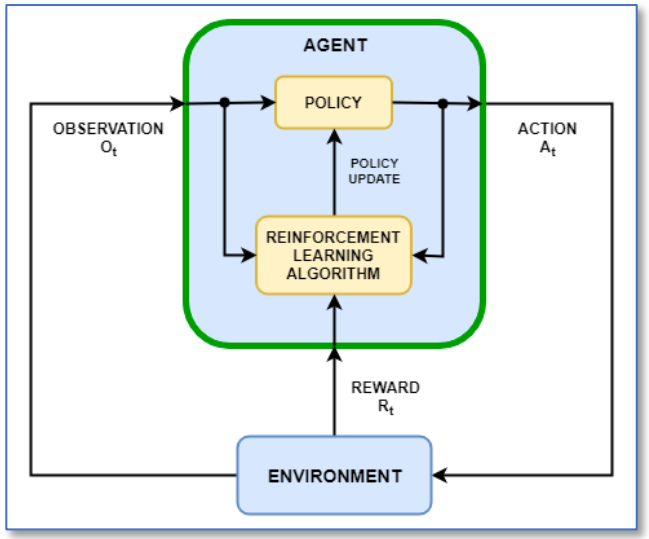




Reinforcement Learning

- Machine learns optimal strategies by means of a goal-oriented exploration of a parameter or state-space to optimize a reward function (Sutton, & Barto, 1998).
- Key components
  - Agent
  - States
    - Hidden
    - Observable (observation)
  - Actions: what the agent can do
  - Policy/strategy: tell what the agent should do
  - Reward: what the agent gets after an action

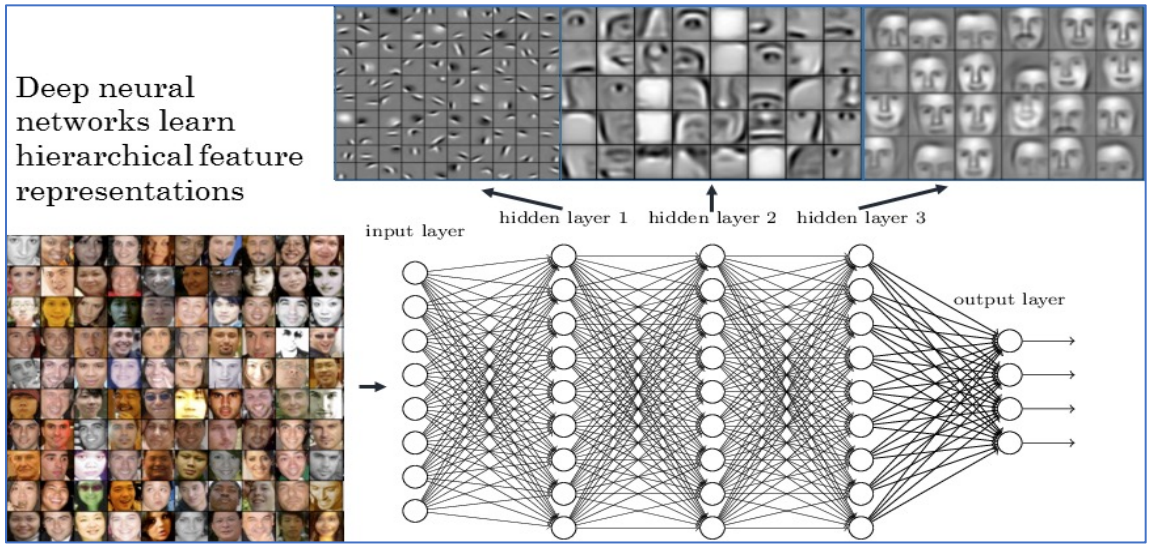




From MathWorks

Representation Learning

- In supervised learning, feature representation is a crucial step
- Features design
  - By human experts – time consuming and expensive
  - By algorithms - representation learning
- End-to-end learning



https://www.strong.io/blog/deep-neural-networks-go-to-the-movies

Supervised Machine Learning

Question 1

- How supervised learning is different from statistical inference, as they both aim at mapping the relationship between the independent and dependent variables?

I. Machine learning is more prediction-driven while statistical inference emphasizes both prediction and model parameter estimation.

II. Statistical inference emphasizes developing a probabilistic model to characterize the data and then estimating the model parameters based on some (usually well-studied) probability distribution functions, while machine learning emphasizes computation algorithms that can efficiently carry out the inference process by minimizing certain loss functions that do not necessarily have a probabilistic underpinning.

III. Statistical inference usually deals with data with a small number of variables obtained through planned *experiments* or quasi-experimental comparisons while machine learning handles sparse and high-dimensional data with a large number of variables (features), usually obtained through passive and uncontrolled *observations* (RWE).

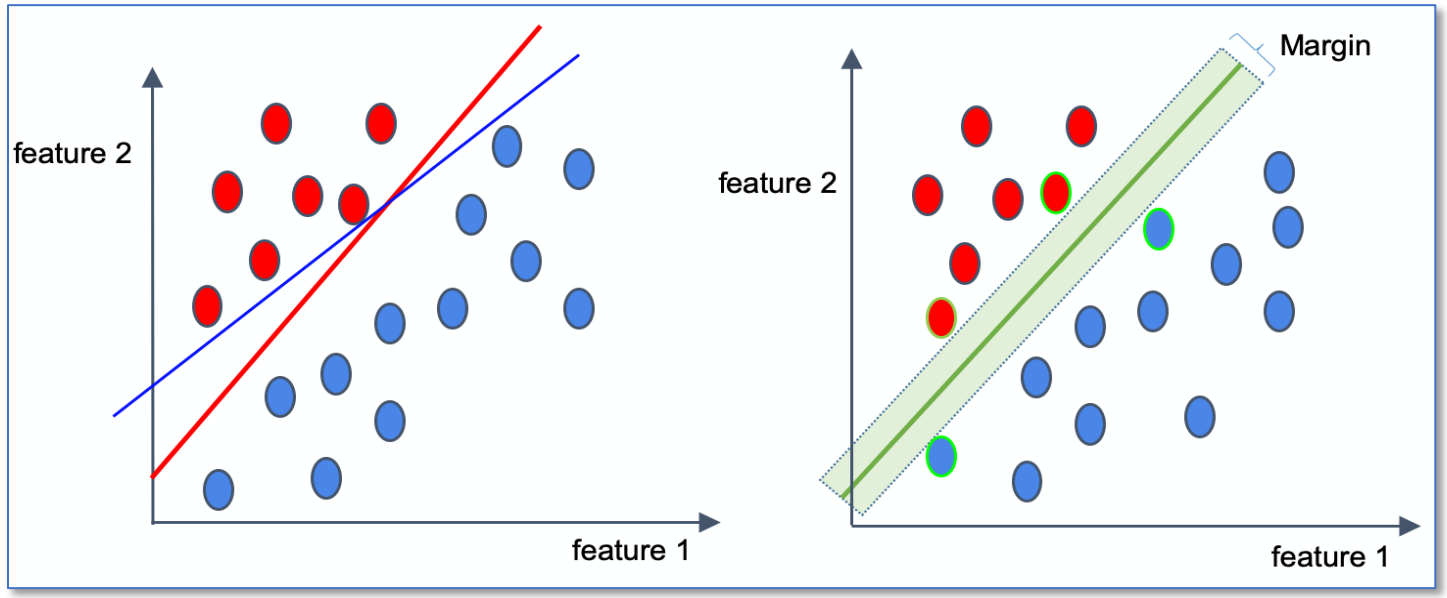
Hao & Ho, 2019; Hao, 2021

Question 2

- Which, if any, supervised learning methods are consistently superior to the others?
  - It is generally data and problem dependent
  - Some methods work well based on many explorations
    - SVM, Random Forest, XGBoost (Gradient Boosting Machine)
    - Deep Neural Network

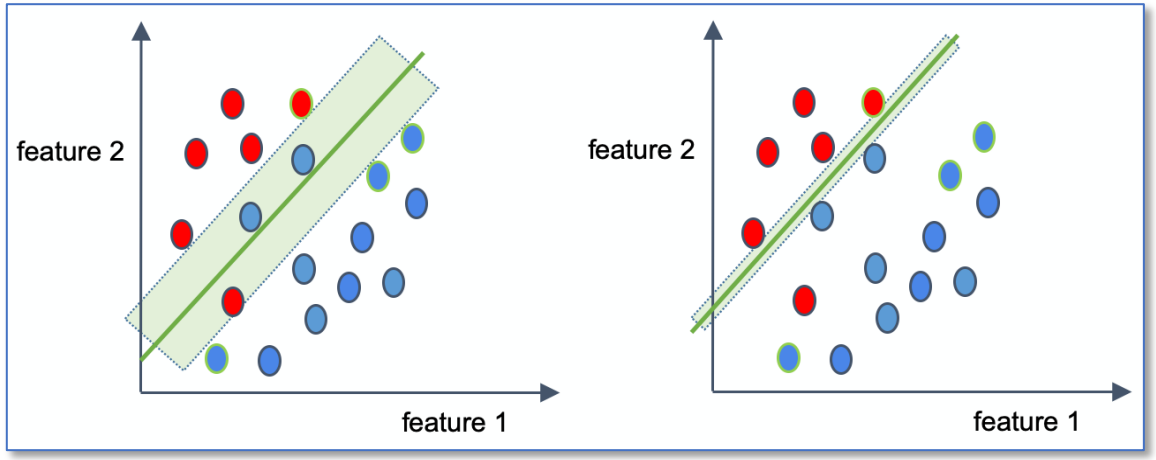
Support Vector Machine

It aims at finding a decision hyperplane in the feature space so that data points can be separated into different categories with a maximum margin that is defined as the distance of the closest points (support vectors) from each category to the decision hyperplane (Vapnik, 1963).



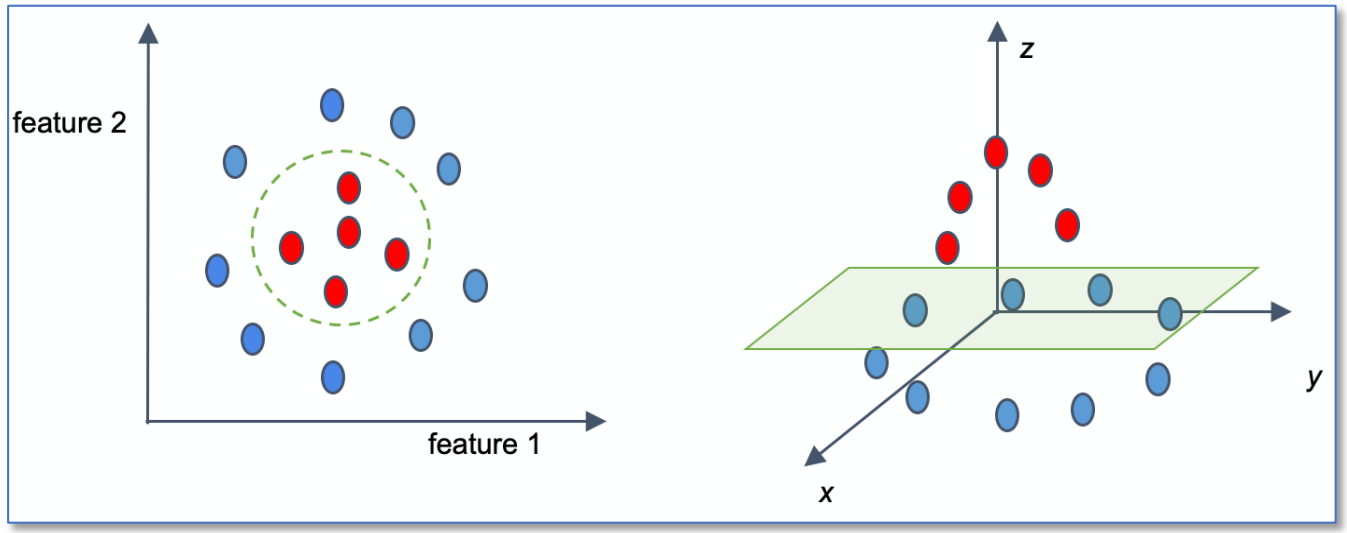
Non-linearly Separable Case

- Choose between a decision hyperplane that has a larger margin but more misclassified data points, and the one with a narrower margin but fewer misclassified data points, as illustrated in the left and right panels
- A hyperplane with a narrower margin may lead to better classification accuracy based on the training data, but is more susceptible to overfitting, while a hyperplane with a larger margin has lower classification accuracy with respect to the training data, but it is more robust against overfitting.
- In SVM algorithm, the regularization hyper-parameter (*C*), is used to control this balance. Choosing a larger *C* indicates that one favors a narrower margin with a lower tolerance of misclassified data points



Kernel Trick

- The transformation of the original feature space into a higher dimensional space to make the data linearly separable is done through a procedure called kernel trick.
- The characteristic of data being more likely to be linearly separable when being projected into a higher dimensional space via some non-linear transformation is known as Cover's theorem (Cover, 1965).
- In SVM algorithms, one can choose the kernel type – another hyperparameter. Typical kernels are polynomial and RBF.



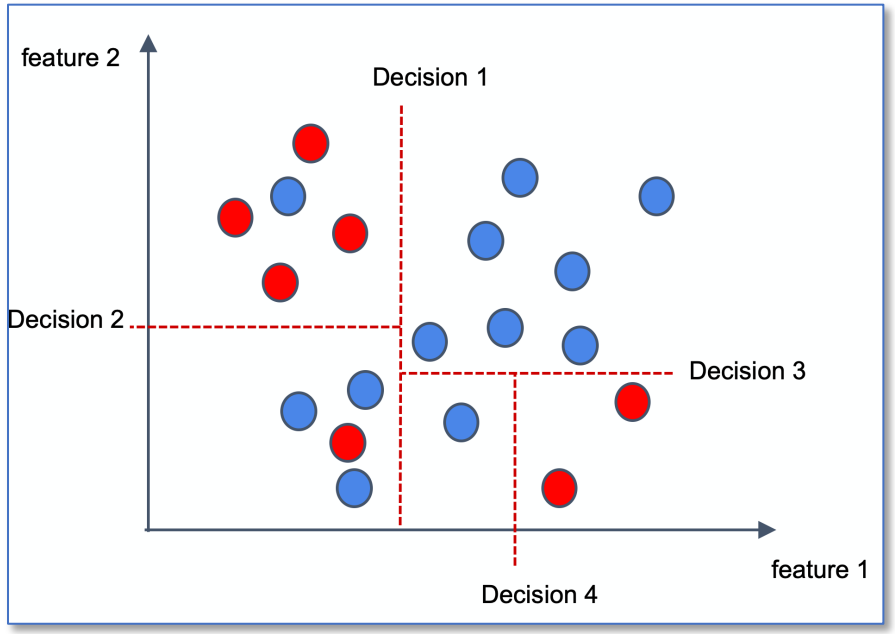
$$x = feature\ 1$$
$$y = feature\ 2$$
$$z = (feature\ 1)^2 + (feature\ 2)^2$$

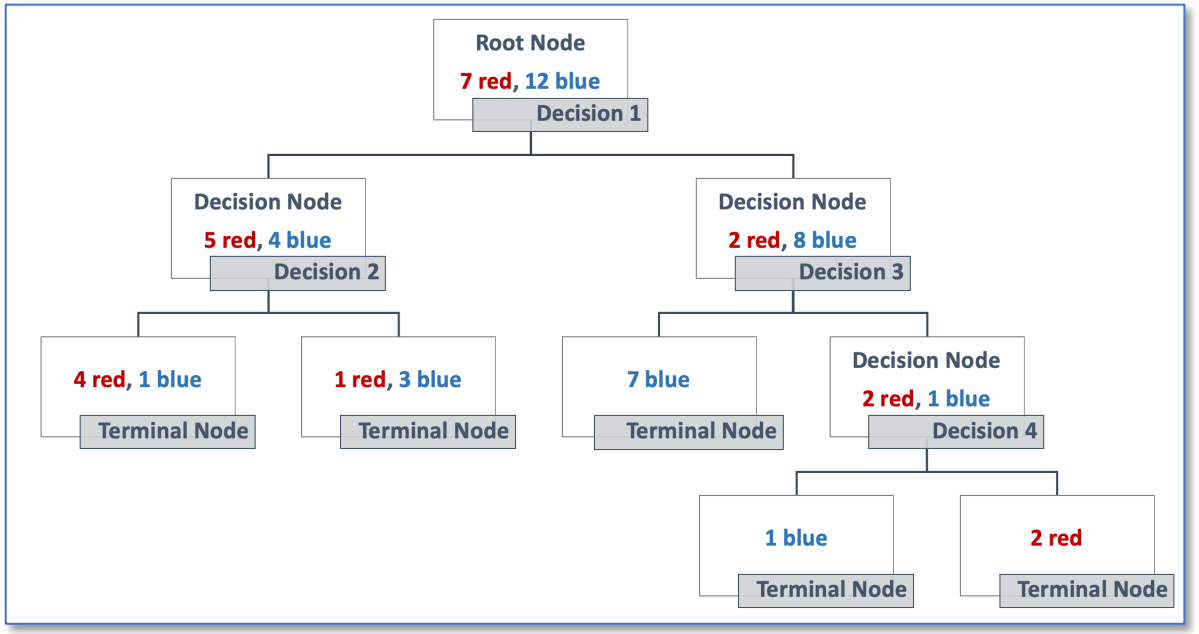
Ensemble Learning

- Given there are many supervised learning methods, it is always tantalizing to think whether some sort of “average” from an ensemble of learners will lead to better predictive performance. Research studies in this direction leads to ensemble learning (Dietterich, 2000).
- Two leading ideas regarding how an ensemble of learners is constructed:
  - Bagging (bootstrap aggregating) (Breiman, 1996)
    - The ensemble is created through bootstrapping the dataset (Efron & Tibshirani, 1994)
    - The outputs from the learners from each sample will be combined (average or voting) to improve the predictive performance.
    - Example: Random Forest (Ho, 1995; Breiman, 2001)
  - Boosting (Friedman, 2001)
    - The ensemble is constructed sequentially, rather than in parallel, as in the bagging approach.
    - A learner is first trained on the full dataset and subsequent learners are added by fitting the residuals (after applying all preceding learners), by which new learners will assign more weights to the poorly predicted observations by the preceding learners.
    - Example: Gradient Boosting Machine

Decision Tree

A decision tree works by forming decision rules on the features recursively to minimize some loss function of the classification.





Decision Tree Cont'd

- Loss function
  - Gini index as used in the CART algorithm (Breiman et al., 1984).
  - Information Gain as used in the Iterative Dichotomiser 3 algorithm (Quinlan, 1986).
- Stopping rule
  - Condition under which no further splitting will happen.
  - E.g., minimum leaf size
- Pruning
  - A procedure to remove the sub-nodes from decision nodes performed after the training to avoid overfitting.
- Main drawback:
  - Instability of the results, as a small change in a dataset may lead to quite different decision rules.

