DI501 Introduction to Data Science

Lecture 2 – Data Science Frameworks

From Business Problems to Data Science Tasks

- How do we decompose a data analytics problem into pieces such that each piece matches a known task for which tools are available?
  - Avoid waste of time and resources for reinventing the wheel
  - Focus attention on more interesting parts that require human creativity and intelligence
- There are a large number of algorithms, but there are only a handful of fundamentally different types of tasks that these algorithms address

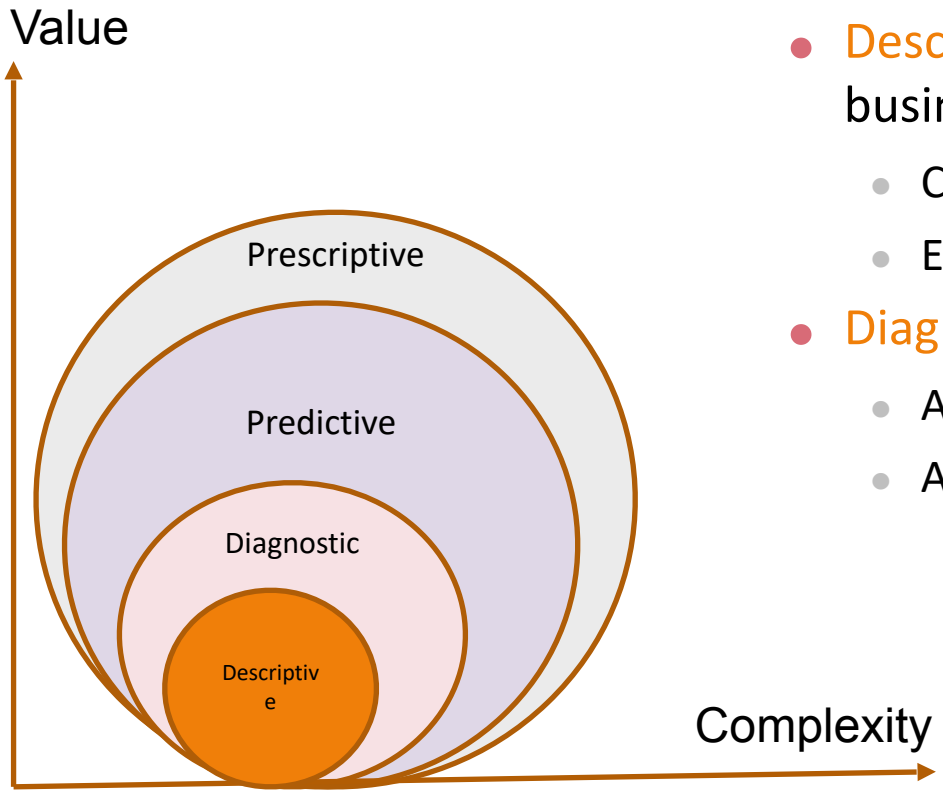
Task Categories

- Regression: Estimate or predict the numerical value of a variable for a given individual
  - Ex: *How much will a given customer spend next week?*
- Classification: Predict which of a set of classes an individual belongs to
  - Ex: *Will a given cell phone customer leave then his/her contract expires?*
- Similarity: Identify similar individuals based on what we know about them
  - Ex: *Which customers are similar to a given customer?*
- Clustering: Group individuals in a given set by their similarity
  - Ex: *How to group our customers into market segments?*

Task Categories

- Co-occurrence Grouping: Find associations between entities based on transactions involving them
  - Ex: *What items are purchased together?*
- Profiling: Characterize the typical behavior of an individual, group, or population
  - Ex: *What is the typical cell phone usage of this customer segment?*
- Link Prediction: Predict connections between items
  - Ex: *Which customers are friends?*
- Data Reduction: Transform a large dataset into a smaller dataset
  - Ex: *How can we reduce product preferences of our customers to a much smaller dataset so that we can deal with data easily*
- …

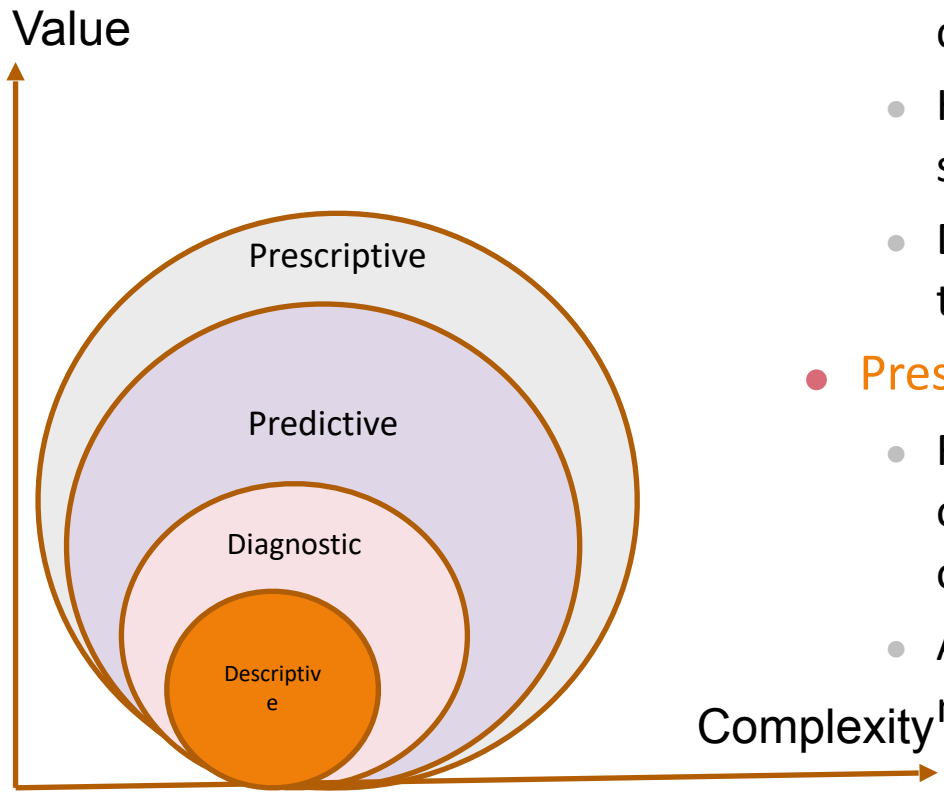
Stages of Analytics



- What is the data telling you?
- Descriptive: What's happening in my business?
  - Comprehensive, accurate and live data
  - Effective visualization
- Diagnostic: Why is it happening?
  - Ability to drill down to the root-cause
  - Ability to isolate all confounding information