Random Forest

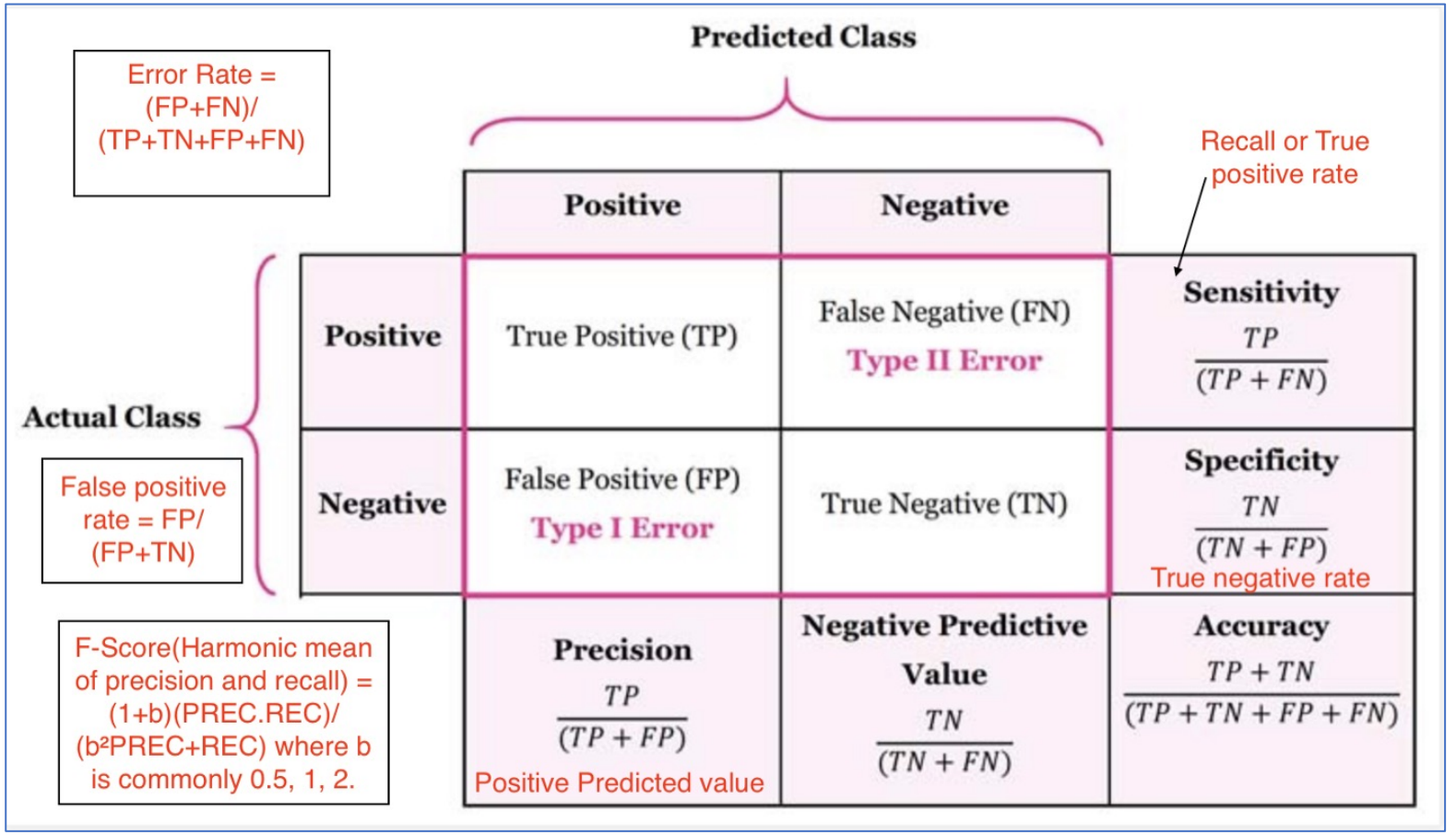
- Suppose we have N observations and M features in a dataset, then a typical random forest classifier goes as follows:
  1. Create K samples of the dataset using the bootstrapping technique, and each sample has N observations
  2. For each of the K samples, a decision tree is applied. In each of the decision nodes of each of the trees, a decision is made based on the most discriminative one of a randomly chosen F features from the total M features – random subspace or feature bagging (Ho, 1998).
  3. The average/voting of multiple trees will be the final estimator
- Hyper-parameters
  - The number of trees (K)
  - The size of the feature subspace (F)

Gradient Boosting Machine

- Assumption: the ideal mapping between the dependent and independent variables could be approximated by the sum of many functions belonging to the same parametric family.
- The parameters are fixed iteratively as following
  1. Fit the data ($\boldsymbol{X}$, y) with a base tree $h_0(\boldsymbol{X};\boldsymbol{\alpha}_0)$
  2. Add another tree $h_1(\boldsymbol{X};\boldsymbol{\alpha}_1)$ by fitting the residuals $(\boldsymbol{X}, y - h_0(\boldsymbol{X};\boldsymbol{\alpha}_0))$
  3. Repeat the procedure leads to $F(\boldsymbol{X}) = \sum_{m=0}^{M} \beta_m h_m(\boldsymbol{X};\boldsymbol{\alpha}_m)$
- Where is the "Gradient" from?
  - For loss function: $L(y, F(\boldsymbol{X})) = \frac{1}{2}[y - F(\boldsymbol{X})]^2$
  - The Residual: $y_i - F_m(\boldsymbol{X}_i) = -\frac{\partial L(y_i,\ F(\boldsymbol{X}_i))}{\partial F(\boldsymbol{X}_i)}\Big|_{F(\boldsymbol{X})=F_{m-1}(\boldsymbol{X})} \equiv -g_m(\boldsymbol{X}_i)$

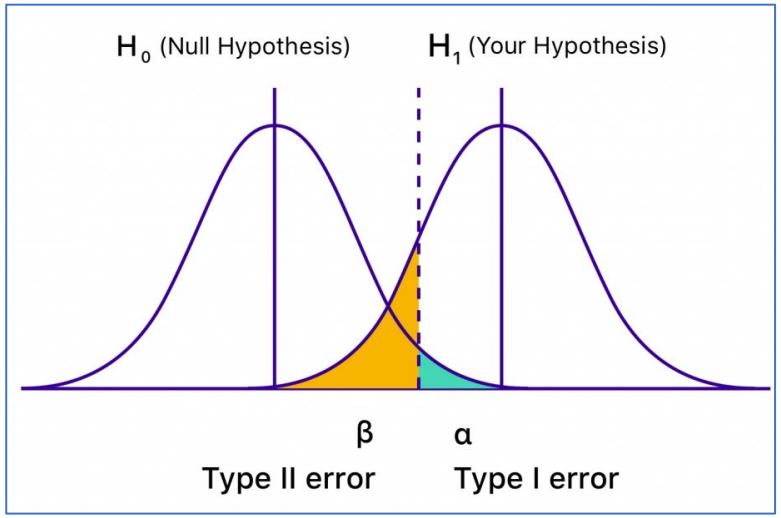
See the book chapter for more details

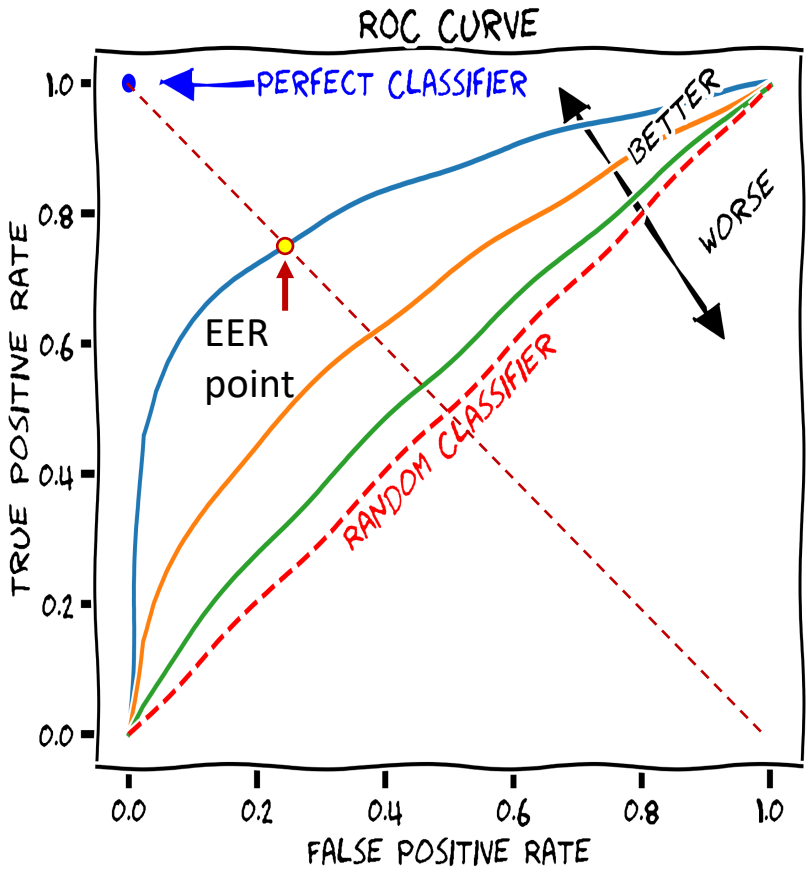
Evaluation Metrics for Binary Classification



https://en.wikipedia.org/wiki/Evaluation_of_binary_classifiers

Receiver Operating Characteristic - ROC



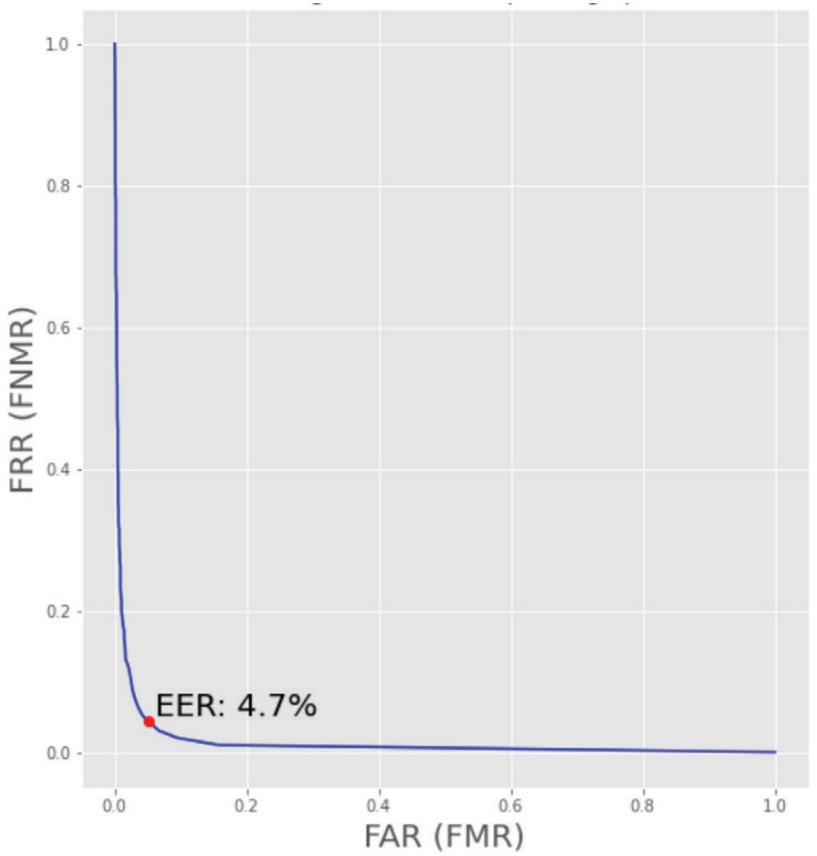


- Area under curve (AUC):
  - 0.5: non discriminative
  - 0.7 – 0.8: acceptable
  - 0.8 – 0.9: excellent
  - > 0.9: outstanding
- Equal Error Rate
  - Fingerprint: 0.2%
  - Signature: 2%
  - Voice: 2%
  - Keystroke: 4.7%
  - TouchID: 0.002%
  - FaceID: 0.001%

Example Application – Keystroke Biometrics

- Writing process data – keystrokes
- Feature representation of keystrokes
- Mapping between keystroke features and labels (same person or not)

Feature name	Definition	Within-person correlation
inword_logIKI_median*	Median duration of in-word keystrokes, measured in log milliseconds	0.95
inword_logIKI_mean	Mean duration of in-word keystrokes in log milliseconds	0.95
wordinitial_logIKI_median*	Median duration of word-initial keystrokes in log milliseconds	0.92
append_interword_interval_logIKIs_mean	The mean log interkey intervals for keystrokes that add white space between words	0.92
wordinitial_logIKI_mean	Mean duration of word-initial keystrokes in log milliseconds	0.92
append_interword_interval_logIKIs_median*	The median log interkey interval for keystrokes that add white space between words	0.91
append_interword_interval_speed_median*	The speed of keystrokes that add white space between words, measured in characters per second	0.91
wordinitial_char_per_sec_median*	Median speed of typing the first character of a word, in characters per second	0.91
iki400_AppendBurst_len_mean	Mean length in characters of bursts of append keystrokes where no pause is greater than 400 milliseconds	0.91
iki400_AllActionBurst_len_mean	Mean length in characters of bursts where all keystrokes count as part of the burst, and bursts end on pauses longer than 400 milliseconds	0.90
initial_backspace_char_per_sec_median*	The median speed of the first in a series of backspace actions, measured in characters per second	0.90
iki200_AppendBurst_len_mean	Mean length in characters of bursts of append keystrokes where no pause is longer than 200 milliseconds	0.90
initial_backspace_logIKI_median*	The median log interkey interval for backspace actions that appear first in a series of backspace actions	0.89



(Choi, Hao, Deane & Zhang, 2021)

Unsupervised Machine Learning

Cluster Analysis

- Group similar things together
- What is “similar”
  - Similarity is measured by certain distance in a metric space

A **metric space** is an ordered pair (M, d) where M is a set and d is a metric on M, i.e., a function

$$d\colon M \times M \to \mathbb{R}$$

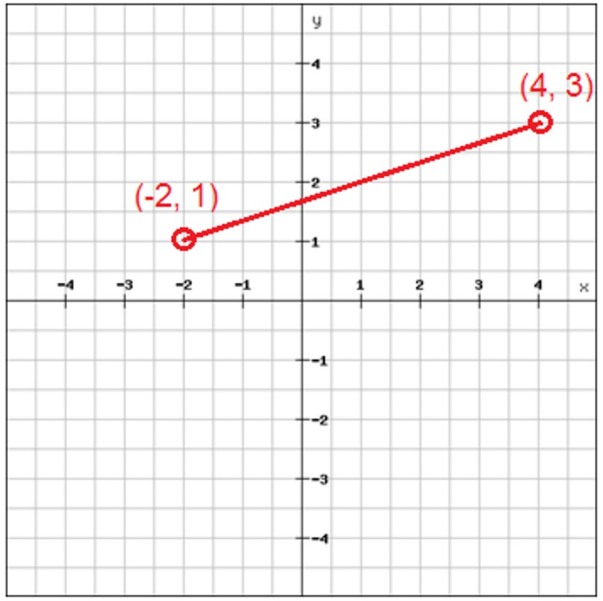
such that for any $x, y, z \in M$, the following holds:[2]

1. $d(x, y) = 0 \Leftrightarrow x = y$ identity of indiscernibles
2. $d(x, y) = d(y, x)$ symmetry
3. $d(x, z) \leq d(x, y) + d(y, z)$ subadditivity or triangle inequality

- Clusters are not unique, depending on the ways you look at or interpret the data

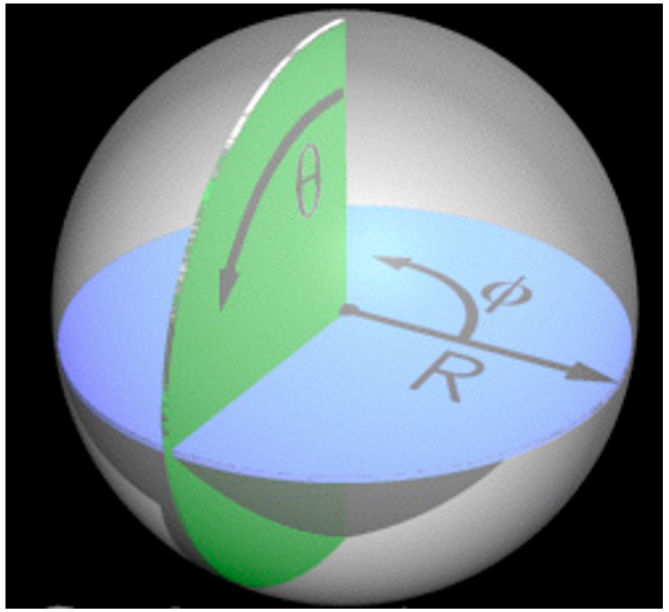
Example Distance

- Euclidean Space



$$ds^2 = dx^2 + dy^2$$

- Non-Euclidean Space



$$ds^2 = R^2 d\theta^2 + R^2 \sin^2\theta d\phi^2$$

Popular Distances in ML

- Euclidean distance
- Minkowski distance $D(X,Y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{\frac{1}{p}}$
- Mahalanobis distance $d(\vec{x},\vec{y}) = \sqrt{(\vec{x}-\vec{y})^T S^{-1} (\vec{x}-\vec{y})}$
- Cosine distance $1 - \frac{u \cdot v}{||u||_2 ||v||_2}$
- Edit distance: the minimum number of editing operations to bring two strings the same

  Analyzing Process Data from Game/Scenario-Based Tasks: An Edit Distance Approach (Hao, Shu, & von Davier, 2015)

Distances in SciPy package

Distance functions between two numeric vectors `u` and `v`. Computing distances over a large collection of vectors is inefficient for these functions. Use `pdist` for this purpose.

braycurtis(u, v[, w])	Compute the Bray-Curtis distance between two 1-D arrays.
canberra(u, v[, w])	Compute the Canberra distance between two 1-D arrays.
chebyshev(u, v[, w])	Compute the Chebyshev distance.
cityblock(u, v[, w])	Compute the City Block (Manhattan) distance.
correlation(u, v[, w, centered])	Compute the correlation distance between two 1-D arrays.
cosine(u, v[, w])	Compute the Cosine distance between 1-D arrays.
euclidean(u, v[, w])	Computes the Euclidean distance between two 1-D arrays.
jensenshannon(p, q[, base])	Compute the Jensen-Shannon distance (metric) between two 1-D probability arrays.
mahalanobis(u, v, VI)	Compute the Mahalanobis distance between two 1-D arrays.
minkowski(u, v[, p, w])	Compute the Minkowski distance between two 1-D arrays.
seuclidean(u, v, V)	Return the standardized Euclidean distance between two 1-D arrays.
sqeuclidean(u, v[, w])	Compute the squared Euclidean distance between two 1-D arrays.
wminkowski(u, v, p, w)	Compute the weighted Minkowski distance between two 1-D arrays.

Distance functions between two boolean vectors (representing sets) `u` and `v`. As in the case of numerical vectors, `pdist` is more efficient for computing the distances between all pairs.

dice(u, v[, w])	Compute the Dice dissimilarity between two boolean 1-D arrays.
hamming(u, v[, w])	Compute the Hamming distance between two 1-D arrays.
jaccard(u, v[, w])	Compute the Jaccard-Needham dissimilarity between two boolean 1-D arrays.
kulsinski(u, v[, w])	Compute the Kulsinski dissimilarity between two boolean 1-D arrays.
rogerstanimoto(u, v[, w])	Compute the Rogers-Tanimoto dissimilarity between two boolean 1-D arrays.
russellrao(u, v[, w])	Compute the Russell-Rao dissimilarity between two boolean 1-D arrays.
sokalmichener(u, v[, w])	Compute the Sokal-Michener dissimilarity between two boolean 1-D arrays.
sokalsneath(u, v[, w])	Compute the Sokal-Sneath dissimilarity between two boolean 1-D arrays.
yule(u, v[, w])	Compute the Yule dissimilarity between two boolean 1-D arrays.

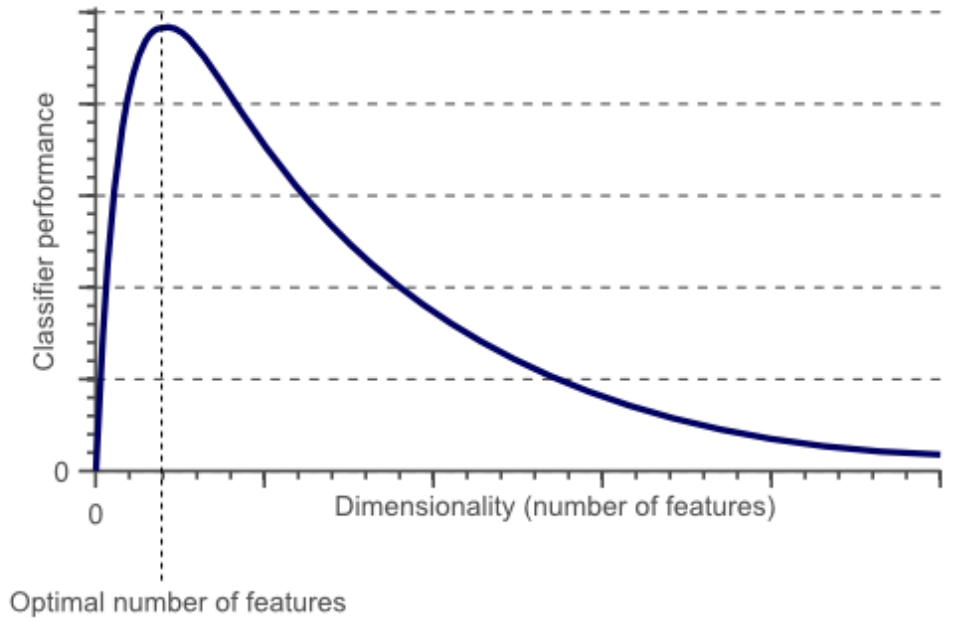
hamming also operates over discrete numerical vectors.

High Dimensional Space

- High dimensional space is beyond our intuition
- Do not take things for granted in high dimensional space
- Curse of dimensionality

$$\lim_{d\to\infty} E\left(\frac{\text{dist}_{\max}(d) - \text{dist}_{\min}(d)}{\text{dist}_{\min}(d)}\right) \to 0$$

Beyer et al., 1997



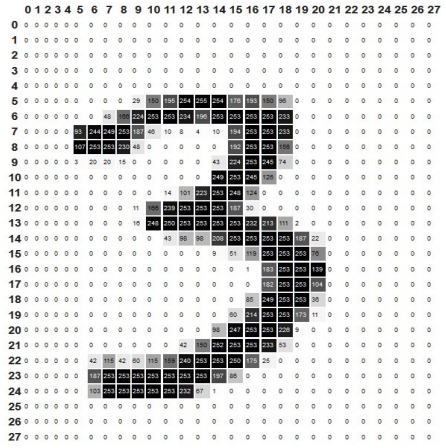
https://www.visiondummy.com/2014/04/curse-dimensionality-affect-classification/

Manifold Hypothesis

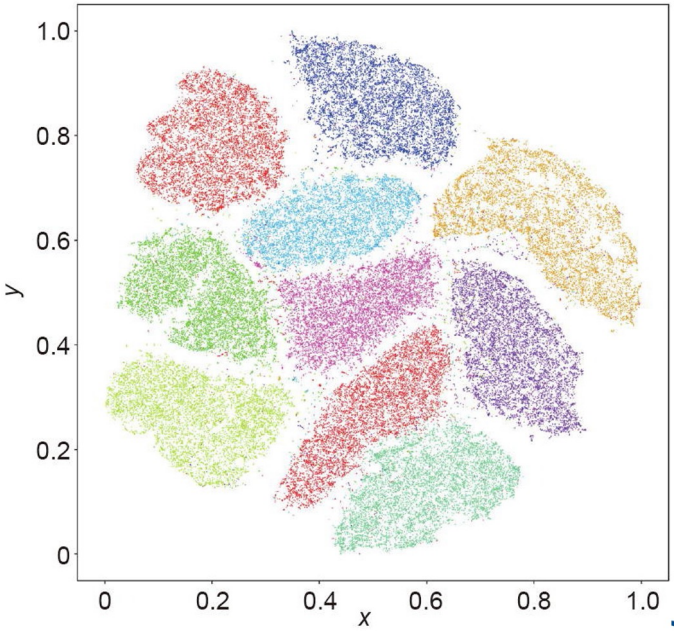
Real world high-dimensional data lie on low-dimensional manifolds embedded within the high-dimensional space.