Ref: https://www.kdnuggets.com/2017/07/4-types-data-analytics.html

Stages of Analytics



- Predictive: What's likely to happen?
  - Business strategies have remained fairly consistent over time
  - Historical patterns being used to predict specific outcomes using algorithms
  - Decisions are automated using algorithms and technology
- Prescriptive: What do I need to do?
  - Recommended actions and strategies based on champion/challenger testing strategy outcomes
  - Applying advanced analytical techniques to make specific recommendations

Ref: https://www.kdnuggets.com/2017/07/4-types-data-analytics.html
www.principa.co.za

Stages of Analytics

Example:

- Descriptive:
  - The admission of patients to a hospital over a year.
  - Complaints, gender, the number of people admitted
- Diagnostic:
  - Specific complaints are highly correlated with the number of admissions (multicollinearity issues?)
  - People forget to enter specific entries due to time constraints.
- Predictive:
  - You try to predict the number of admissions over the next week.
- Prescriptive:
  - As the number of admissions will be high, you know the potential implications. You increased the number of beds and staff.

Machine Learning Categories

- Supervised
- Unsupervised
- Semi-supervised
- Reinforcement

Supervised vs Unsupervised Methods

- Supervised: A specific target (or desired output) is known for each sample in the input dataset
  - classification (categorical target) and regression (numeric target) are generally solved with supervised methods
- Unsupervised: No specific target can be provided for the samples in the input dataset
  - clustering, co-occurrence grouping, and profiling are generally solved with unsupervised methods
- Similarity matching, link prediction, and data reduction could be solved with supervised or unsupervised methods

Two Major Phases

- First, we use available data to find patterns and build models
  - Ex: *Use historical data to produce a model for the prediction of customer churn*
  - This phase is typically composed of many sub-phases
- Then, we use these patterns and models in decision making
  - Ex: *Use the generated model to predict whether a given customer will leave*
    - If the customer is likely to leave, the company may offer special deals prior to the expiration of her/his contract

Semi-Supervised Learning

- Algorithms are trained on both labelled and unlabeled data.
  - Labelled data: low in number
  - Unlabeled data: high in number
- Assumptions:
  - Continuity assumption
  - Cluster assumption
  - Manifold assumption

Reinforcement Learning

- Different from supervised learning, an agent learns from its experience and makes decisions on the run time.
- Main points:
  - Input: The input should be an initial state from which the model will start
  - Output: There are many possible output as there are variety of solution to a particular problem
  - Training: The training is based upon the input. The model will return a state and the user will decide to reward or punish the model based on its output.
  - The model keeps continues to learn.
  - The best solution is decided based on the maximum reward.

Ref: https://www.geeksforgeeks.org/what-is-reinforcement-learning/

How to Execute and Manage Data Science Projects?

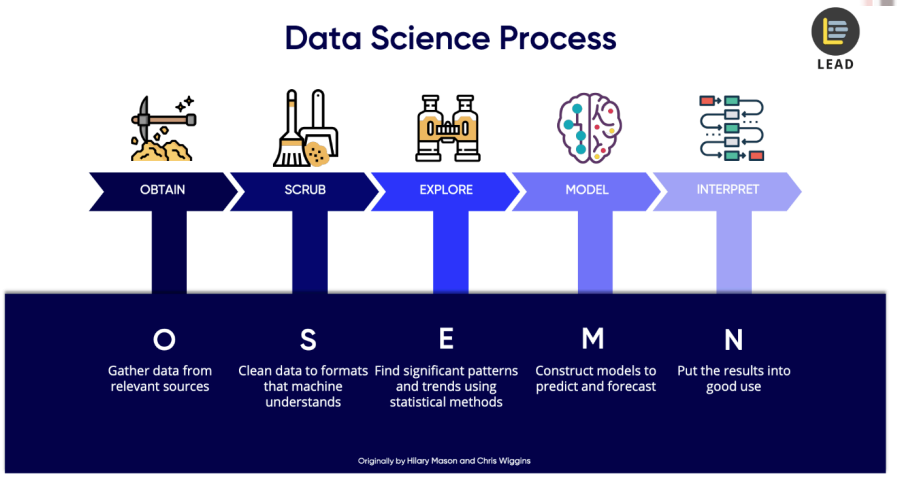
- A well defined framework is needed to carry out data science projects to
  - guide the implementation (i.e., translate business problems into data-driven solutions),
  - effectively manage the projects,
  - get the highest return on investment,
  - repeat previous successes
- Ad hoc processes and lack of a robust methodology leads to several issues such as poor team coordination, slow information sharing, scope creep, producing random and false discoveries, lack of reproducibility, and management inefficiencies

Data Science Frameworks

- Definition: A framework, or software framework, is a platform for developing software applications. It provides a foundation on which software developers can build programs for a specific platform.
- A good framework enables
  - knowing where and how to start,
  - establishing a shared understanding across business and data science teams,
  - getting reasonable expectations,
  - reusing knowledge from previous projects,
  - knowing where and how to use the results,
  - knowing how to evaluate the business impact
- There are several frameworks or process models that can be applied in data science projects
  - Ex: *OSEMN, SEMMA, KDD, CRISP-DM, TDSP*
- Nevertheless, many organizations are not following a well-defined methodology