MNIST dataset

Each handwritten digit image is 28 by 28 pixels, leading to a space of 784 dimensions



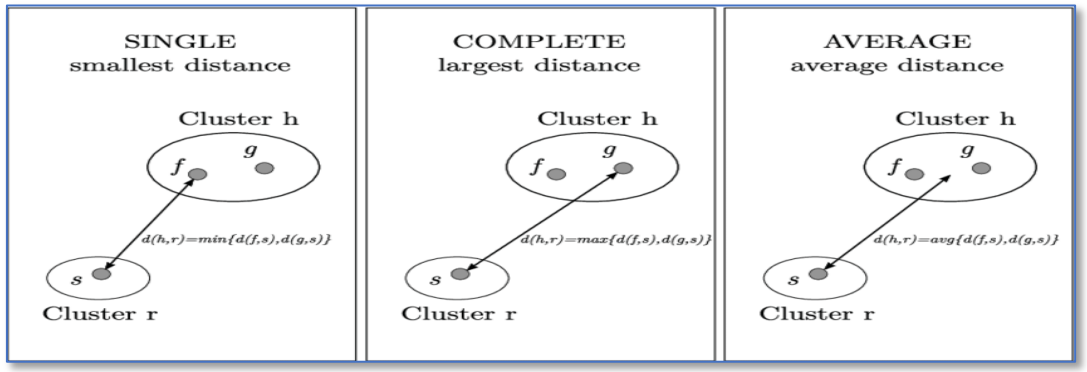
TSNE embedding



https://www.sciencedirect.com/science/article/pii/S2095809919302279#b0015

Hierarchical Clustering

- Hierarchical clustering: method seeking to build a hierarchy of clusters.
- Two strategies:
  - Divisive – top down approach
  - Agglomerative – bottom up approach
- Distance: pairwise distance between the points
- Linkage: determines the distance between sets of points as a function of the pairwise distances between points.

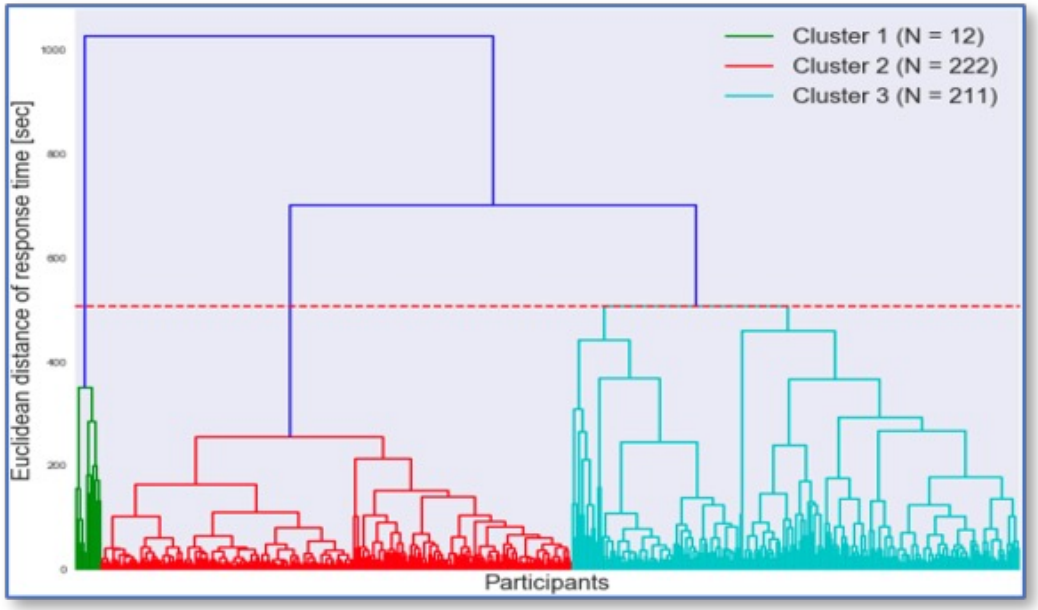


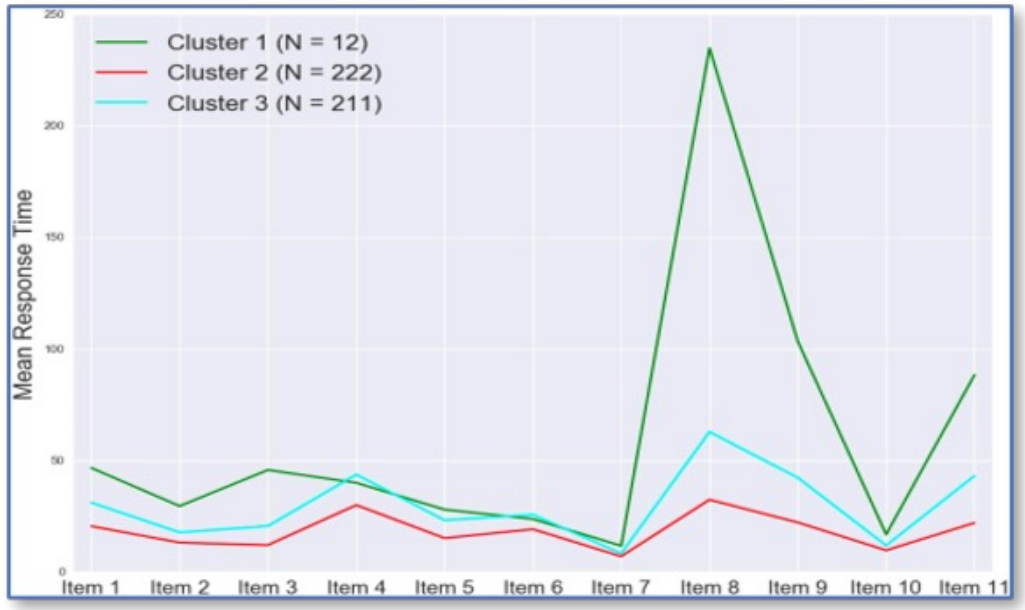
- Ward's minimum variance criterion: minimizes the total within-cluster variance.

https://en.wikipedia.org/wiki/Hierarchical_clustering

Example Application – Student Clustering

- Based on student's response time to different items, cluster them into different groups (response style)





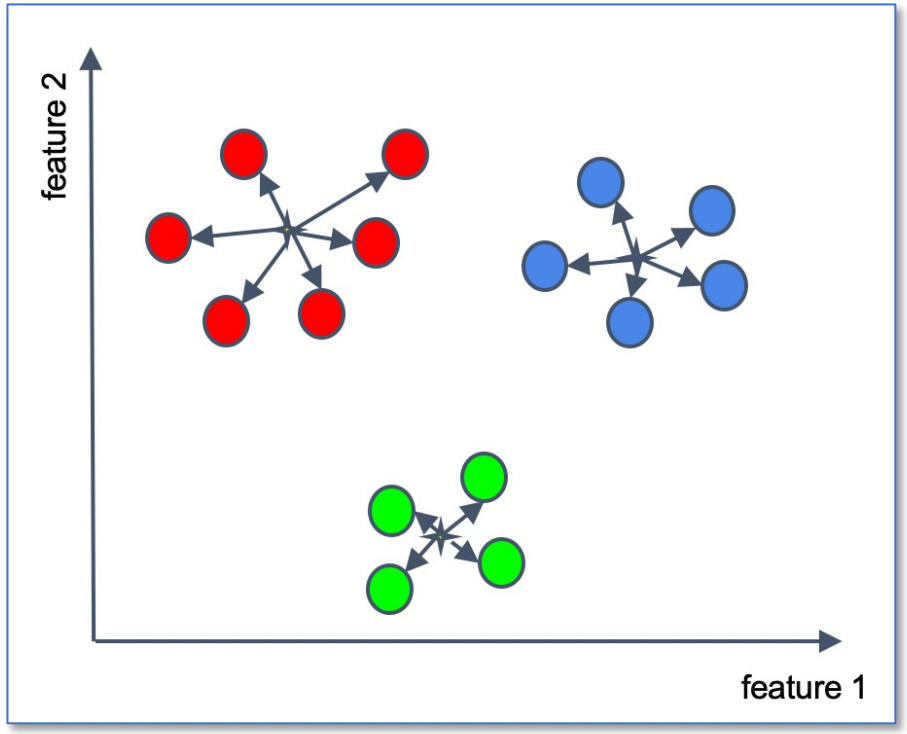
Lee, Hao, et al., 2019

K-means

- K-means (NP hard computation): divides a set of N samples X into K disjoint clusters C, each described by the mean μj of the samples in the cluster. The means are commonly called the cluster “centroids”
- The K-means algorithm aims to choose centroids that minimize the inertia, or within-cluster sum-of-squares criterion:

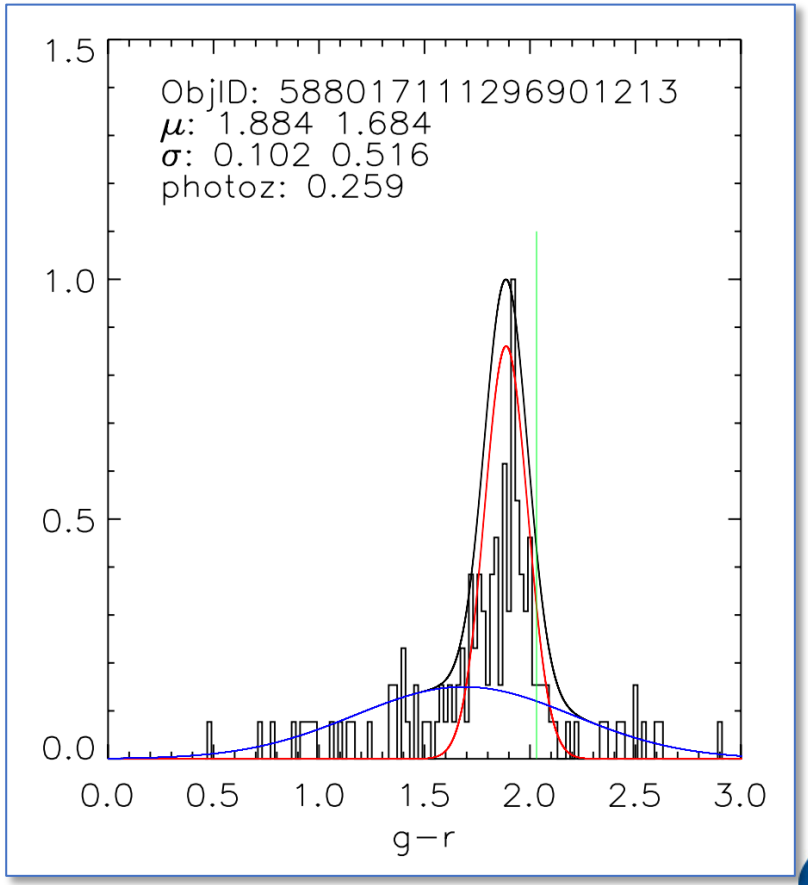
$$\sum_{i=0}^{n} \min_{\mu_j \in C} (||x_i - \mu_j||^2)$$

- Need to specify the number of clusters before running the algorithm



Model-based clustering

- Gaussian mixture model: the underlying distribution is approximated as a weighted sum of gaussian distributions
- Using BIC/AIC to determine the number of mixtures



THE ASTROPHYSICAL JOURNAL
SUPPLEMENT SERIES

A GMBCG GALAXY CLUSTER CATALOG OF 55,424 RICH CLUSTERS FROM SDSS DR7

Jiangang Hao[1], Timothy A. McKay[2,3], Benjamin P. Koester[4], Eli S. Rykoff[12,5,6], Eduardo Rozo[13,7], James Annis[1], Risa H. Wechsler[8,9], August Evrard[2,3], Seth R. Siegel[2], Matthew Becker[10]
+ Show full author list

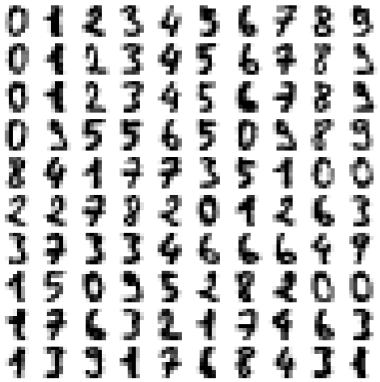
Published 2010 November 23 • © 2010. The American Astronomical Society. All rights reserved.
The Astrophysical Journal Supplement Series, Volume 191, Number 2
Citation Jiangang Hao *et al* 2010 *ApJS* **191** 254

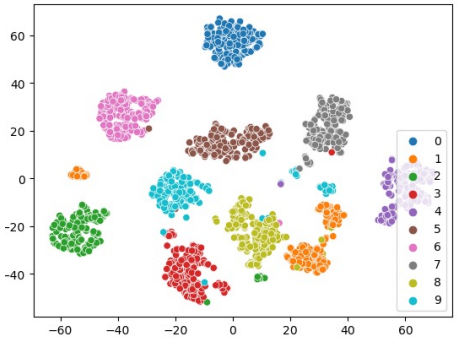
Number of Clusters

- A tricky question
- Overall, clusters should maximize between cluster distance and minimize within cluster distance
- Statistical measures: function(Data, Cluster_label)
  - Calinski-Harabasz index: ratio of the sum of between-cluster dispersion and of within-cluster dispersion. The bigger the better.
  - Silhouette score: measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). Best value is 1 and worst is -1.
- Hierarchical clustering
  - Visual check
- K-means
  - Inertia: within-cluster sum of square $\sum_{i=0}^{n} \min_{\mu_j \in C}(||x_i - \mu_j||^2)$

Dimensionality Reduction – t-SNE

- Principal Component Analysis (PCA) does not address non-linear variance
- t-SNE: t-distributed stochastic neighbor embedding
  - N data points x_1, x_2, …, x_n in high dimensional space
  - Look for a low dimensional representation y_1, y_2, …, y_n of the N data points
  - Such that $p_{ij}(X) \sim q_{ij}(Y)$, where p_{ij} and q_{ij} are distribution that characterize how close the data points are in the corresponding space.





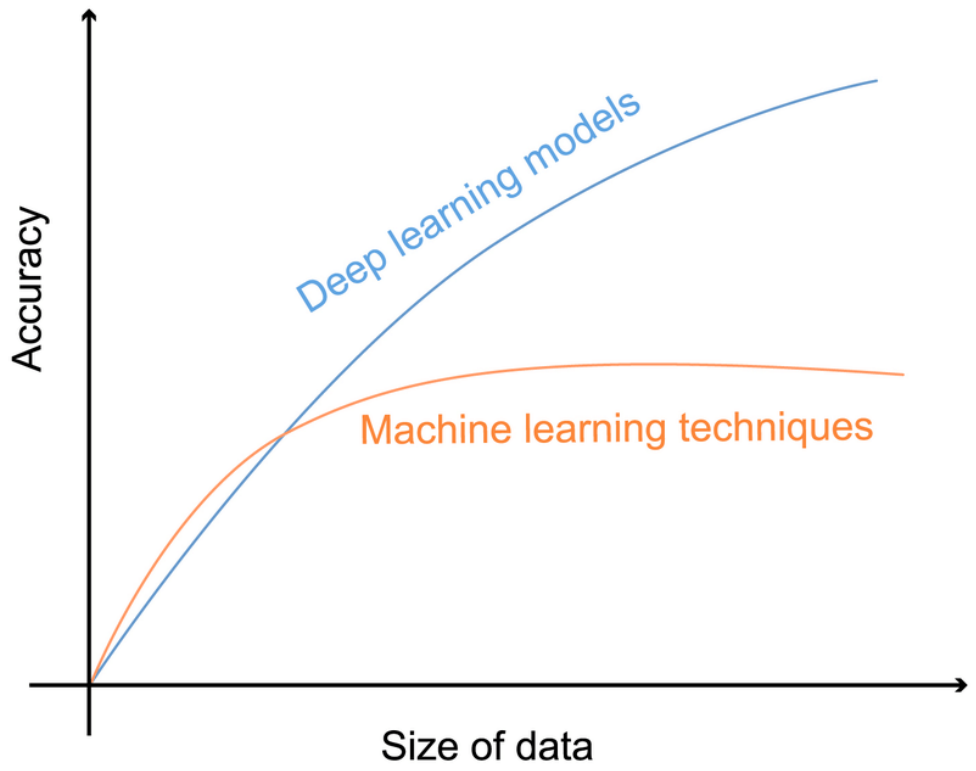
$$p_{ij} = \frac{\exp\left(-\|x_i - x_j\|^2 / 2\sigma^2\right)}{\sum_{k \neq l} \exp\left(-\|x_k - x_l\|^2 / 2\sigma^2\right)} \qquad q_{ij} = \frac{\left(1 + \|y_i - y_j\|^2\right)^{-1}}{\sum_{k \neq l} \left(1 + \|y_k - y_l\|^2\right)^{-1}} \qquad C = KL(P||Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

t-SNE Hyperparameters

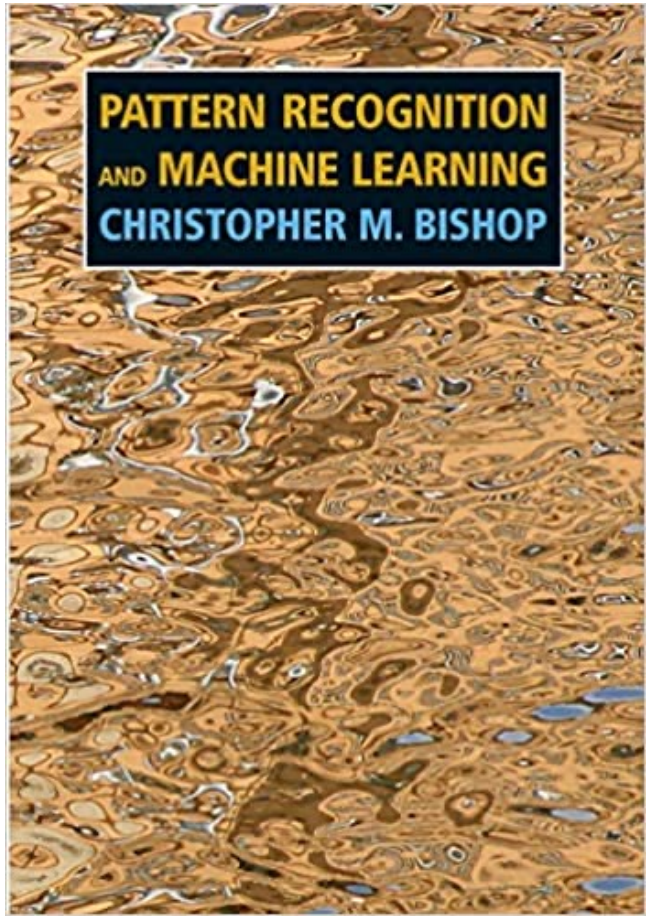
- Perplexity: related to the number of nearest neighbors that are used in other manifold learning algorithms. Larger datasets usually required a larger perplexity
  - Usually set between 5 and 50
- Early exaggeration: controls how tight natural clusters in the original space will be in the embedded space, and how much space will be between them
  - Default = 12 in sklearn
- Parameter playground: https://distill.pub/2016/misread-tsne/

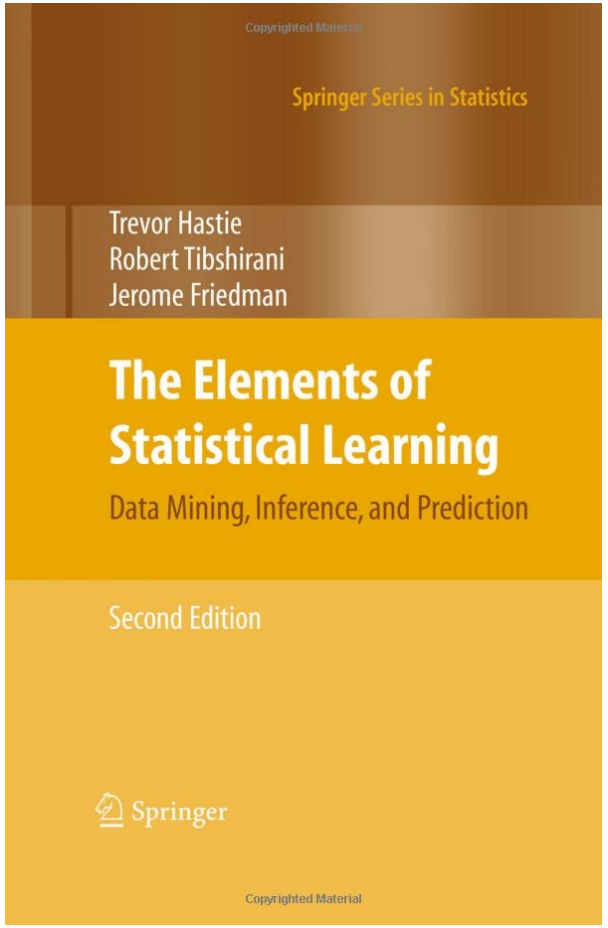
Summary

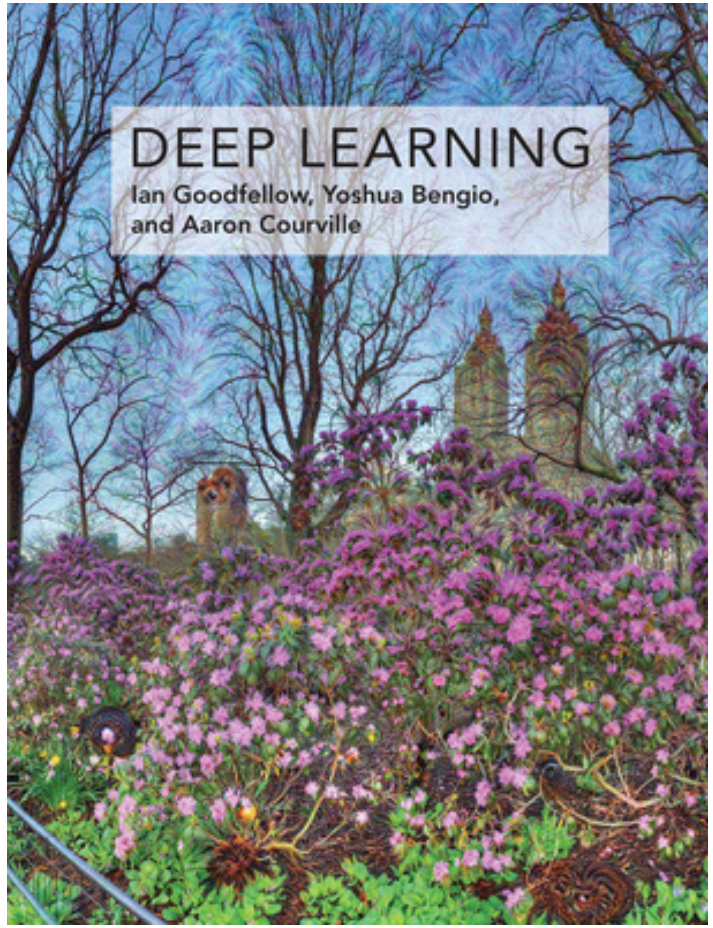
- Machine learning is a very active research field
- We discussed the so-called traditional machine learning methods and did not cover neural network and deep learning
- Deep learning refers to neural network models with multiple layers and its accuracy increases as size of data increases.
- The more you learn, the less you “know”



Recommended Books

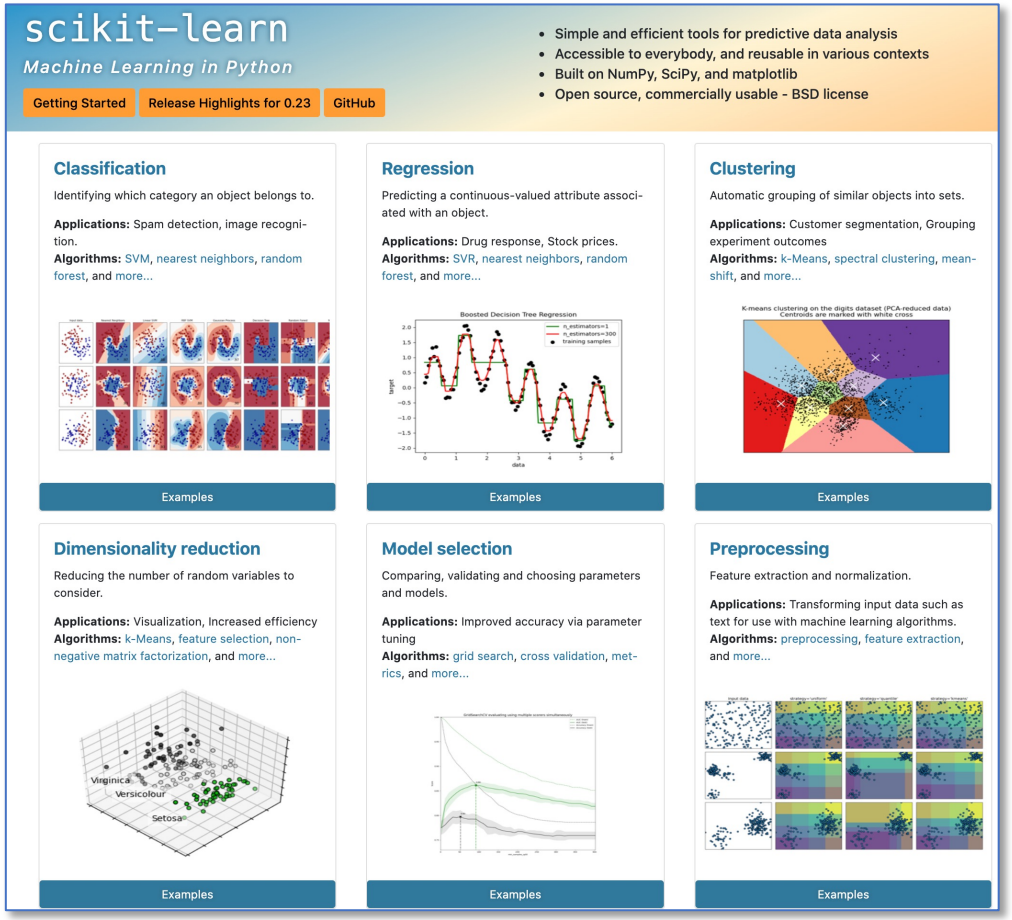




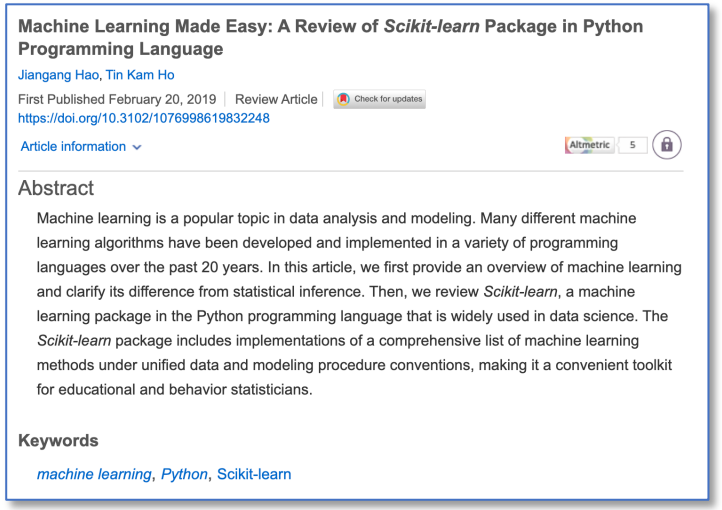


Software and Packages

Scikit-learn



One of the main open-source ML package in Python



Machine Learning Made Easy: A Review of *Scikit-learn* Package in Python Programming Language

Jiangang Hao, Tin Kam Ho

First Published February 20, 2019 | Review Article | Check for updates

https://doi.org/10.3102/1076998619832248

Article information

Altmetric 5

Abstract

Machine learning is a popular topic in data analysis and modeling. Many different machine learning algorithms have been developed and implemented in a variety of programming languages over the past 20 years. In this article, we first provide an overview of machine learning and clarify its difference from statistical inference. Then, we review *Scikit-learn*, a machine learning package in the Python programming language that is widely used in data science. The *Scikit-learn* package includes implementations of a comprehensive list of machine learning methods under unified data and modeling procedure conventions, making it a convenient toolkit for educational and behavior statisticians.

Keywords

machine learning, *Python*, Scikit-learn

```
model = classifier(hyperparameters = something)
model.fit(X_train, y_train)
y_test = model.predict(X_test)
```

Setup Environment – from Anaconda to Mamba

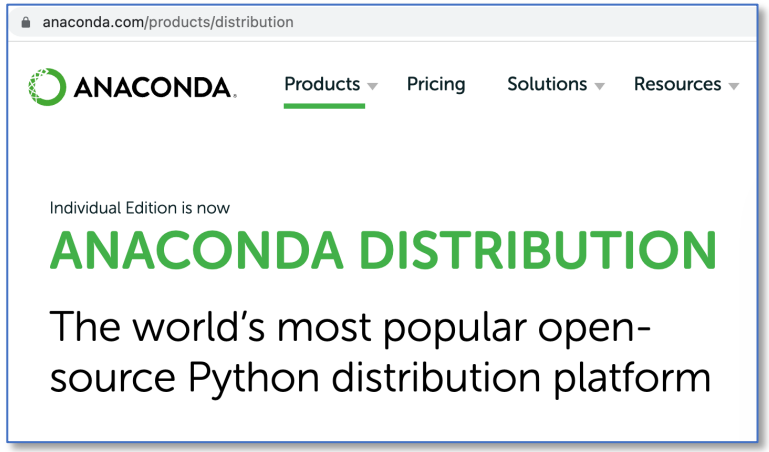
- **Step 1: Install mamba**
  - Go to https://github.com/conda-forge/miniforge to download the "Mambaforge" (NOT the other ones). After downloading, install it into your system.
- **Step 2: Create working virtual environment**
  - you want to create an environment for python 3.9, you can do:

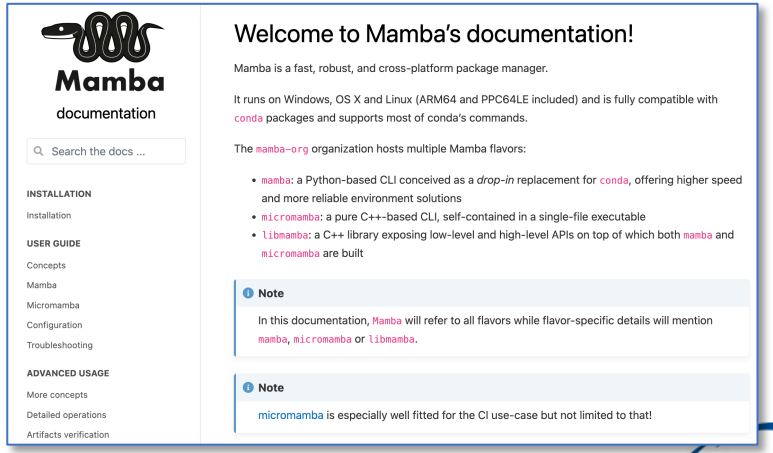
    ```
    mamba create --name py39 python=3.9
    ```
  - After the virtual environment is created, you can activate/deactivate it as:

    ```
    mamba activate py39 mamba deactivate
    ```
  - If you want to install python packages into your newly created environment, you can do (after you are in that environment), for example,

    ```
    mamba install numpy scipy pandas jupyterlab seaborn
    ```
  - After you installed jupyterlab, you can start it by typing

    ```
    jupyter lab
    ```



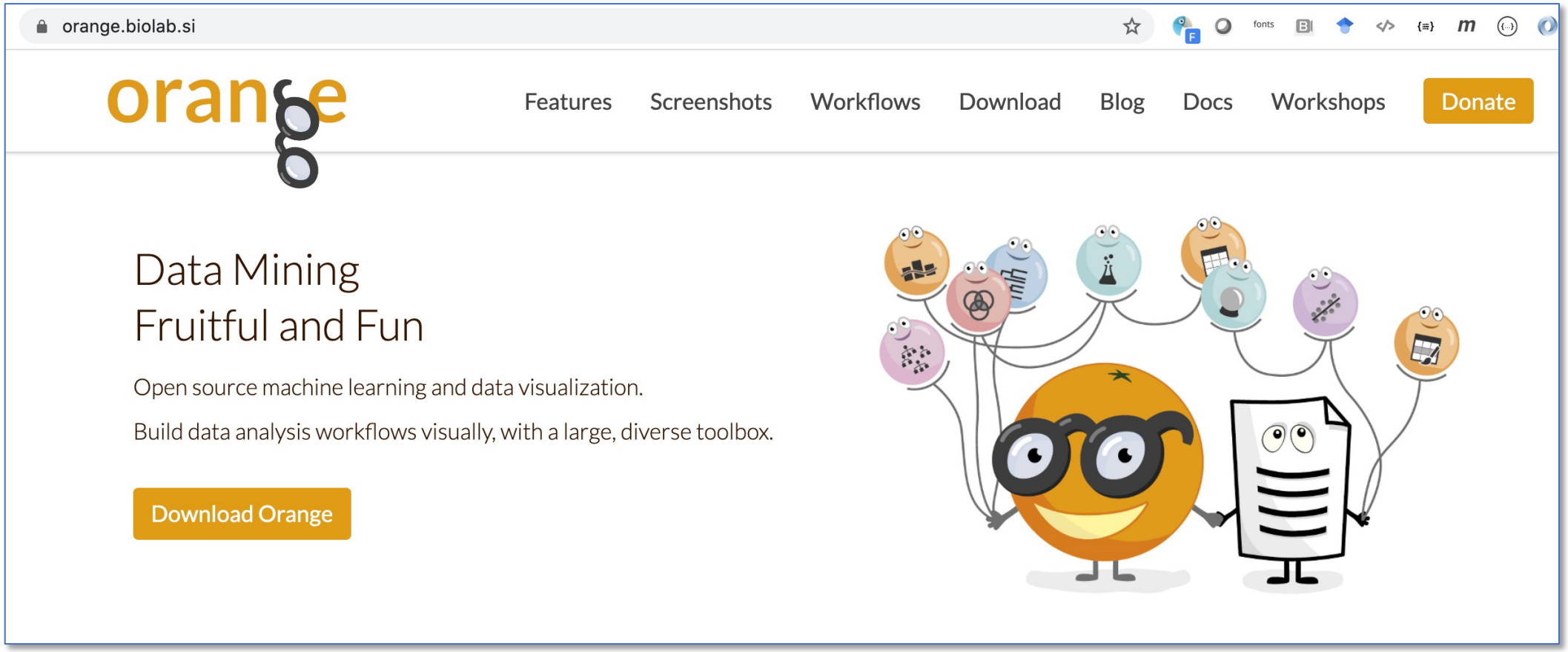


Scikit-learn PCA Example

- Iris Data Set: 150 instances x 4 features (Fisher, 1950)
  - sepal length [cm]
  - sepal width [cm]
  - petal length [cm]
  - petal width [cm]
  - Class: Iris Setosa, Iris Versicolour, Iris Virginica

Notebook demo

Orange


orange.biolab.si
orange
Features
Screenshots
Workflows
Download
Blog
Docs
Workshops
Donate
Data Mining
Fruitful and Fun
Open source machine learning and data visualization.
Build data analysis workflows visually, with a large, diverse toolbox.
Download Orange

Natural Language Processing Basics: Text Mining and Automated Scoring

- Language models
- Text representation and mining
- Automated scoring
- Deep learning models

Chapter 14
Text Mining and Automated Scoring

Michael Flor and Jiangang Hao

Abstract Natural Language Processing (NLP) is playing an increasingly important role in learning and assessments. Some typical applications of NLP in education include automated scoring, automated item generation, conversation-based assessments, writing assistants, text mining for education, and so on. In this chapter, we aim at introducing some basics of NLP through two typical applications in educational contexts, text mining and automated scoring. We hope readers can get an overall picture of NLP and get familiarized with some basic tools for handling natural language data, which may serve as stepping stones for their future work with NLP.

14.1 Overview

Natural Language Processing, also known as Computational Linguistics, is an interdisciplinary area of research, spanning computer science and artificial intelligence (AI), linguistics and cognitive psychology, as well as other disciplines, such as mathematics, logic, philosophy, and neuroscience. The central questions of research revolve around the notion of enabling computer programs to automatically analyze and represent natural human language (e.g. English). Sub-areas of inquiry include natural language understanding and natural language generation (for text-based data), and speech recognition and generation (for spoken data). As an area of study, computational linguistics has both theoretical and applied perspectives, and the notion of NLP often refers to the more applied areas of the field.

The R or Python codes can be found at the GitHub repository of this book: https://github.com/jgbrainstorm/computational_psychometrics

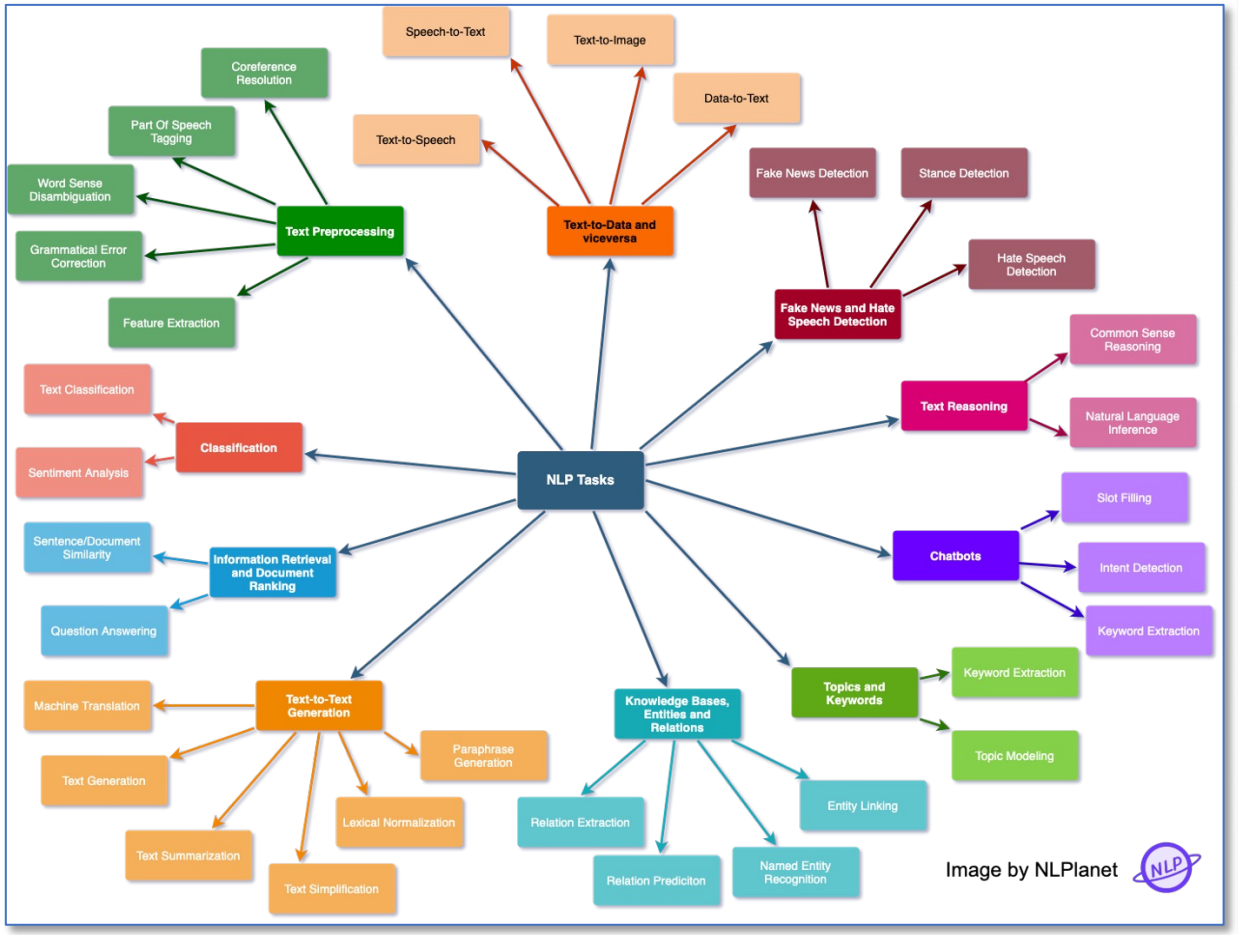
M. Flor (✉) · J. Hao
Educational Testing Service, Princeton, NJ, USA
e-mail: mflor@ets.org; jhao@ets.org

The limits of my language are the limits of my mind. All I know is what I have words for.