OSEMN Framework

- OSEMN Pipeline
  - **O**btaining data
  - **S**crubbing / Cleaning data
  - **E**xploring / Visualizing data
    - allow us to find patterns and trends
  - **M**odeling data
    - give us our predictive power as a wizard
  - i**N**terpreting models and data



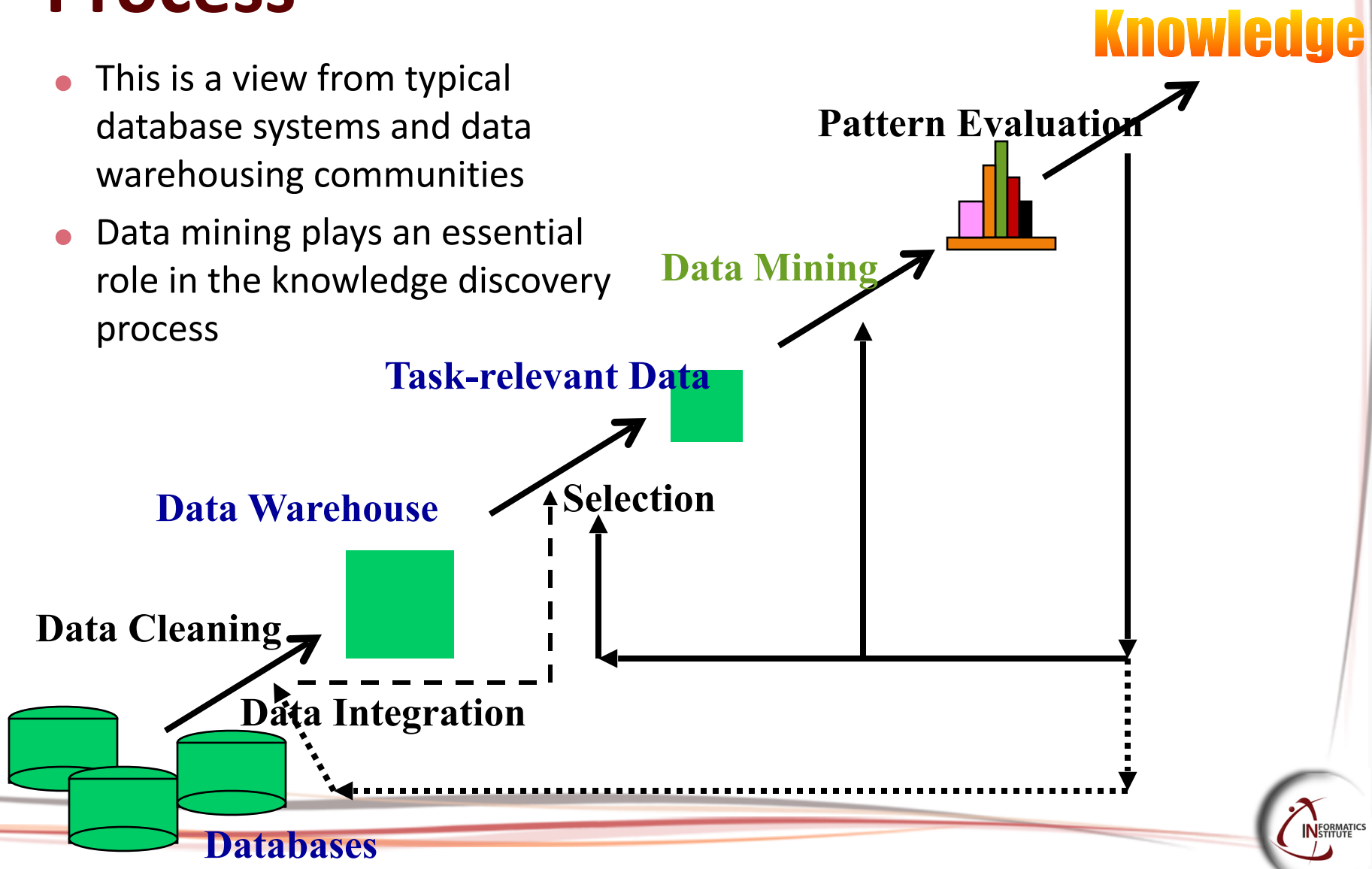
Data Science Process (a.k.a the O.S.E.M.N. framework)

OSEMN

- Obtain: Get data from sources such as
  - SQL/NoSQL Databases, Data warehouses, Big Data (e.g., Hadoop)
  - Web APIs, Web scraping
  - Repositories (e.g., Kaggle)
- Scrub: organizing and tidying up the data by filtering, cleaning, imputing, merging, transforming
  - Python, R, MapReduce, Spark, …
- Explore: Inspect available data and its properties, visualize data, compute descriptive statistics, correlation, etc.
  - Visualization, Inferential Statistics (R, Python with Numpy, Pandas, Matplotlib, etc., Spark, Flink, …)
- Model: create models such as regression, classification, and clustering, select features, tune parameters, evaluate models
  - Python with scikit-learn, R with caret, Spark ML, Flink ML
- iNterpret: Interpret models and data, present findings in such a way that business problems can be solved
  - Data storytelling, Visualization tools

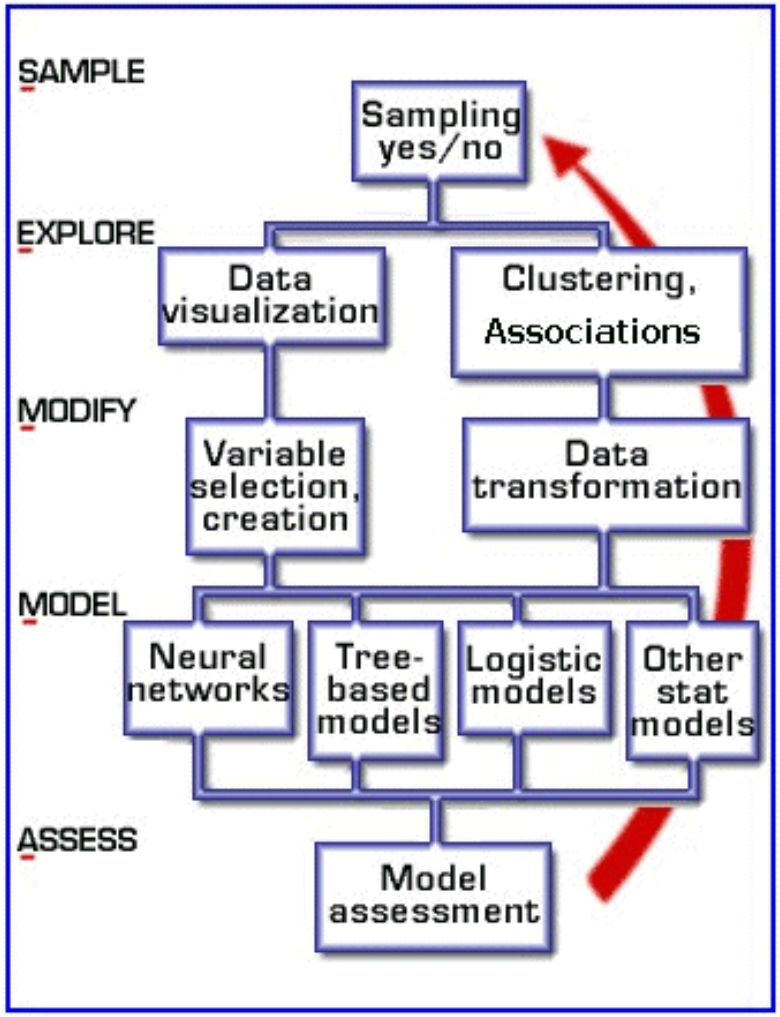
Knowledge Discovery (KDD) Process

- This is a view from typical database systems and data warehousing communities
- Data mining plays an essential role in the knowledge discovery process



SEMMA

- SEMMA: **S**ample, **E**xplore, **M**odify, **M**odel and **A**ssess
- It refers to core process of conducting data mining
  - All SEMMA steps may not be included in analysis
  - It may be necessary to repeat one or more of the steps several times
- SAS Enterprise Miner is an integrated product that provides a front end to the SEMMA mining process

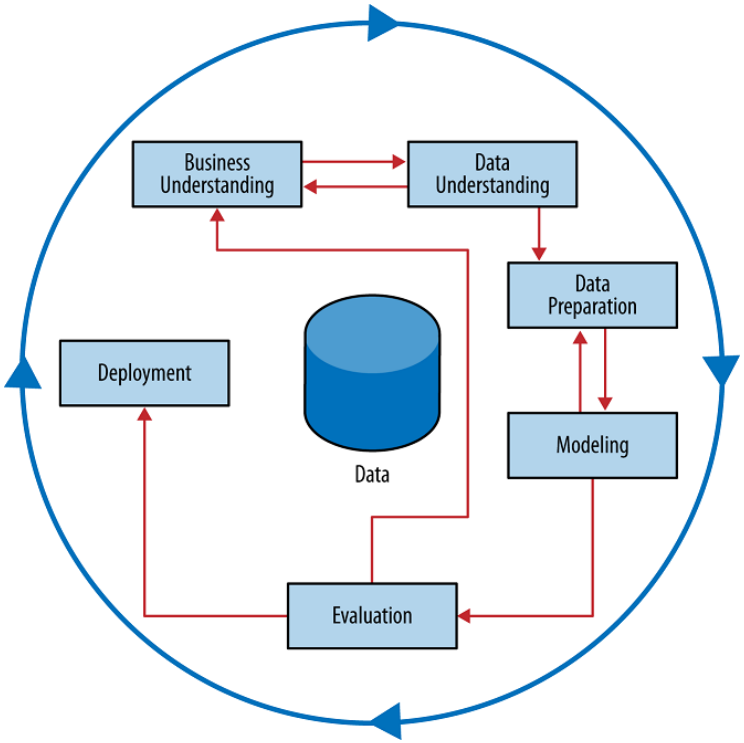


SEMMA

- Sample: Sample data by extracting a portion of a large dataset big enough the contain the significant information, yet small enough to manipulate
- Explore: Search for unanticipated trends and anomalies in order to gain understanding and ideas
- Modify: create, select, transform variables to prepare the data for analysis. Identify outliers, replace missing values, etc.
- Model: Fit a predictive model to a target variable
- Assess: Evaluate the competing models

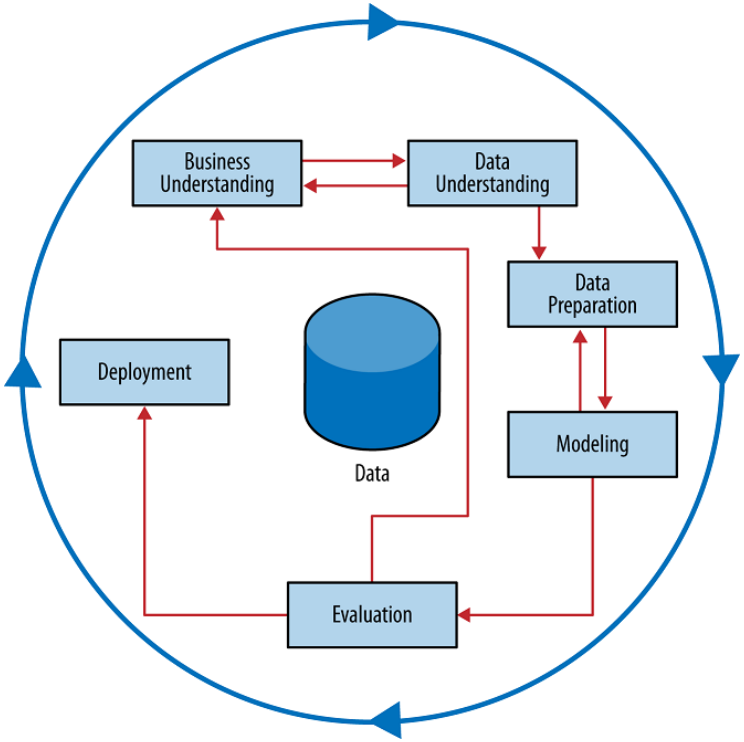
CRISP-DM

- CRISP-DM: **CR**oss **I**ndustry **S**tandard **P**rocess for **D**ata **M**ining
  - Proposed in 1996
  - In 2015, IBM released a new methodology called ASUM-DM which refines and extends CRISP-DM
- It is the most widely-used analytic methodology according to many opinion polls