-Ludwig Wittgenstein

NLP Basics

- Natural Language Processing, also known as computational linguistics, focuses on enabling computer programs to automatically analyze and represent natural human language
- NLP tasks
- NLP in educational assessment
  - Text mining
  - Conversational assessment
  - Automated scoring
  - Question/item generation
  - …



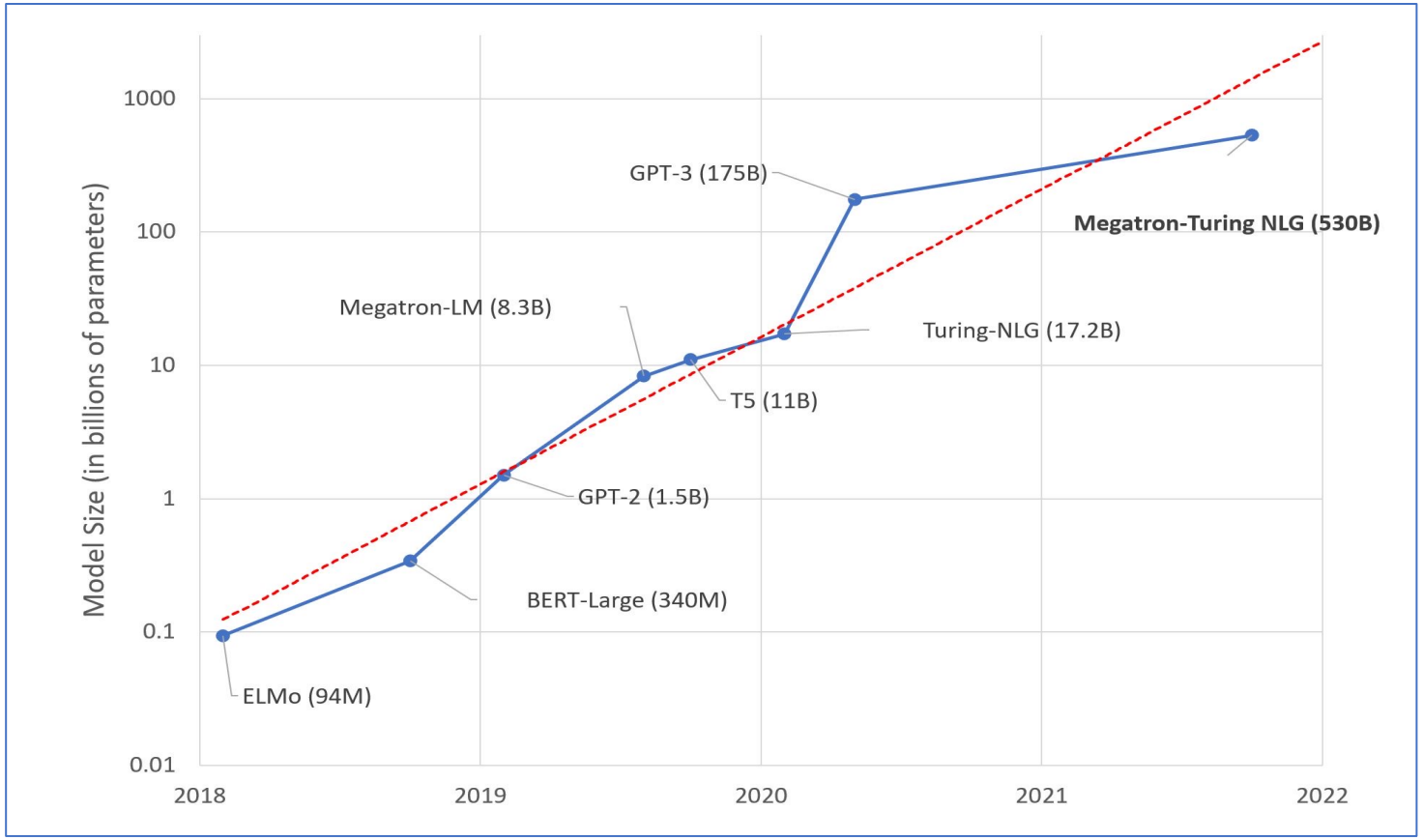
https://medium.com/nlplanet/two-minutes-nlp-33-important-nlp-tasks-explained-31e2caad2b1b

Language Model

- Language model specify the probability of a given sequence of words
  - $P(w_1)$
  - $P(w_1,w_2) = P(w_1|w_2)P(w_2)$
  - $P(w_1,w_2, w_3) = P(w_1|w_2,w_3)P(w_2|w_3)P(w_3)$
  - All we need to know is $P(w_t|w_1,w_2,\ldots, w_{t-1})$ and $P(w)$
- N-gram model: https://web.stanford.edu/~jurafsky/slp3/3.pdf
  - Bi-gram model: $P(w_t|w_1, w_2, \ldots, w_{t-1}) = P(w_t|w_{t-1})$ – Markov assumption
  - N-gram model: $P(w_t|w_{t-n+1}, \ldots, w_{t-1})$
  - Problems: the computation scales exponentially with N; Sparse
- Neural language models – good for large corpus

https://web.stanford.edu/~jurafsky/slp3/7.pdf

Large Language Models - LLM



https://huggingface.co/blog/large-language-models

https://clockwise.software/blog/what-is-gpt-3/

Text Representation

- Convert text into numerical representations
- Preprocessing – clean the text
- Bag of words
  - Counts of unique words
- N-gram
  - Counts of n consecutive words
- Word embedding
  - Each word is mapped into a vector of smaller dimension
  - Latent semantic analysis - LSA
  - Neural embedding, e.g., Word2vec

Preprocessing

- Tokenization
- Case normalization
- Stop words removal
- Typo correction
- POS tagging
- Stemming and lemmatization

```
CC  coordinating conjunction
CD  cardinal digit
DT  determiner
EX  existential there (like: "there is" ... think of it like "there exists")
FW  foreign word
IN  preposition/subordinating conjunction
JJ  adjective   'big'
JJR adjective, comparative  'bigger'
JJS adjective, superlative  'biggest'
LS  list marker 1)
MD  modal   could, will
NN  noun, singular 'desk'
NNS noun plural 'desks'
NNP proper noun, singular   'Harrison'
NNPS    proper noun, plural 'Americans'
PDT predeterminer   'all the kids'
POS possessive ending   parent\'s
PRP personal pronoun    I, he, she
PRP$    possessive pronoun  my, his, hers
RB  adverb  very, silently,
RBR adverb, comparative better
RBS adverb, superlative best
RP  particle    give up
TO  to  go 'to' the store.
UH  interjection    errrrrrrrm
VB  verb, base form take
VBD verb, past tense    took
VBG verb, gerund/present participle taking
VBN verb, past participle   taken
VBP verb, sing. present, non-3d take
VBZ verb, 3rd person sing. present  takes
WDT wh-determiner   which
WP  wh-pronoun  who, what
WP$ possessive wh-pronoun   whose
WRB wh-abverb   where, when
```

stemming

Form	Suffix	Stem
studies	-es	studi
studying	-ing	study
niñas	-as	niñ
niñez	-ez	niñ

lemmatization

Form	Morphological information	Lemma
studies	Third person, singular number, present tense of the verb study	study
studying	Gerund of the verb study	study
niñas	Feminine gender, plural number of the noun niño	niño
niñez	Singular number of the noun niñez	niñez

Bag of Words - Unigram

Document-Term Matrix

	Term 1	*Term 2*	...	*Term t*
Document 1	2	1	...	...
Document 2	5	1	...	...
Document 3	6	1	...	...
...	...	...	...	
Document d	1	2	...	

- The cell numbers are the counts of the term in the document
- The matrix is sparse

	about	bird	heard	is	the	word	you
About the bird, the bird, bird bird bird	1	5	0	0	2	0	0
You heard about the bird	1	1	1	0	1	0	1
The bird is the word	0	1	0	1	2	1	0

TF-IDF Transformation

- Limitation of the document-term matrix
  - Frequency of different terms depends on the length of the documents
  - Some words more likely to appear in more documents
- Term frequency weighting (TF)

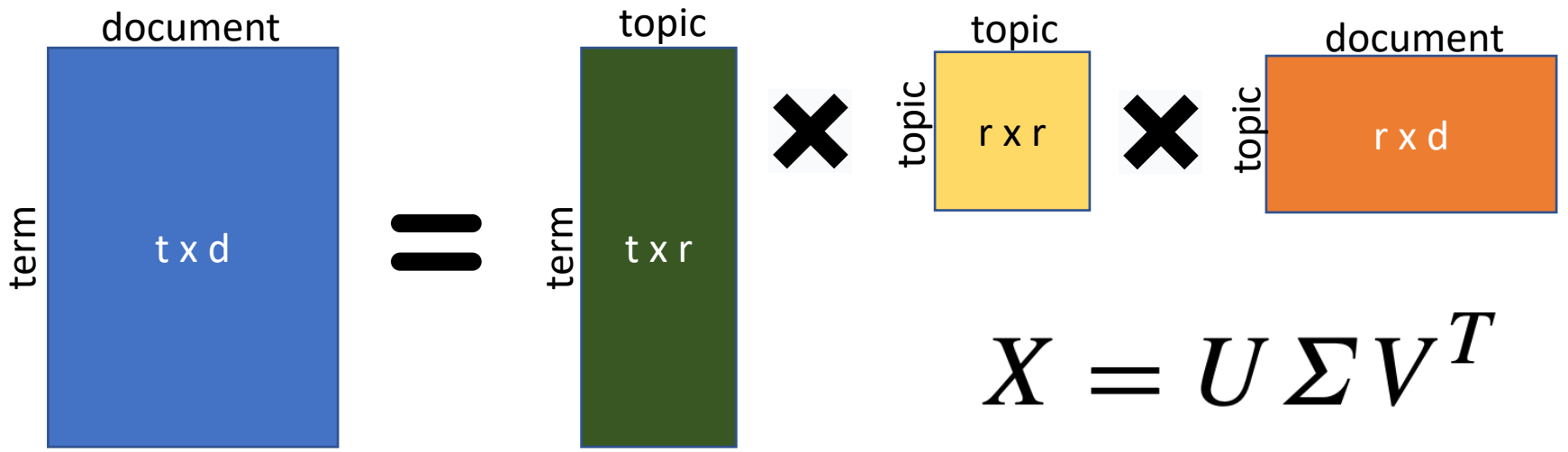
$$TF\left(\text{term}_j\right) = \frac{\text{Total frequency of term}_j \text{ in document}_i}{\text{Total frequency of all terms in document}_i}$$

- Inverse document frequency weighting (IDF)

$$IDF\left(\text{term}_j\right) = \log \frac{\text{Total number of documents}}{1 + \text{Number of documents containing term}_j}$$

Latent Semantic Analysis - LSA

Singular Value Decomposition



$$X = U \Sigma V^T$$

Where U is a t by r matrix and V is a d by r matrix with r is the rank of X. Both U and V have orthonormal columns, e.g., $UU^T = VV^T = I$. Σ is a r by r diagonal matrix of singular values that are non-negative and ordered in decreasing magnitude by convention.

Dimensional "Reduction"

Approximate the original term-document matrix

$$\hat{X} = U_k \Sigma_k V_k^T \xrightarrow{\text{Approximate}} X = U \Sigma V^T$$

SVD provides a way to find a lower rank matrix to approximate the term-document matrix. If we consider only the part of Σ that contains the largest k singular values (e.g., k topics in the LSA context), we have a $\hat{X} = U_k \Sigma_k V_k^T$, where U_k is a t by k matrix, Σ_k is a k by k matrix, and V_k is a d by k matrix. The U_k matrix contains information regarding how the k topics are related to the terms and the V_k matrix contains information on how the k topics are related to the documents, as

Vectors from LAS Embedding

Term (word) vectors

	Topic 1	Topic 2	...	Topic k
Term 1	...	...	...	...
Term 2	...	...	...	...
Term 3	...	...	...	...
...	...	...	...	...
Term t	...	...	...	...

Document vectors

	Topic 1	Topic 2	...	Topic k
Document 1	...	...	...	...
Document 2	...	...	...	...
Document 3	...	...	...	...
...	...	...	...	...
Document t	...	...	...	...

Choosing the k: coherence score

$$\text{Coherence} = \sum \text{score}(w_i, w_j)$$

$$\text{score}_{\text{UMass}}(w_i, w_j) = \log \frac{D(w_i, w_j) + 1}{D(w_i)}$$

- $D(w_i)$ is the count of documents containing the word w_i
- $D(w_i,w_j)$ is the count of documents containing both words w_i and w_j

Neural Embedding

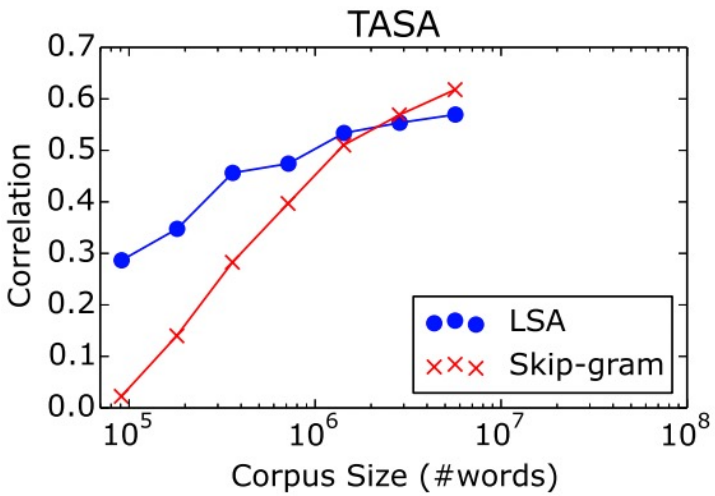
- Word2vec: Mikolov et al, 2013
  - Each word is represented by a vector of several hundred dimensions
  - Words that share common contexts in the corpus are located close to each other in the vector space – distributional hypothesis
  - Better than LSA for large language copra
- Two model architectures of word2vec
  - Continuous bag-of-words (CBOW): predict the target word from the window of surrounding context words
  - Continuous skip-gram: predict the surrounding window of context words using the target word

https://web.stanford.edu/~jurafsky/slp3/slides/6_Vector_Apr18_2021.pdf

Performance of Different Embedding

- For large corpus, neural embedding (word2vec) is generally doing better
- For small corpus, LSA embedding is generally doing better than word2vec
- For small corpus, ngram is also doing better than word2vec

Altszyler et al, 2017



Methods	N-gram		Document vector	
	Accuracy	Kappa	Accuracy	Kappa
MaxEnt	0.669	0.551	0.625	0.488
Random Forest	0.697	0.588	0.611	0.468
Linear SVC	0.723	0.623	0.619	0.479
Linear Chain CRF	0.736	0.641	0.643	0.515

Hao, et al, 2018 (unpublished)

Corpus dependent!

Automated Scoring

Numerical Features

Models

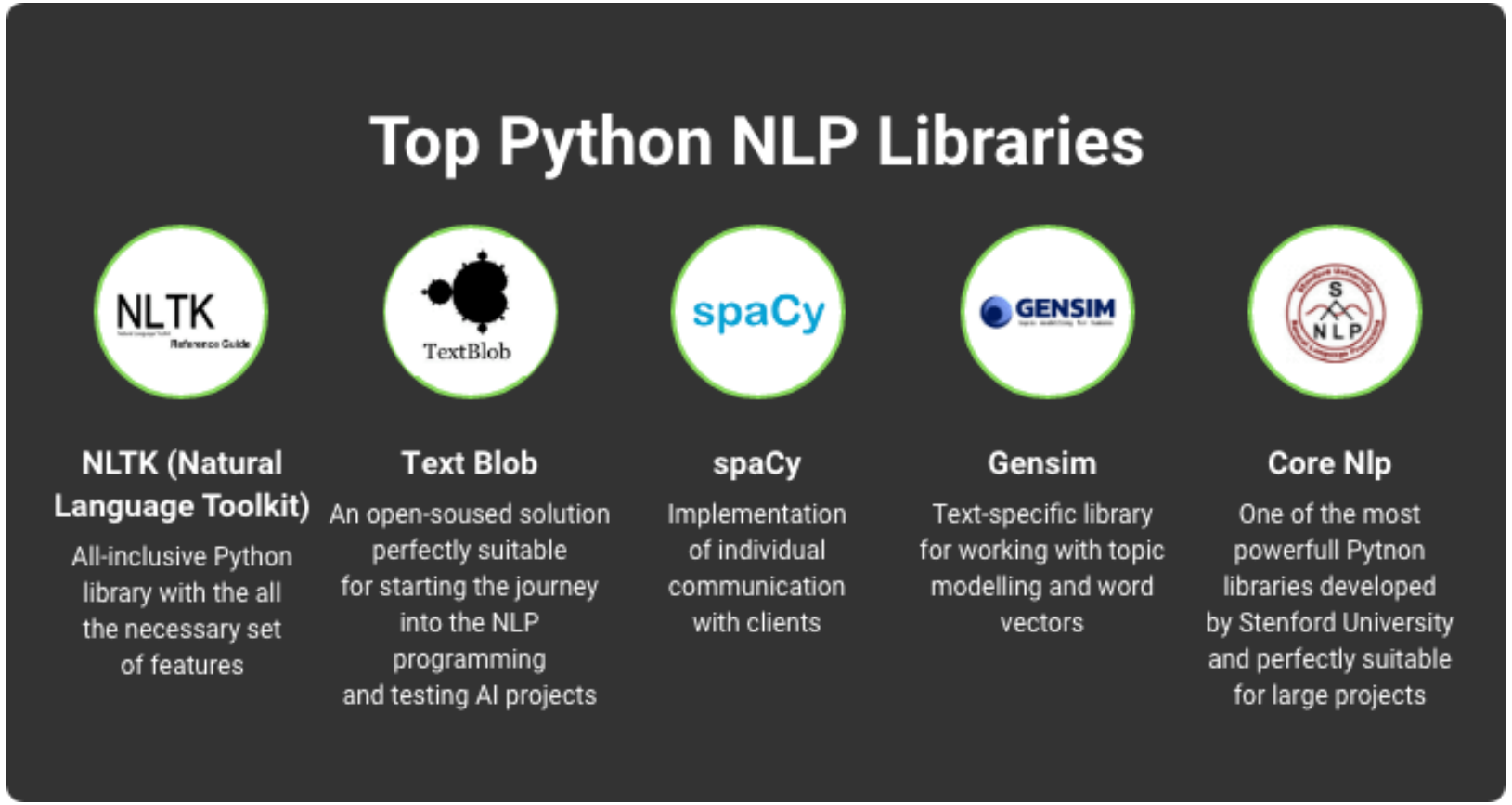
Scores

SAS vs. AES

- Short Answer Scoring (SAS)
  - Contextualized features
    - Ngram
    - Word/sentence vectors
  - Machine learning model to map the features to scores
  - Example: ETS CPS-rater (Hao, et al., 2017)
- Automated Essay Scoring (AES)
  - Generic features
    - E.g., Grammar, Usage, Mechanics, Style, Development, etc.
  - Contextualized features (on/off topic)
  - Statistical/Machine learning model to map the features to scores
  - Example: ETS e-rater (Burstein, 2003)

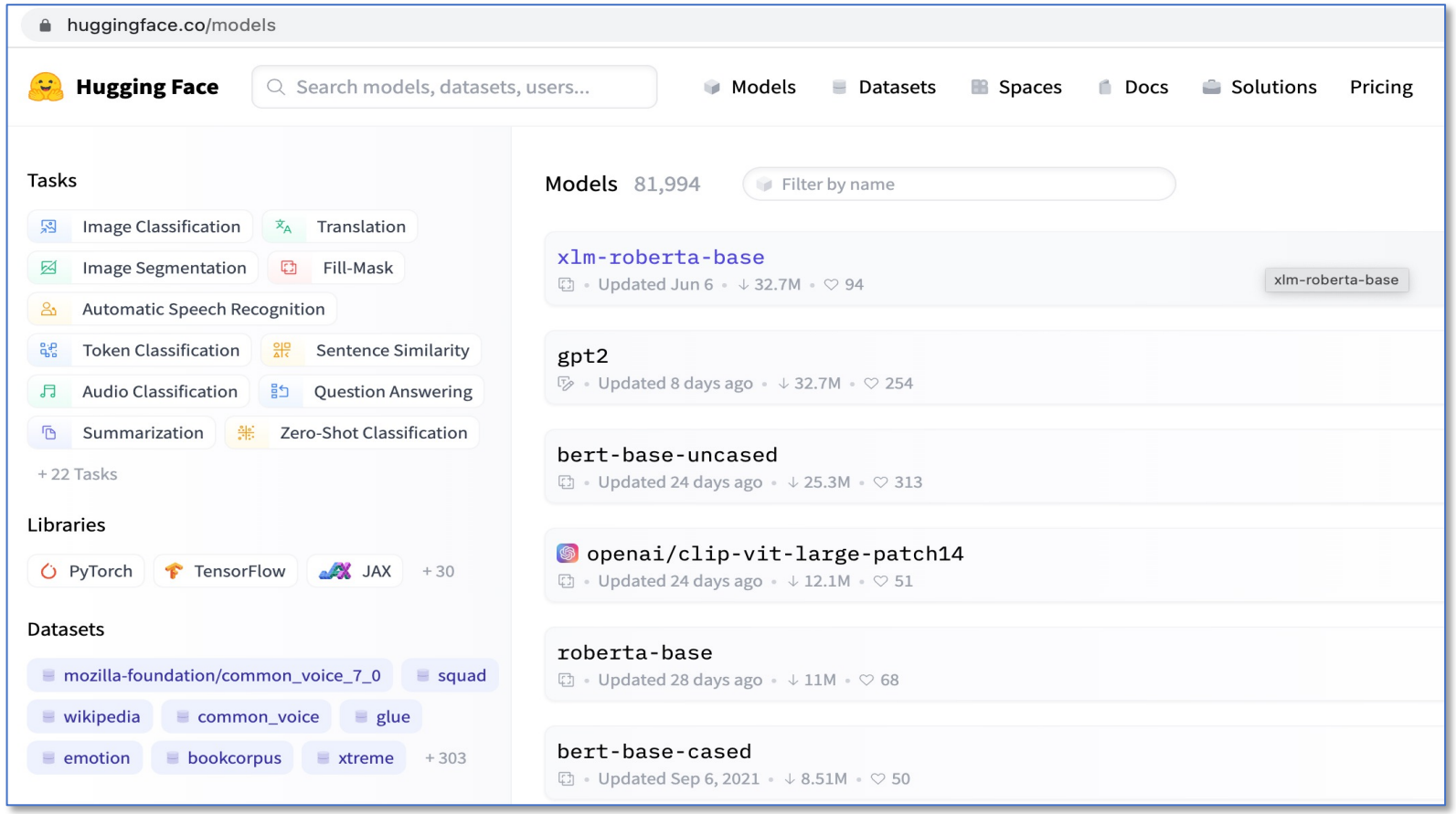
Code Demo

Software Packages



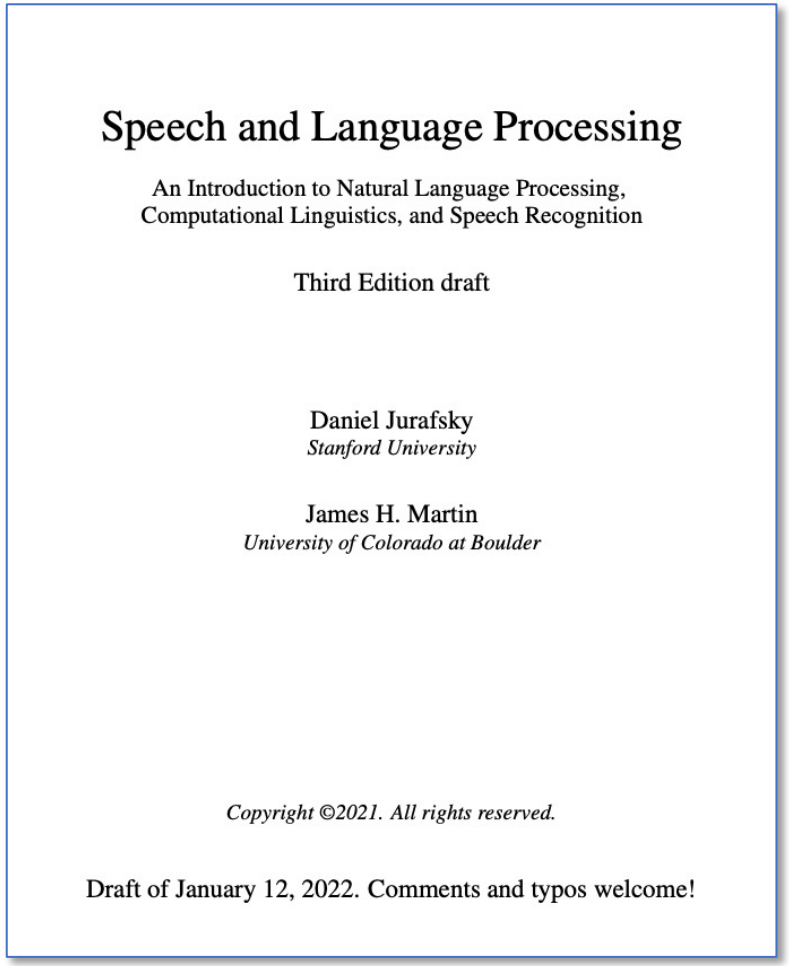
Many many others…

Deep Learning Models



Code Demo

Recommended Books



Speech and Language Processing

An Introduction to Natural Language Processing,
Computational Linguistics, and Speech Recognition

Third Edition draft

Daniel Jurafsky
Stanford University

James H. Martin
University of Colorado at Boulder

Draft of January 12, 2022. Comments and typos welcome!

Summary

- Digital technology provides tremendous possibilities for assessments
- New methods from data science, machine learning, NLP and other quantitative disciplines are needed to handle complex data from digital learning and assessments.
- Psychometric researchers need to acquire these new skills to meet the challenges from digital learning and assessment. You are playing critical roles in helping to build the future of the field.

Thank You!

jhao@ets.org

Self Study

Data Science

- Data Science basics
- Data wrangling and processing
- Interactive visualization and dashboarding

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 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

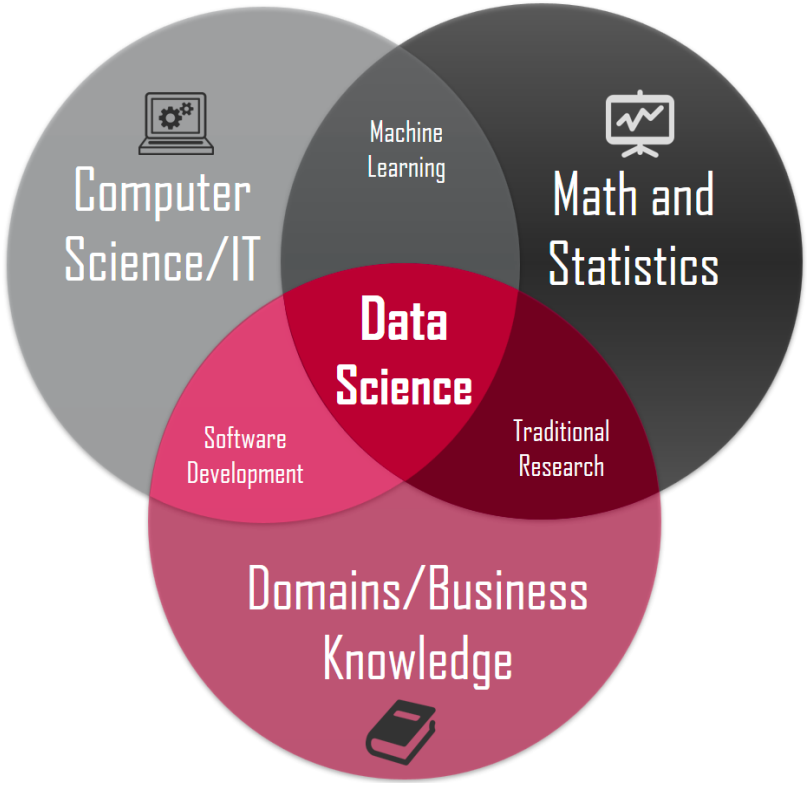
J. Hao (✉) · R. J. Mislevy
Educational Testing Service, Princeton, NJ, USA
e-mail: jhao@ets.org; rmislevy@umd.edu

Data Science Basics

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



Skills Needed for a Data Scientist

Quality as a scientist

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

Technical skills

- Programming in Python/R
- Computer/IT/DevOps
- Cloud computing for big data
- Data wrangling and visualization
- Statistics, Math
- Machine learning

Data Types and Data Storage

Data Types

- **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 **s**tructured 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.

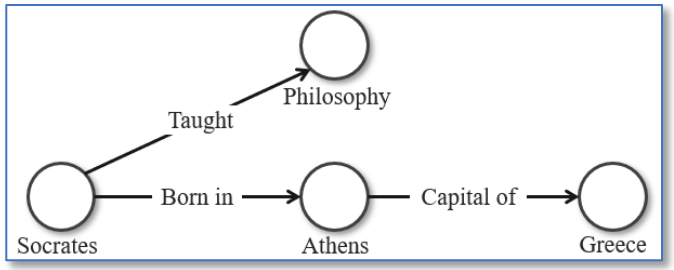
Design Data through Data Model

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.

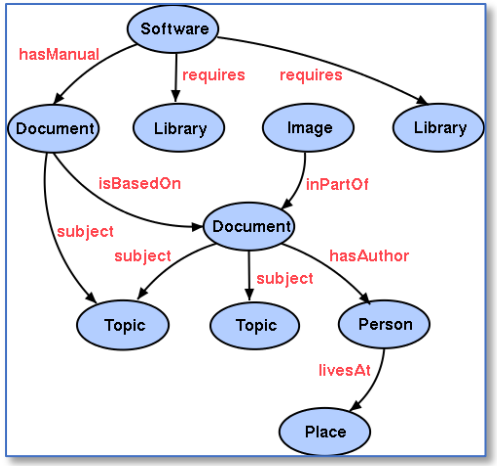
Triple Store and RDF

- **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)

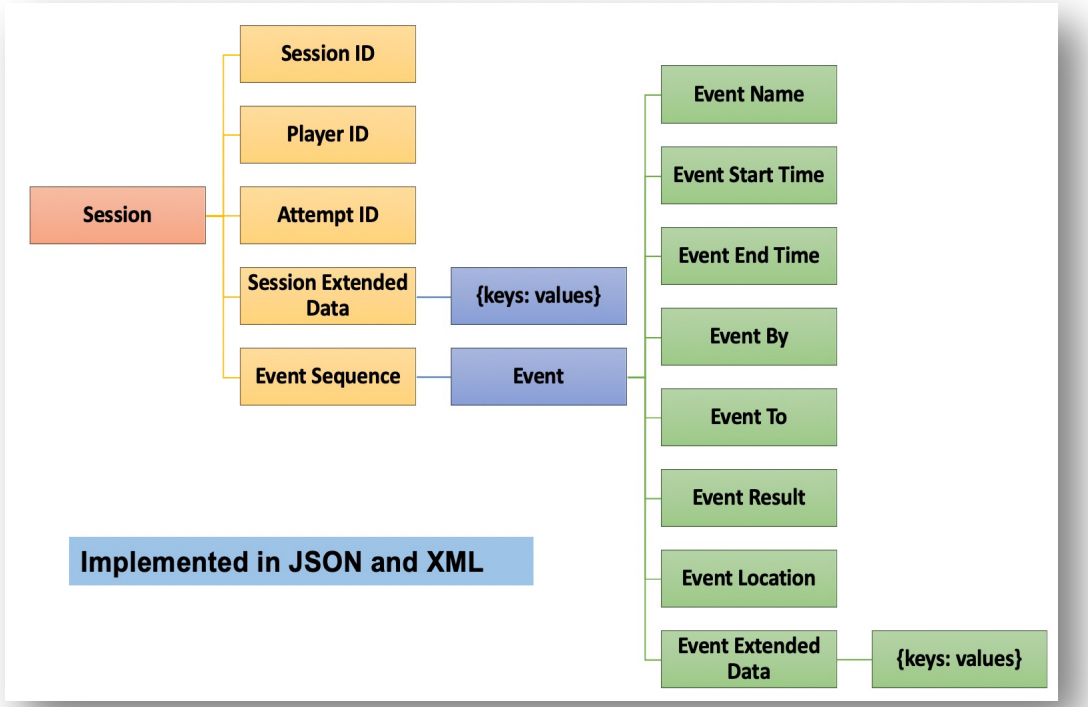


https://vodkhang.com/intelligent-system/semantic-technology-and-its-application

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



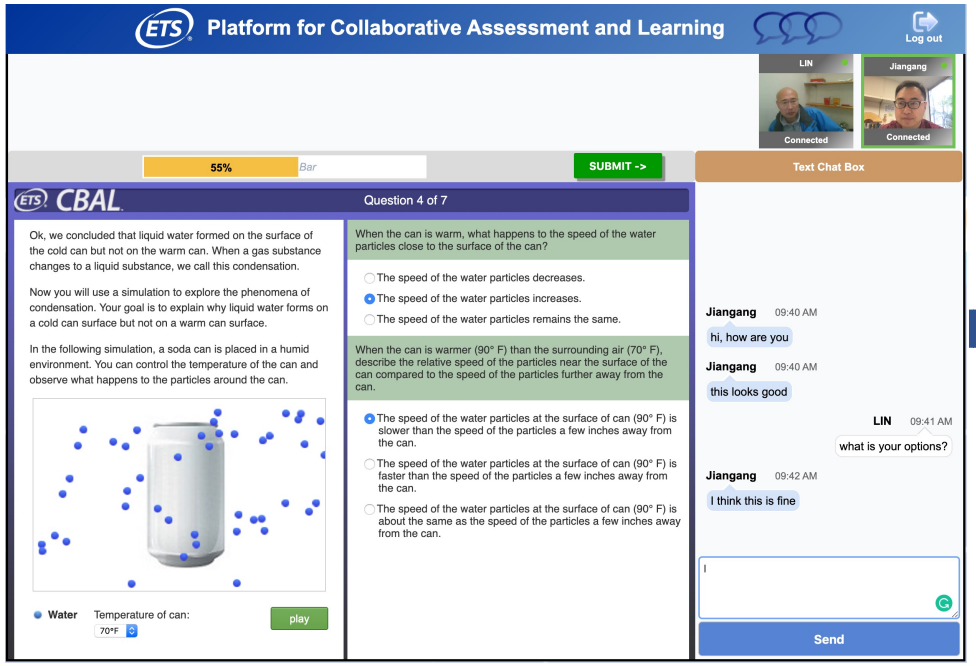
- Evidence Identification Centered Data Design - an extra process in Evidence Centered Design
- Log file -> Evidence Trace File

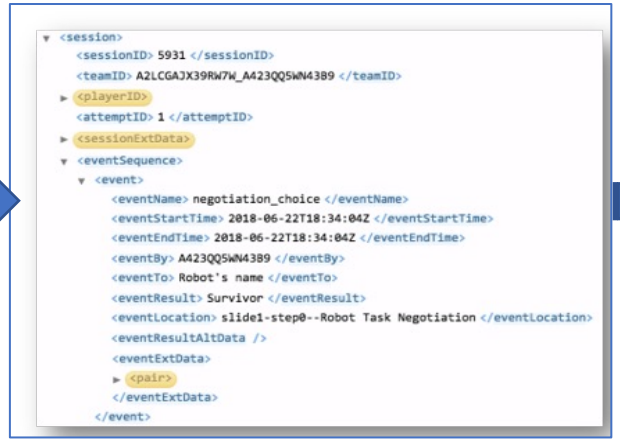
JSON Implementation

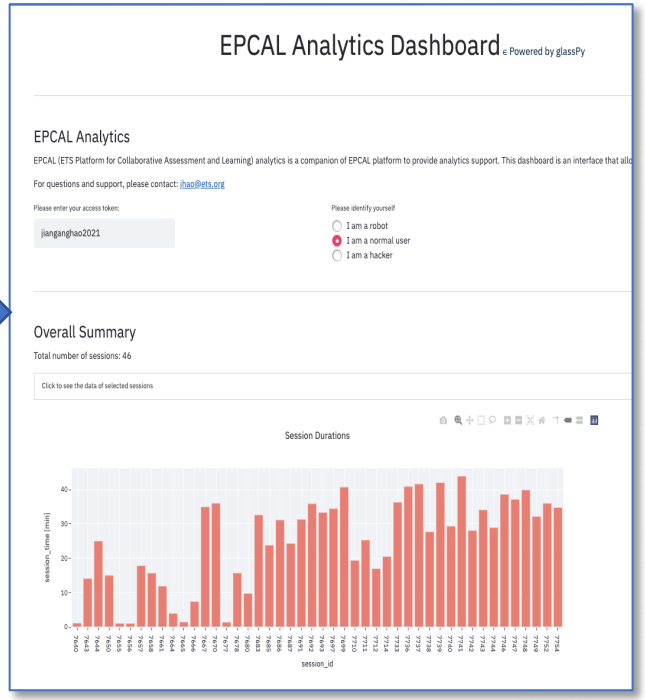
XML Implementation

A Full Implementation in EPCAL

Full implemented in the ETS Platform for Collaborative Assessment and Learning



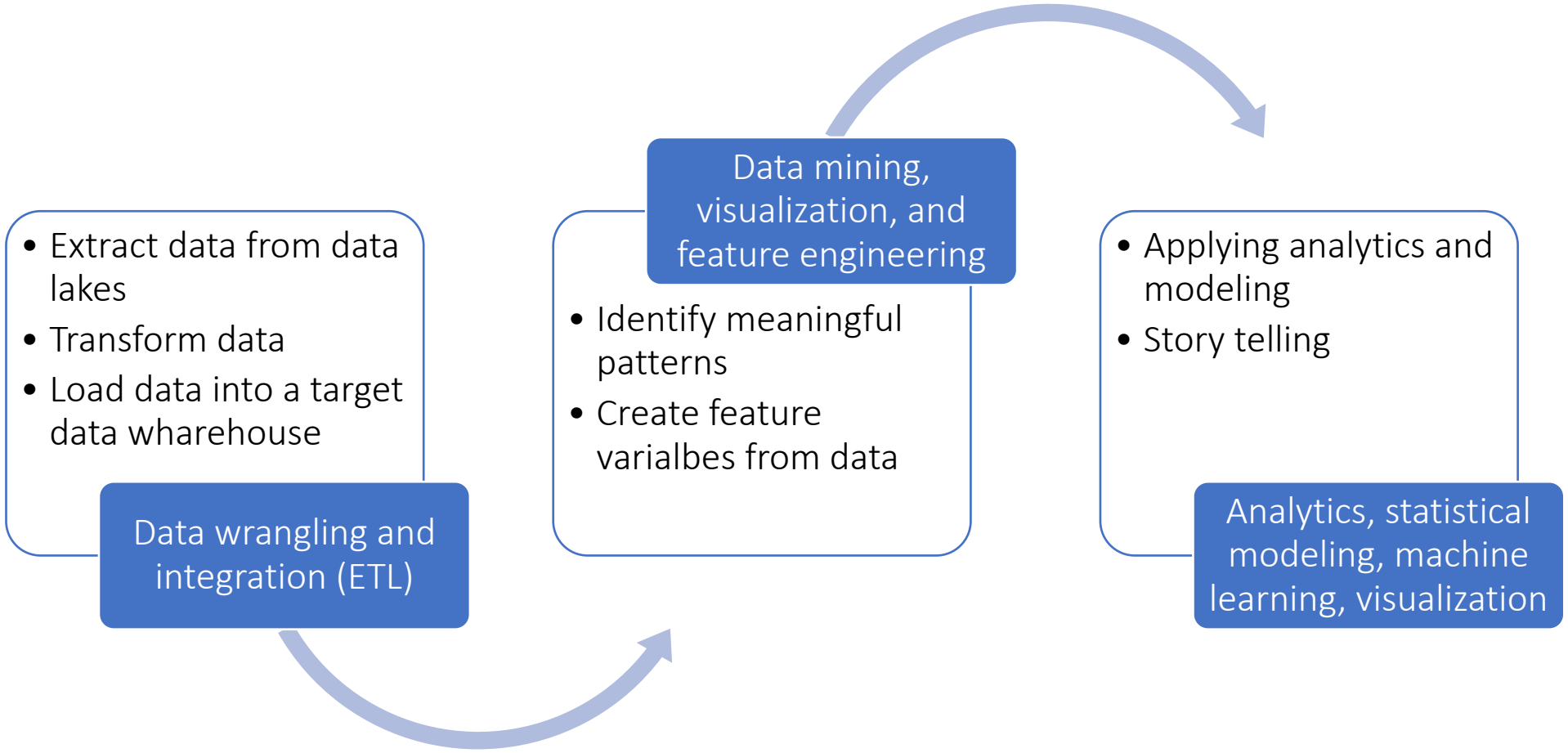




https://onlinelibrary.wiley.com/doi/full/10.1002/ets2.12181

Data Processing and Wrangling

General Steps of Data Processing



Types of Data Processing

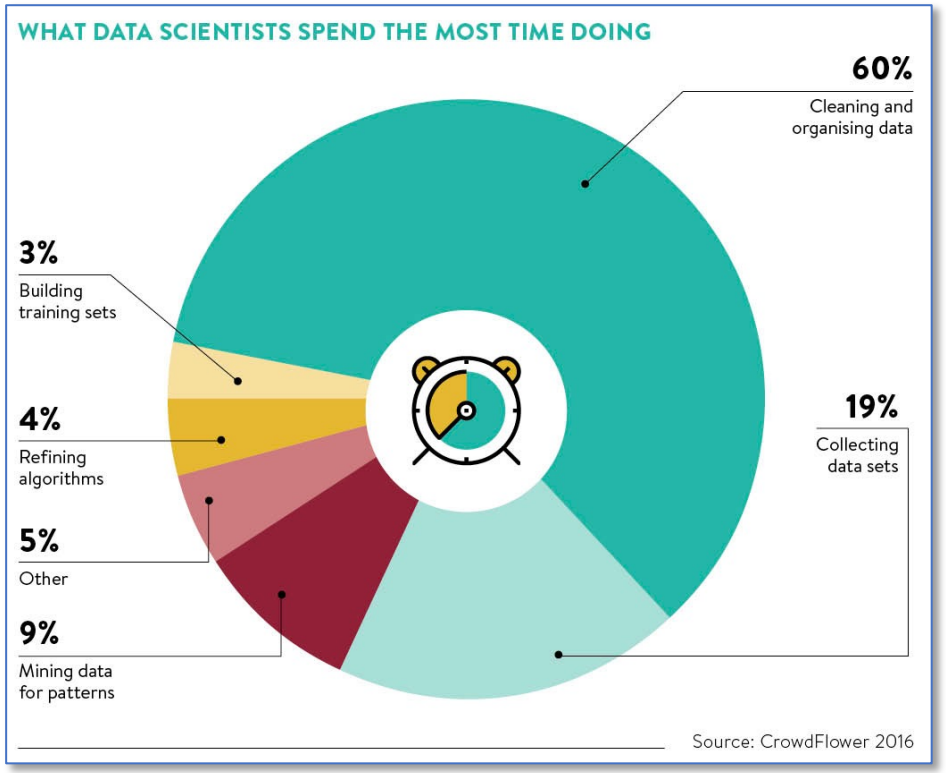
- Batch processing
  - Process data after the data are collected
  - E.g., Apache Hadoop, …
- Stream processing
  - Process live data
  - E.g., Apache Kfaka, Storm, Streamz, …
- Micro-batch processing
  - Something in between the Batch and Stream processing
  - Apache Spark