CRISP-DM

- It makes explicit the fact that iteration is the rule rather than the exception
- There are six phases
- The sequence of the phases is not strict
  - The arrows indicate only the most important and frequent dependencies
  - Outcome of the current phase may determine the next phase
  - Backtracks and repetitions are common



CRISP-DM:
Business Understanding

- Initially, it is vital to understand the problem to be solved
  - Business projects seldom come pre-packaged as clear and unambiguous problems
  - Iterations may be necessary for an acceptable solution to appear
  - It is also necessary to think carefully about the use scenarios
- Involves, understanding the project objectives and requirements from a business perspective, then converting this knowledge into a data mining problem definition and a preliminary plan designed to the achieve objectives
  - What the client really wants to accomplish?
  - Uncover important factors (constraints, competing objectives)

CRISP-DM:
Data Understanding

- Data comprise the available raw material from which the solution will be built
- It is important to understand the strengths and limitations of the data because rarely is there an exact match with the problem
- We start with an initial data collection and proceed with activities in order to get familiar with the data, to identify data quality problems, to discover initial level of insights into the data or to detect interesting subsets to form hypotheses for hidden information

CRISP-DM:
Data Preparation

- It covers all activities to construct the final dataset from the initial raw data
- Quality of cleaned data will impact on model performance
- Data preparation tasks are likely to be performed multiple times and not in any prescribed order
- Tasks include table, record and attribute selection, and integration as well as transformation and cleaning of data for modeling tools

CRISP-DM: Modelling

- In this phase, various modeling techniques are selected and applied and their parameters are calibrated to optimal values
- There are several techniques that can be applied for the same data mining problem type
  - Some techniques have specific requirements on the form of data.
  - Therefore, stepping back to the data preparation phase is often necessary.

CRISP-DM: Evaluation

- Thoroughly evaluate the model and review the steps executed to construct the model to be certain it properly achieves the business objectives
- A key objective is to determine if there is some important business issue that has not been sufficiently considered
- At the end of this phase, a decision on the use of the data mining results should be reached

CRISP-DM: Deployment

- In this phase, we need to determine how the results to be utilized and plan deployment, monitoring and maintenance
- The knowledge gained will need to be organized and presented in a way that the customer can use it
- Depending on the requirements, the deployment phase can be as simple as generating a report or as complex as implementing a repeatable data mining process across the enterprise

CRISP-DM: Criticisms

- It is widely adopted
- It has a good focus on the business understanding
- It also covers deployment
- The model is not actively maintained
  - The official site, crisp-dm.org, is no longer being maintained
- The framework itself has not been updated on issues on working with new technologies, such as big data
  - Big data means that additional effort can be spent in the data understanding phase

KDD, SEMMA and CRISP-DM Comparison

- Summary of the correspondences between KDD, SEMMA and CRISP-DM

KDD	SEMMA	CRISP-DM
Pre KDD	----------	Business understanding
Selection	Sample	Data Understanding
Pre processing	Explore	
Transformation	Modify	Data preparation
Data mining	Model	Modeling
Interpretation/Evaluation	Assessment	Evaluation
Post KDD	----------	Deployment

Azevedo, Ana Isabel Rojão Lourenço, and Manuel Filipe Santos. "KDD, SEMMA and CRISP-DM: a parallel overview." *IADS-DM* (2008).

KDD, SEMMA and CRISP-DM Based on Waterfall Model

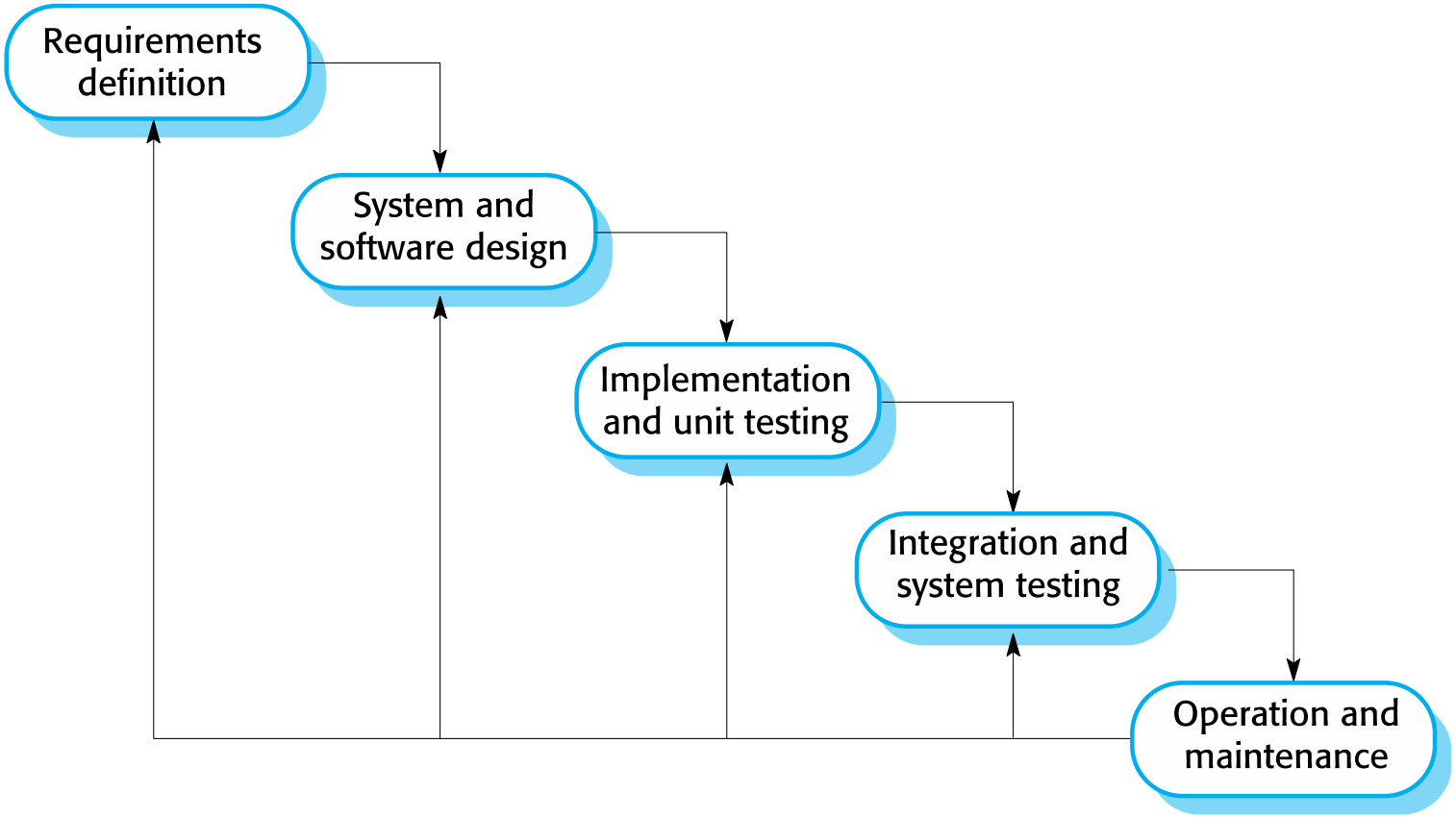


Figure: Ian Sommerville – Software Engineering

Big Data Analytics Life Cycle

- «The data science lifecycle for BDA systems is currently in the same situation as SDLC (software development life cycle)-driven development prior to the introduction of agile methods*»
- The de-facto standard CRISP-DM (Cross-Industry Standard Process Model–Data Mining) was considered essentially a waterfall model
- Some organizations are employing specific project management approaches that typically combine elements of agile and data mining methodologies

*Grady, N. W., Payne, J. A., & Parker, H. (2017, December). Agile big data analytics: AnalyticsOps for data science. In *2017 IEEE International Conference on Big Data (Big Data)* (pp. 2331-2339). IEEE.

Emerging Approaches

- The purpose of using agile analytics is to reach a point of optimality between generating value from data and the time spent getting there.
- One hybrid approach can be to use Scrum with CRISP-DM
  - Scrum is a lightweight, iterative and incremental framework that divides the project into sprints (mini projects) undertaken by self organizing cross-functional teams
- Team Data Science Process (TDSP) is an agile, iterative, data science process for executing and delivering advanced analytics solutions

Agile Software Development

Agile software development is a set of principles and practices used by self-organizing teams to rapidly and frequently deliver customer-valued software.

«It is an umbrella term for a set of frameworks and practices based on the values and principles expressed in the Manifesto for Agile Software Development and the 12 Principles behind it»*

*Agile Allience

Agile Software Development

- Agile is engrained as a philosophy, not a methodology,
- It is an adaptive approach to doing work based on empirical evidence,
- Agile embraces continuous improvement and expects ongoing evolution,
- There is no one or the «Agile» method

Ref: Özden Özcan Top's slides

The Agile Manifesto

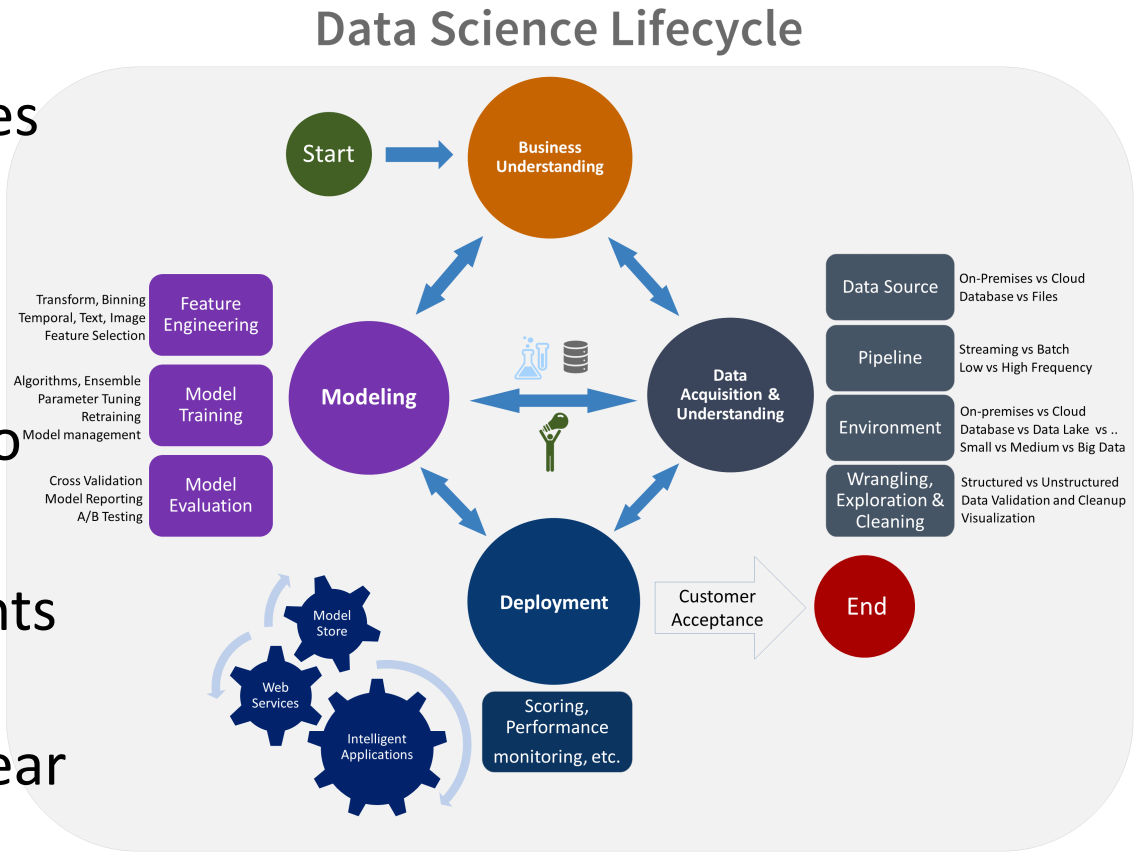
Individuals and Interactions	over	Processes and Tools
Working Software	over	Comprehensive Documentation
Customer Collaboration	over	Contract Negotiation
Responding to change	over	Following a plan