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

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



Structured and Unstructured Data

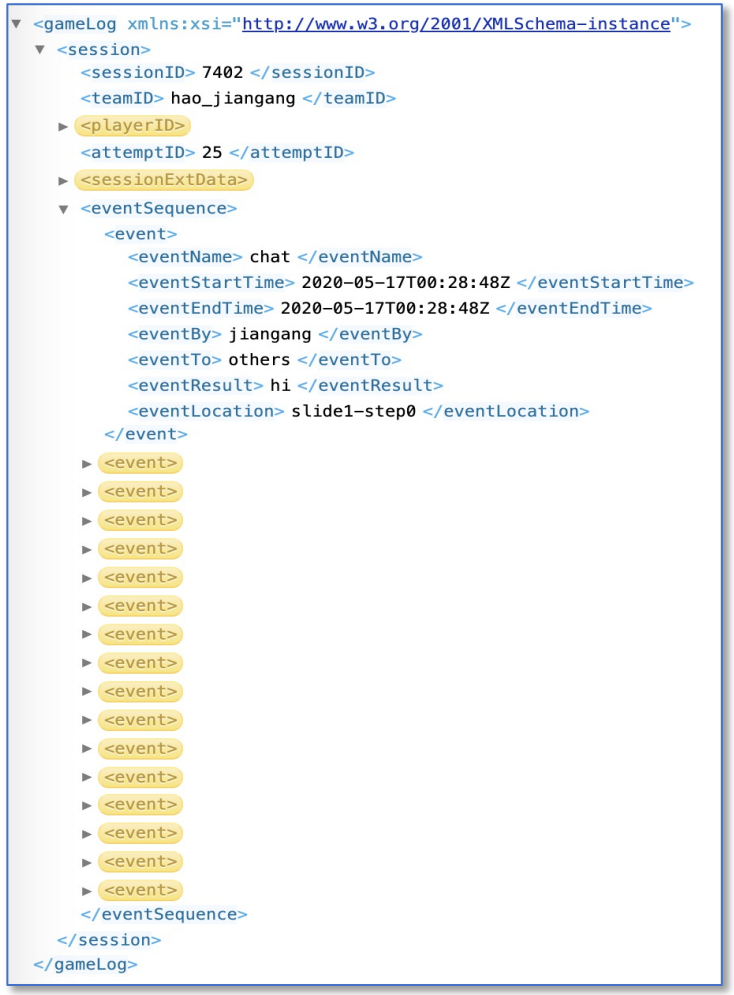
- **Unstructured data**: information that either does not have a predefined data model or is not organized in a pre-defined manner
- **Structured data**: information created by following a predefined data model (usually a tabular format)
- **Semi-structured data (also known as self-describing structure):**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

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.
```

Semi-structured - XML

- XML – Extensible Markup Language, 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"?>

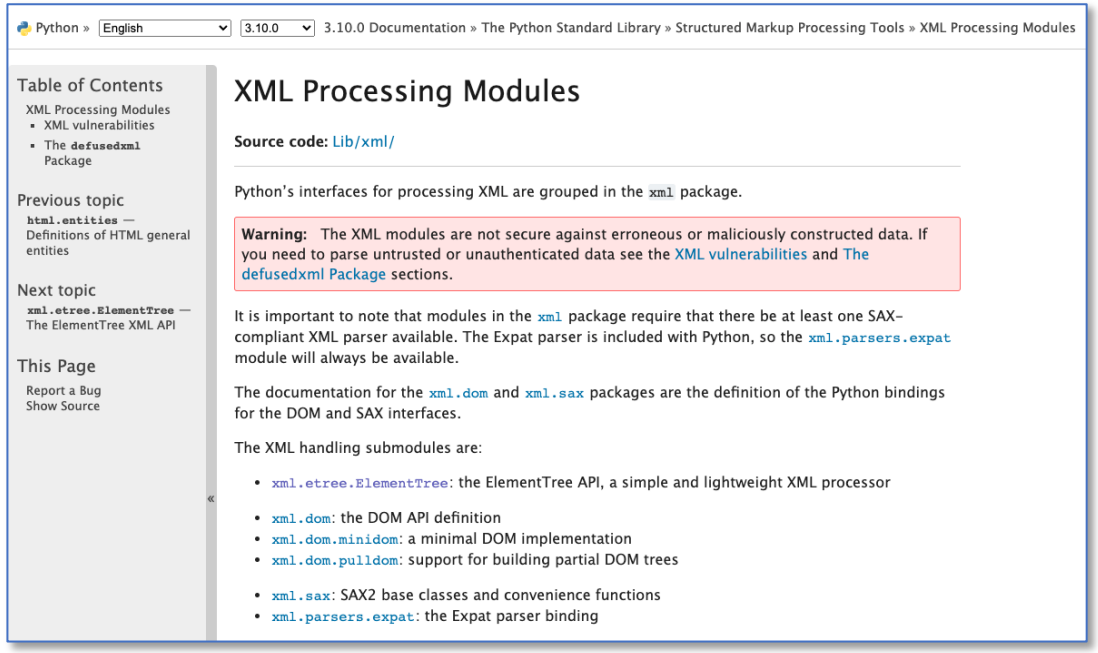


Data from EPCAL

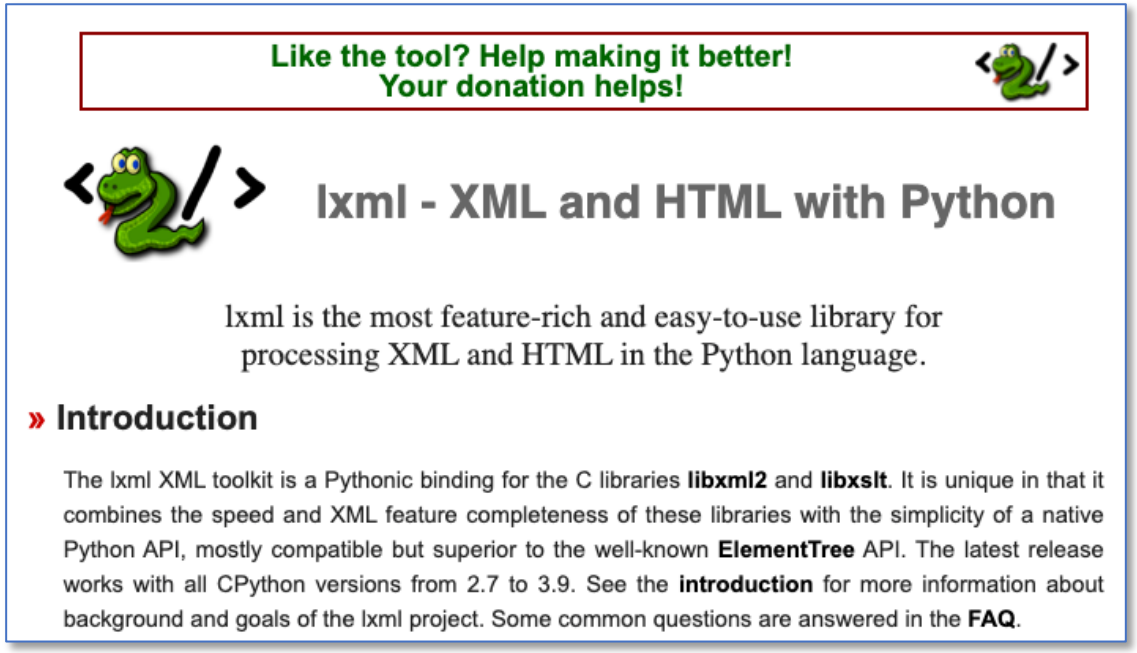
https://www.w3schools.com/xml/

Python Tool for XML

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



- LXML: https://lxml.de/



Semi-structutred - 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},
            + {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/

Specifying Structure - Schema

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

XML schema

```
<?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>
```

JSON schema

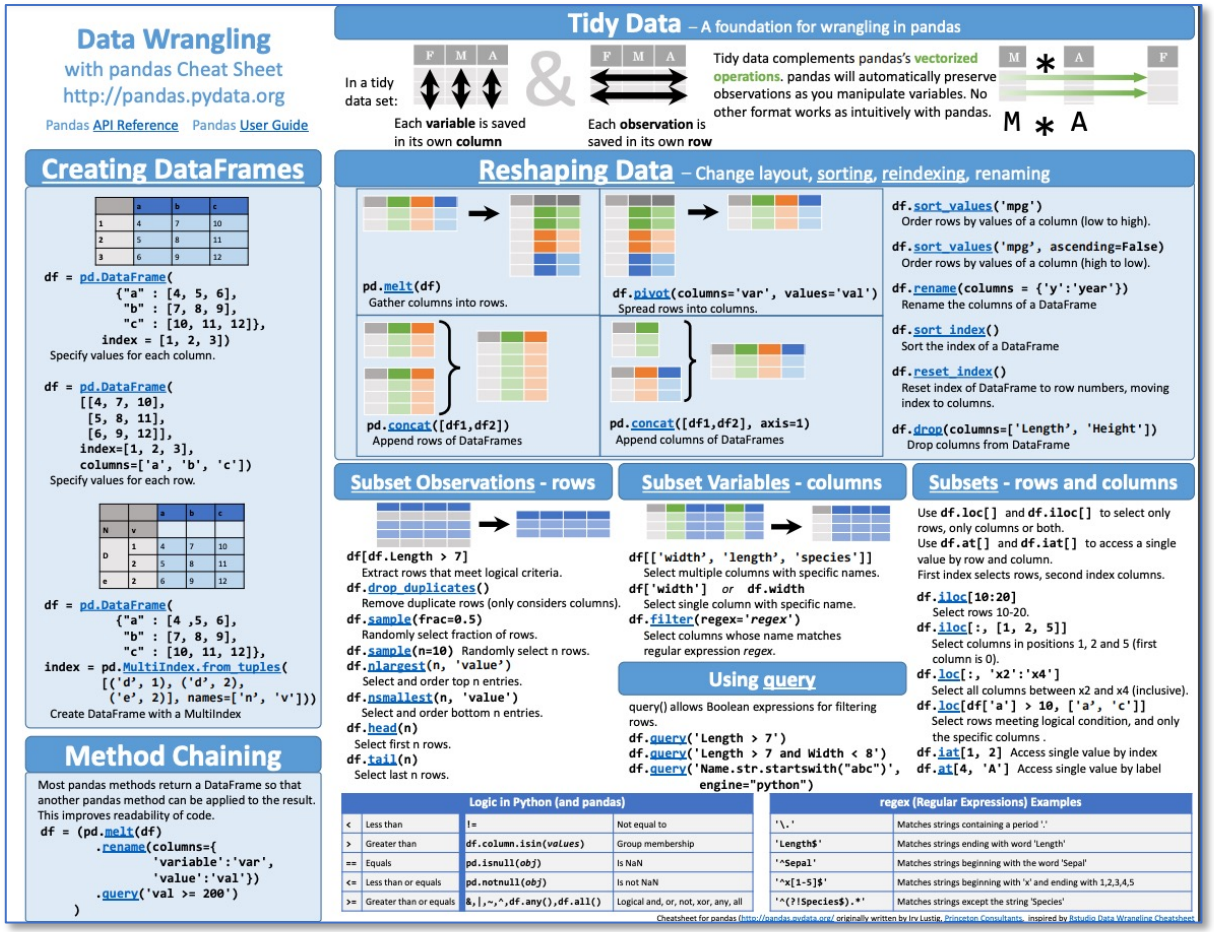
```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "description":"Gamelog Schema v1.2, Created by Jiangang Hao @ ETS",
  "type": "object",
  "properties": {
    "gameLog": {
      "type": "array",
      "items": {
        "type": "object",
        "properties": {
          "sessionID": {
              "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"
                },
```

Examples from ETS VPA data model

Python for Data Wrangling

pandas

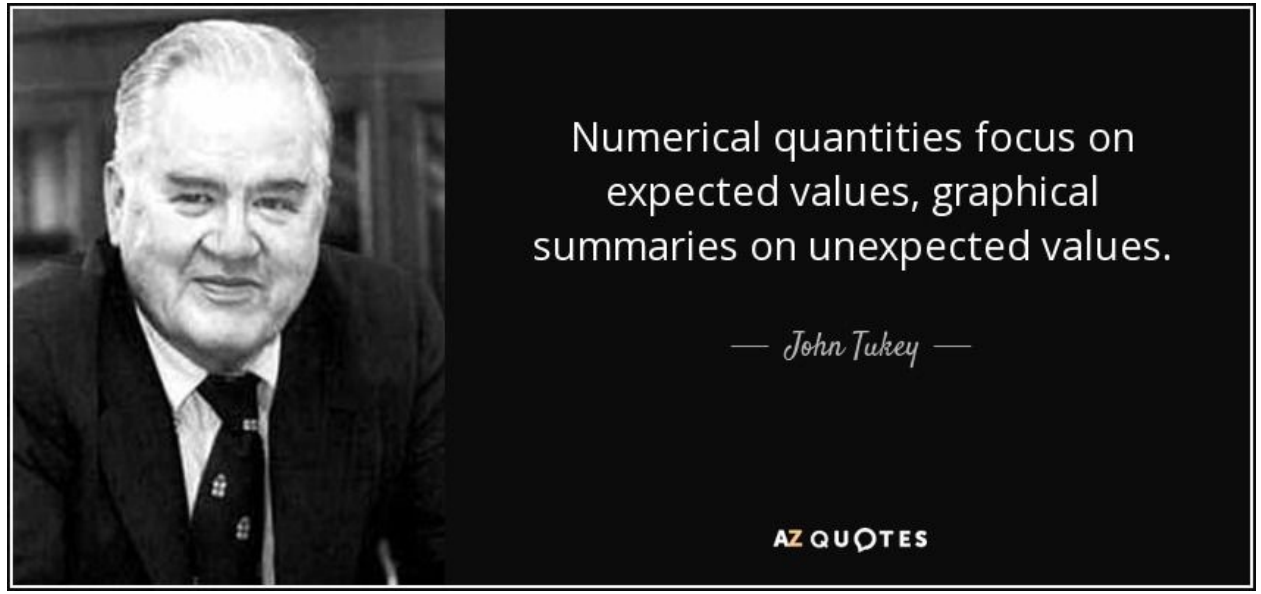
Original author(s)	Wes McKinney
Developer(s)	Community
Initial release	11 January 2008; 13 years ago[*citation needed*]
Stable release	1.3.0[1] / 2 July 2021; 3 months ago
Repository	github.com/pandas-dev/pandas
Written in	Python, Cython, C
Operating system	Cross-platform
Type	Technical computing
License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

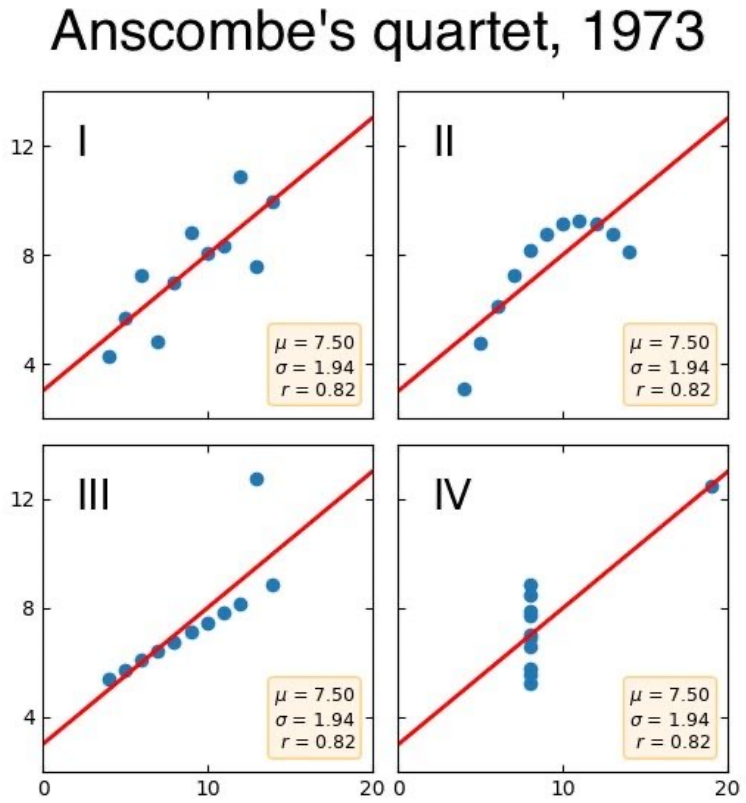


https://pandas.pydata.org/Pandas_Cheat_Sheet.pdf

Interactive Visualization and Dashboarding

Visualization is Important





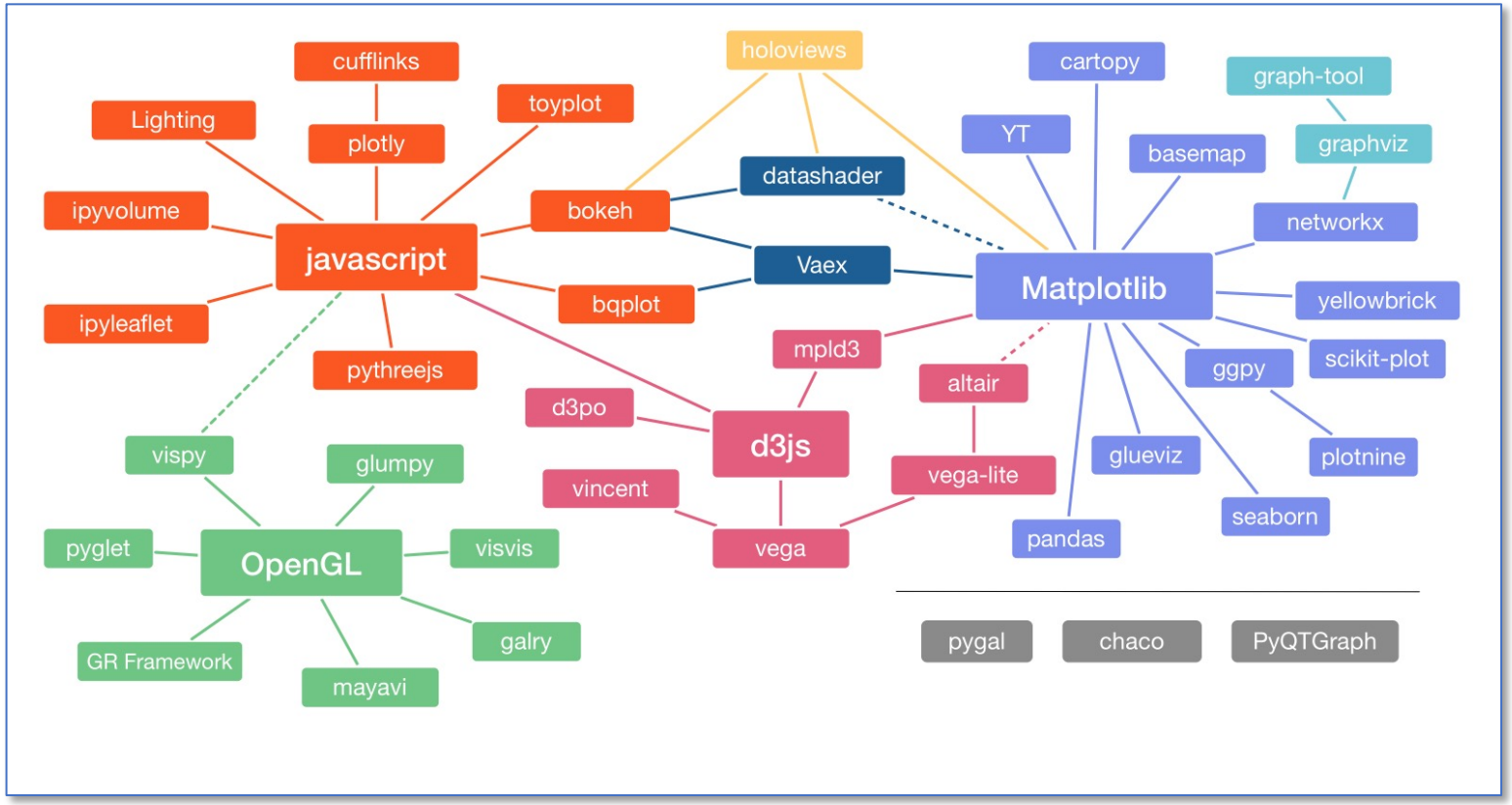
Steps to Visualization

- Understand the nature of your data
  - Categorical/continuous
  - Sparse
- Understand your goals
  - 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/

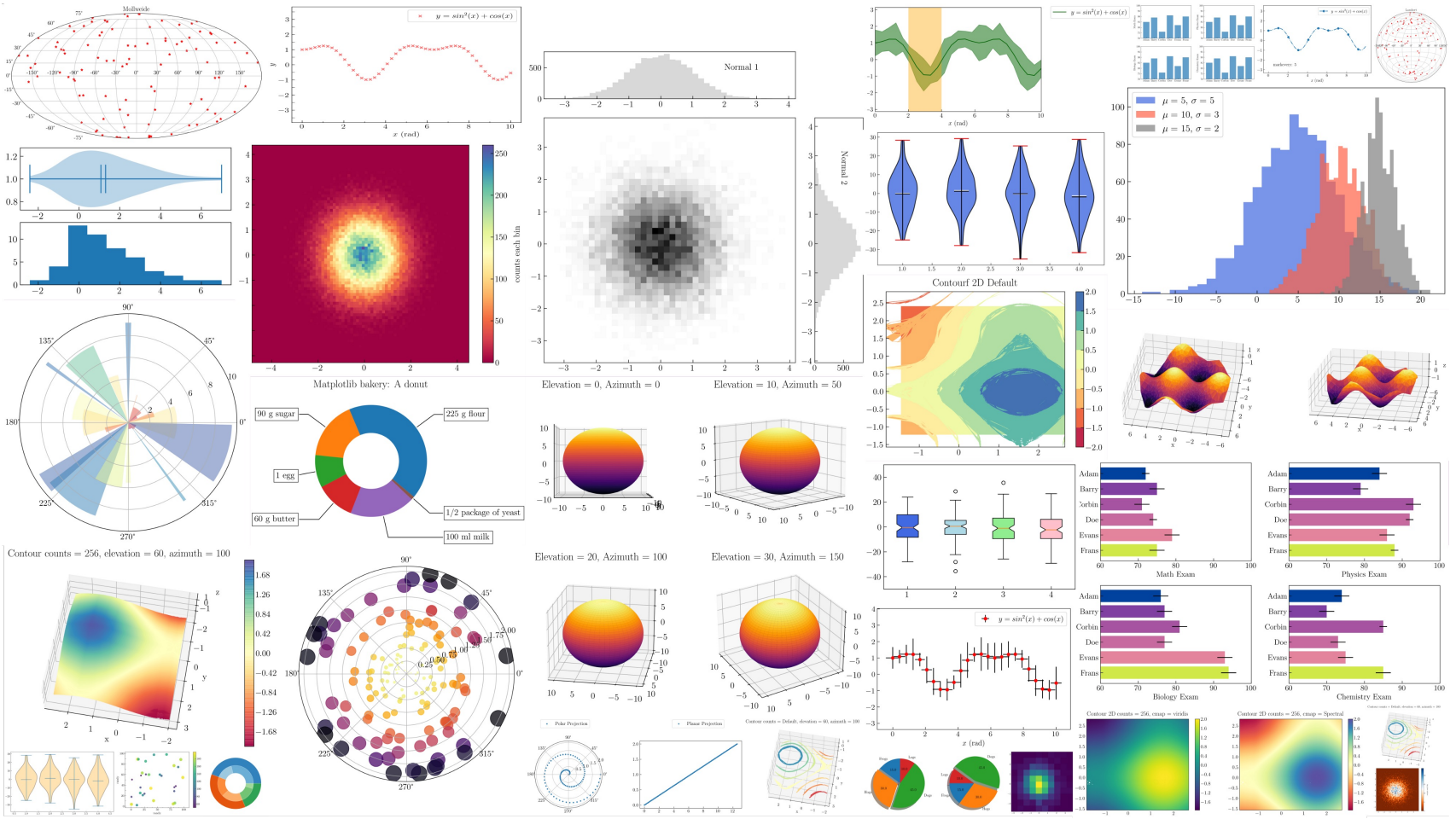
Visualization in Python

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



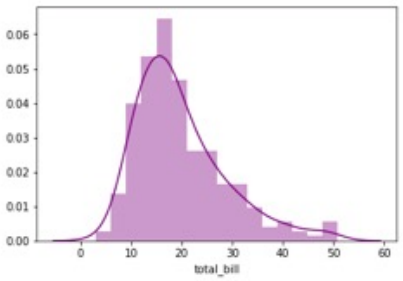
Matplotlib

https://matplotlib.org/

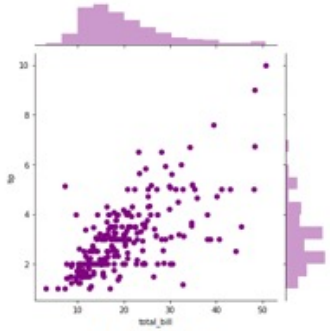


Seaborn: built on top of matplotlib

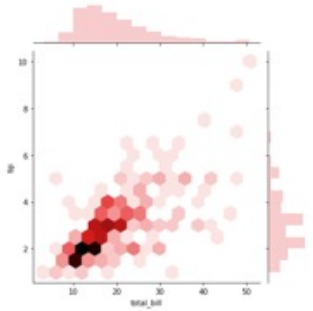
https://seaborn.pydata.org/



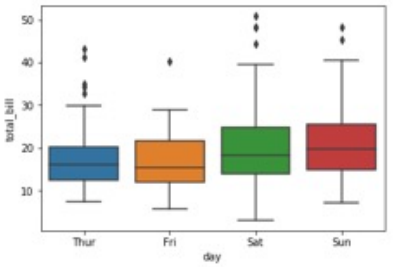
distplot



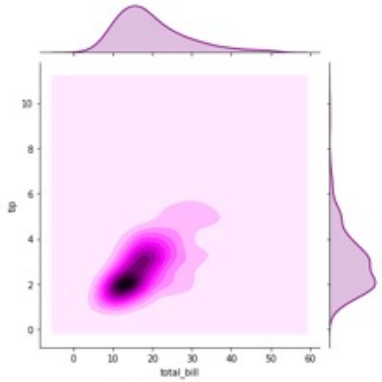
Jointplot



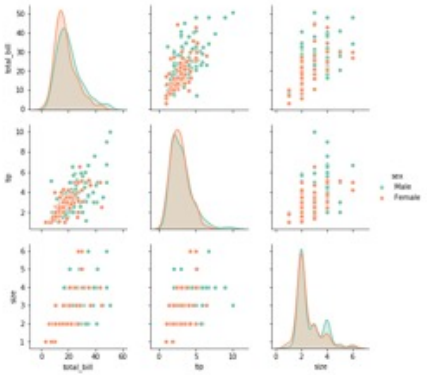
Hexplots



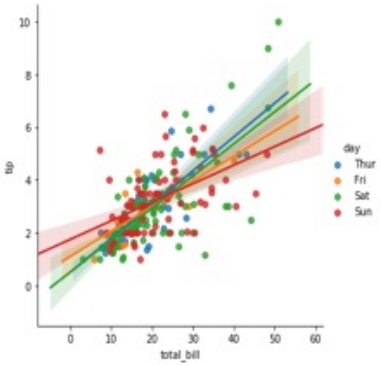
Boxplots



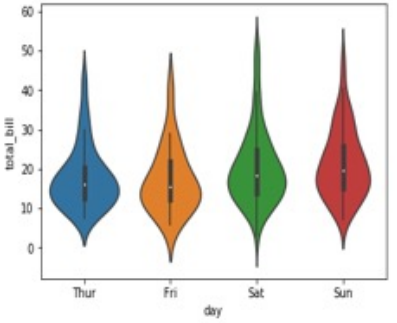
KDE Plot



Pair Plots

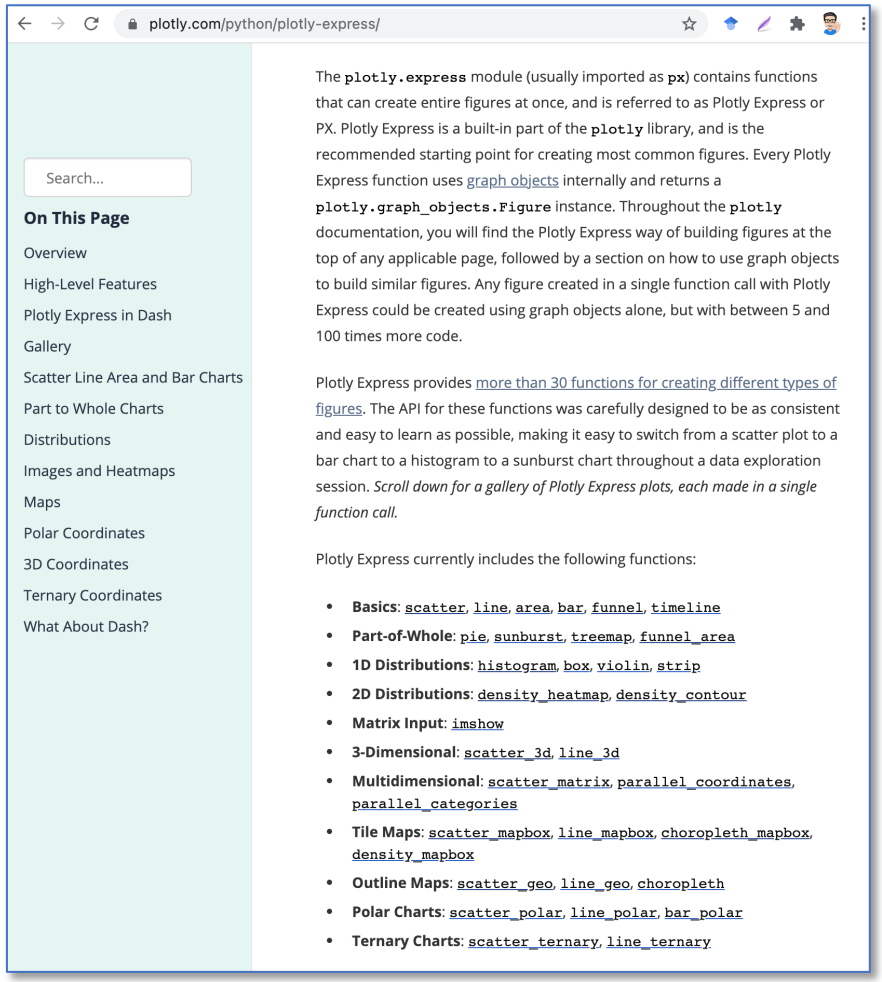


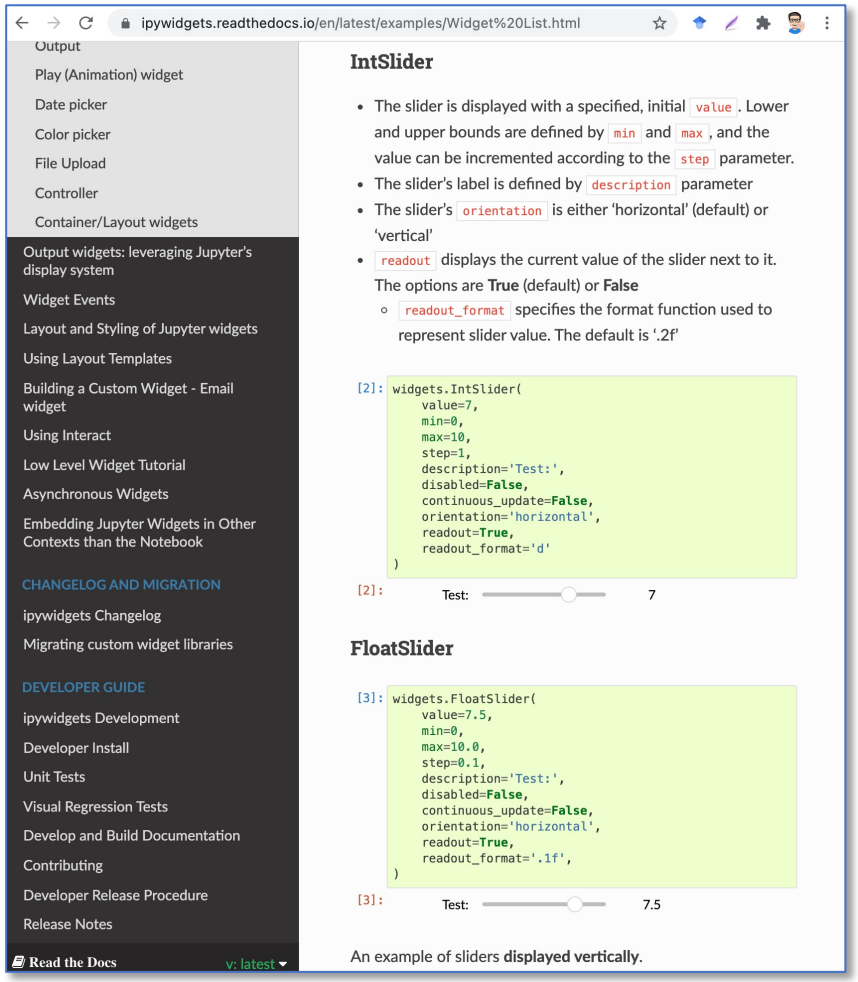
LM Plots



Violin Plots

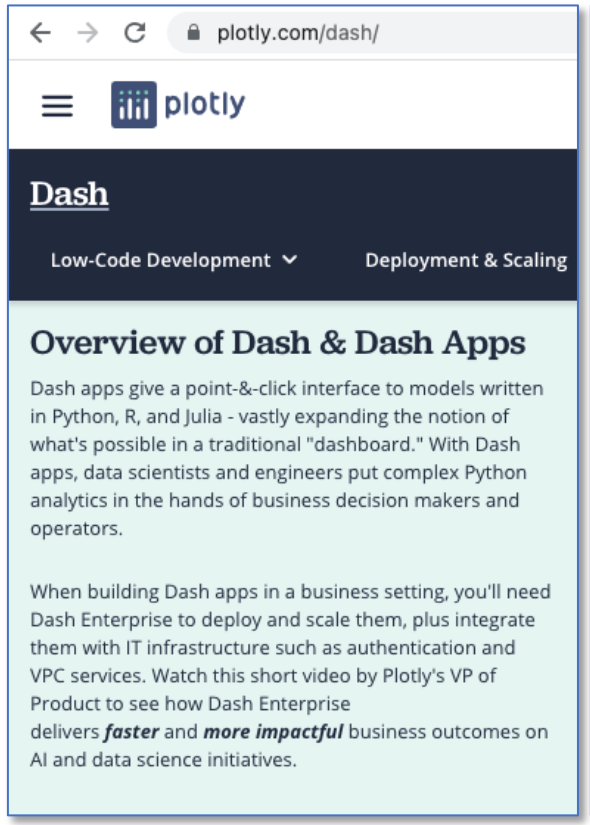
Plotly Express and ipywidgets

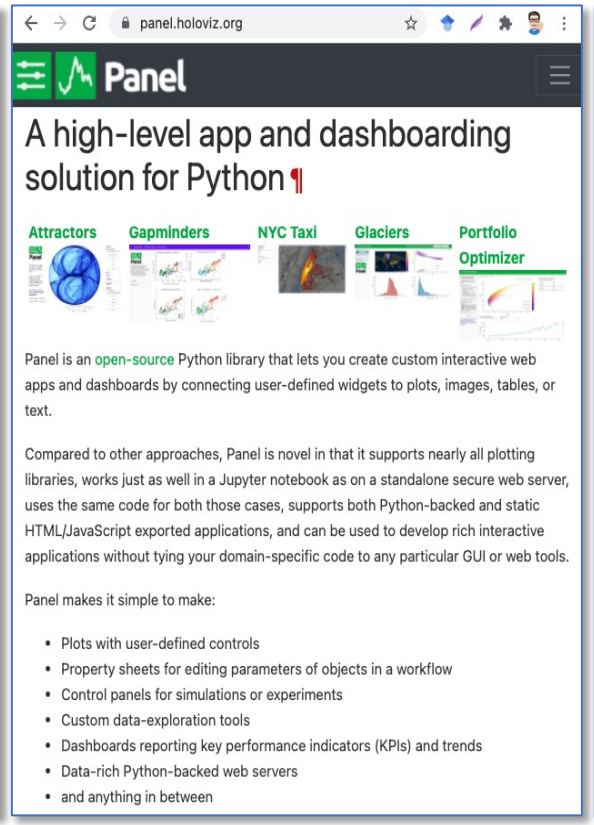


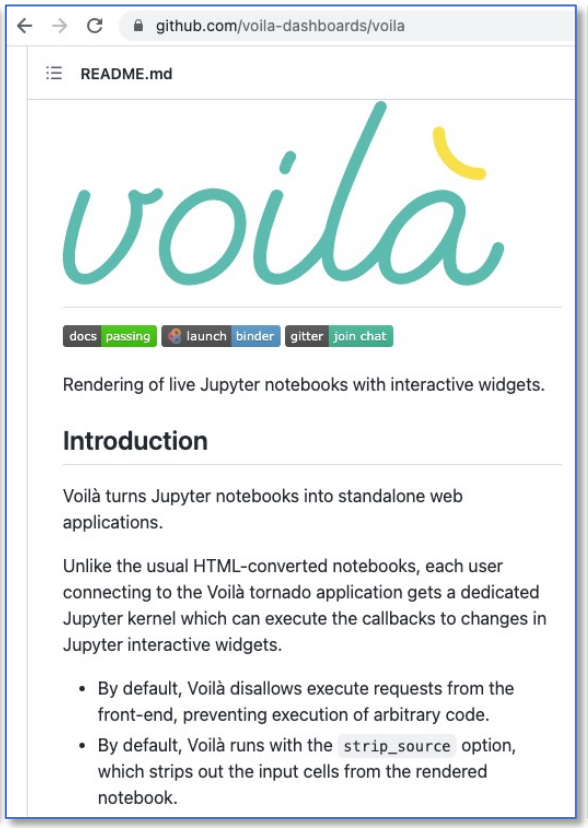


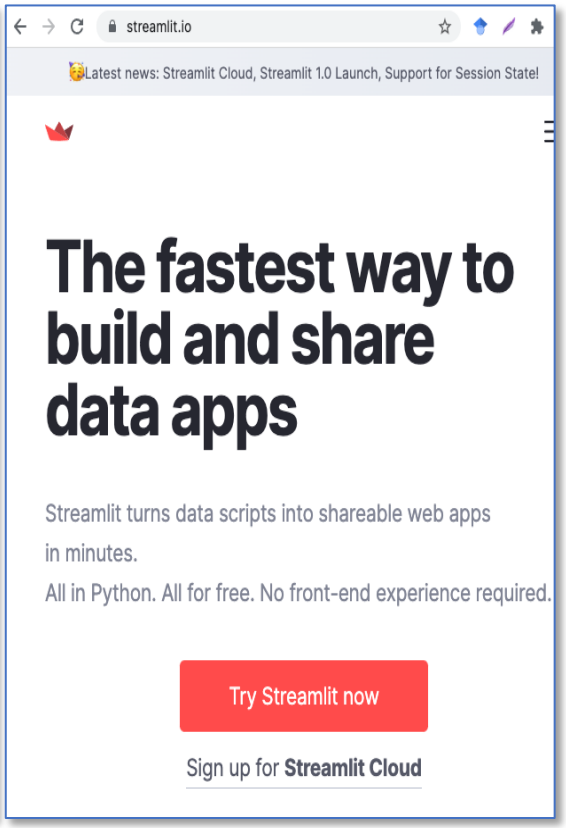
Demos

Major Dashboard Tools in Python



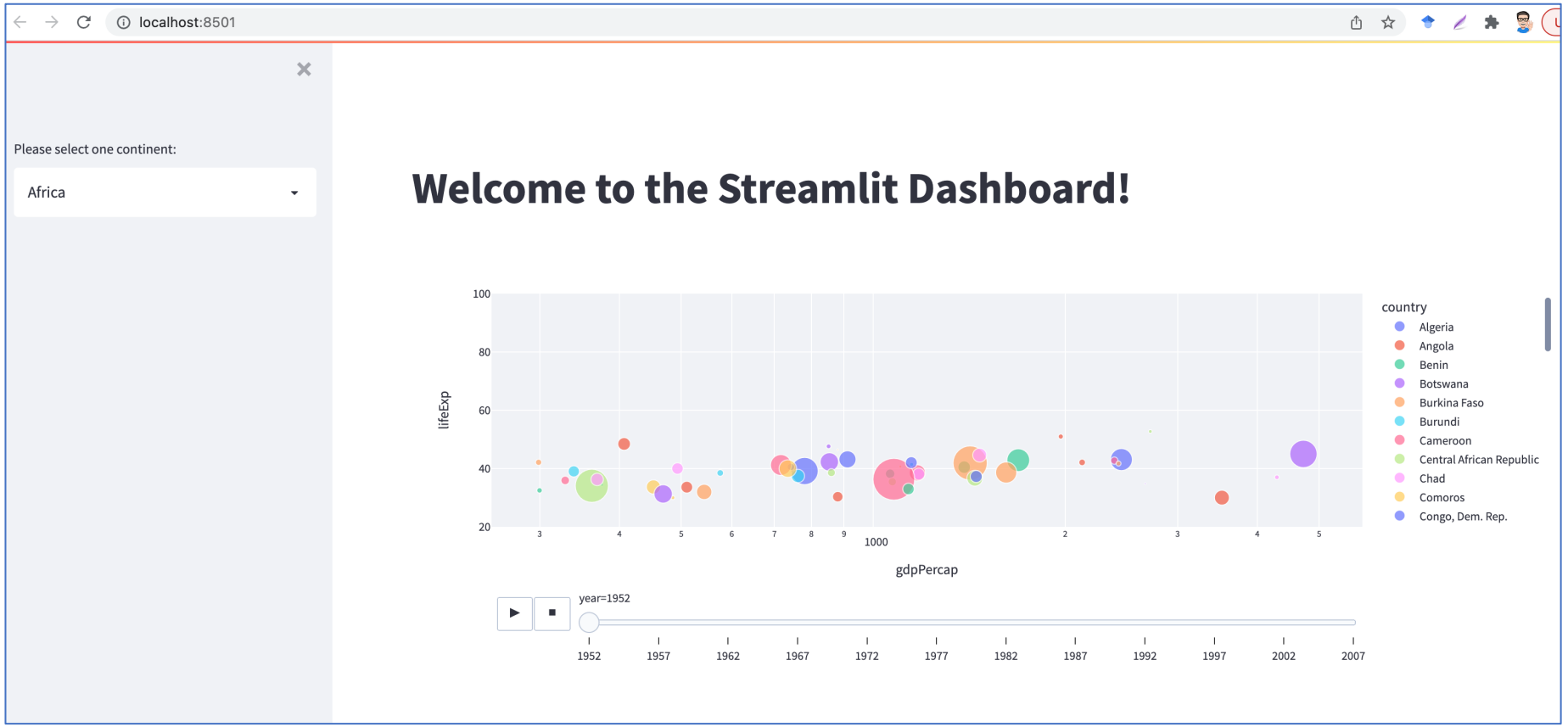






They are kind of equivalent to the Shiny package in R

Examples



Demo

Data Science Recap

- 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!