That is, while there is value in the items on the right, we value the items on the left more.

*www.agilemanifesto.org

Microsoft Team Data Science Project (TDSP)

- It is a team-oriented solution that emphasizes teamwork and collaboration throughout the project
- The lifecycle is similar to CRISP-DM
- Its takes several elements from Scrum such as backlog, sprints, and clear team roles



Microsoft Team Data Science Project (TDSP): Work Item Types

- **Feature**: A Feature corresponds to a project engagement. Different engagements with a client are different Features, and it's best to consider different phases of a project as different Features.
- **User Story**: User Stories are work items needed to complete a Feature end-to-end. Examples of User Stories include:
  - Get data
  - Explore data
  - Generate features
  - Build models
  - Operationalize models
  - Retrain models
- **Task**: Tasks are assignable work items that need to be done to complete a specific User Story. For example, Tasks in the User Story *Get data* could be:
  - Get SQL Server credentials
- **Bug**: Bugs are issues in existing code or documents that must be fixed to complete a Task. If Bugs are caused by missing work items, they can escalate to be User Stories or Tasks.

Ref: https://docs.microsoft.com/en-us/azure/architecture/data-science-process/agile-development

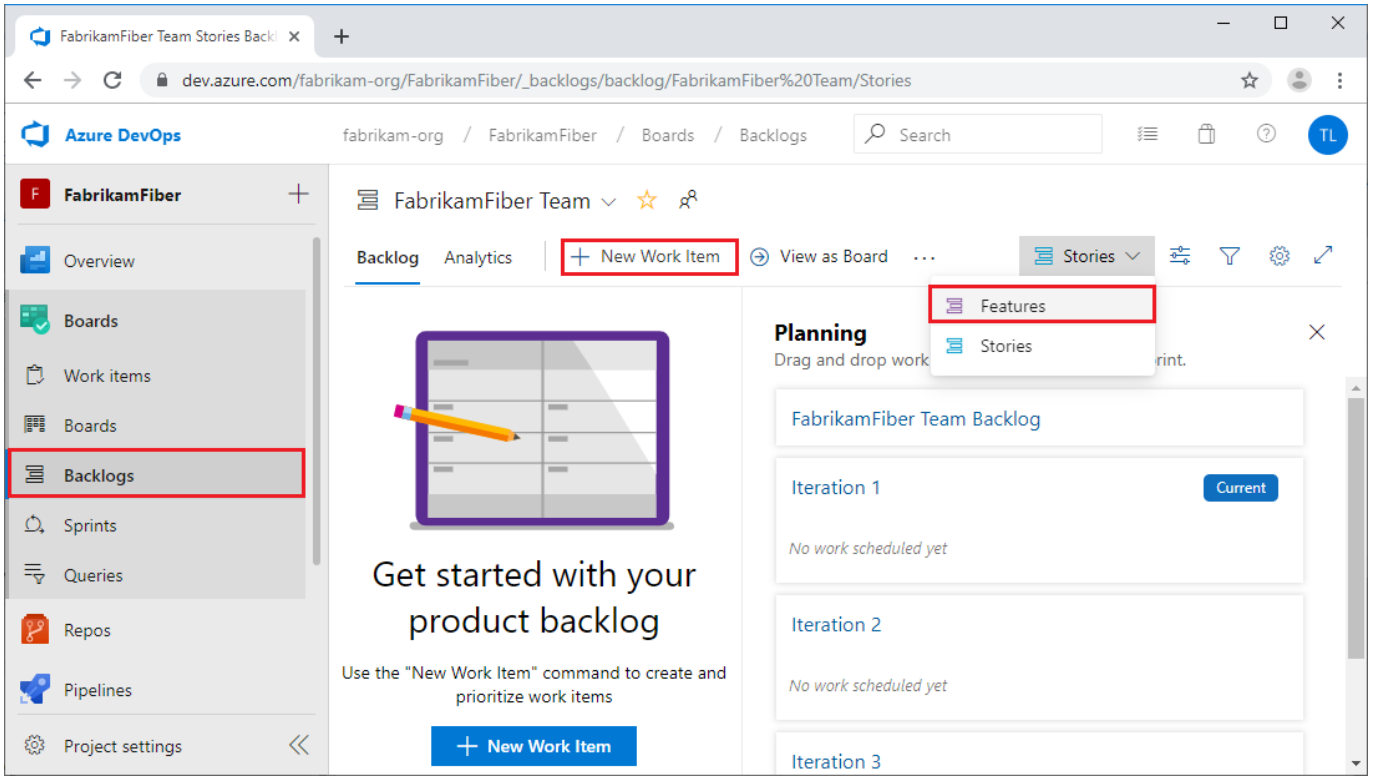
Microsoft Team Data Science Project (TDSP): Work Item Types: Example

- **Feature 1**: Showing recommendations to a user
- **User Story 1.1**: Get user X's previous purchase history
- **User Story 1.2:** Get user X's browsing patterns
- **User Story 1.3:** Get all users' data for training dataset
- **User Story 1.4:** Apply collaborative filtering algorithm for recommendation
- **User Story 1.5:** Give a survey to a user who has registered to the system for initial recommendation
- **Task 1.1.1**: Query database for retrieving user X data
- **Bug 1.1**: SQL error when retrieving user X data

- TDSP borrows the concepts of Features, User Stories, Tasks, and Bugs from software code management (SCM). The TDSP concepts might differ slightly from their conventional SCM definitions.

Microsoft Team Data Science Project (TDSP): Work Item Types: Example

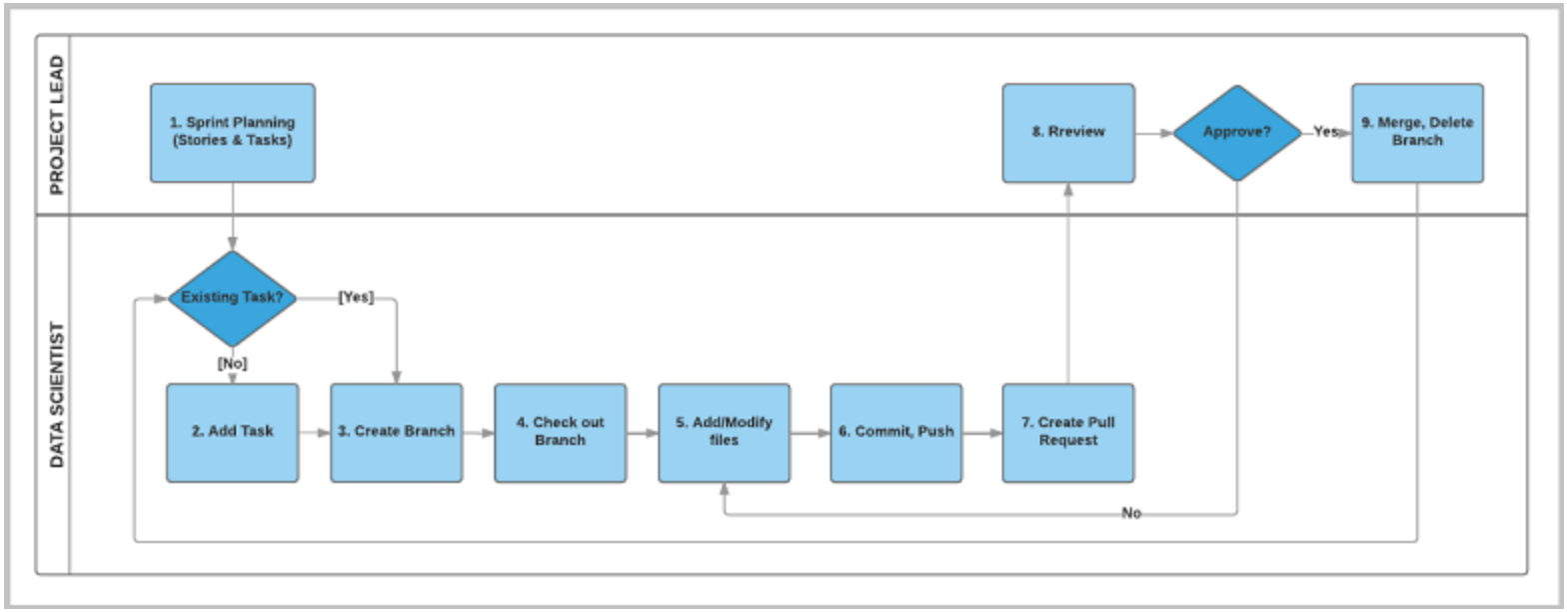
- Sprint planning can be done via Azure Boards.



Ref: https://docs.microsoft.com/en-us/azure/architecture/data-science-process/agile-development

Microsoft Team Data Science Project (TDSP): Work Item Types: Example

- The following figure outlines the TDSP workflow for project execution:



The workflow steps can be grouped into three activities:
•Project Leads conduct sprint planning
•Data Scientists develop artifacts on git branches to address work items
•Project Leads or other team members do code reviews and merge working branches to the primary branch

Ref: https://docs.microsoft.com/en-us/azure/architecture/data-science-process/agile-development

DI501 Introduction to Data Science

Lecture 2 – Part2 Platforms

Data Science Platforms

- A data science platform is a software hub around which all data science work takes place.
- That work usually includes integrating and exploring data from various sources, coding and building models that leverage that data, deploying those models into production, and serving up results, whether that's through model-powered applications or reports.
- A recent study conducted by Markets and Markets predicts the market for data science platforms will climb to $101.4 billion by the year 2021

Data Science Platforms

- A data science platform is a software hub around which
- That w oring data f models that leverage that data, deploying those mode whet ations or rep
- A recent study conducted by MarketsandMarkets predicts the market for data science platforms will climb to $101.4 billion by the year 2021.

If you have no idea how many models you have, or you're spending a lot of time maintaining your models once deployed, then you need a platform!

Good platforms can create logical workflows and facilitate integrations as well as give you version controls.

Version Control & Issue Tracking

- All artifacts can be stored in version control systems such as Git and Subversion
- Version control systems
  - allow keeping track of the artifacts created,
  - facilitate managing changes to artifacts over time,
  - enable team collaboration
- Tasks can be tracked by means of issue tracking systems like ***Jira***
  - They facilitate collaboration in large or distributed teams, resource allocation, priority management, monitoring progress

Commonly Used Programming Languages

- Python: An interpreted language that makes the development quicker and enjoyable. Supported by an enormous variety of libraries, doing everything from scraping to visualization to linear algebra and machine learning.
- R: Widely used by statisticians. It has powerful libraries for data analysis and visualization.
  - Linkages exist between R and Python, it is possible to call R library functions in Python code, and vice versa.
- Julia: A general purpose programming language. It is well-suited for high performance numerical analysis and computational science.

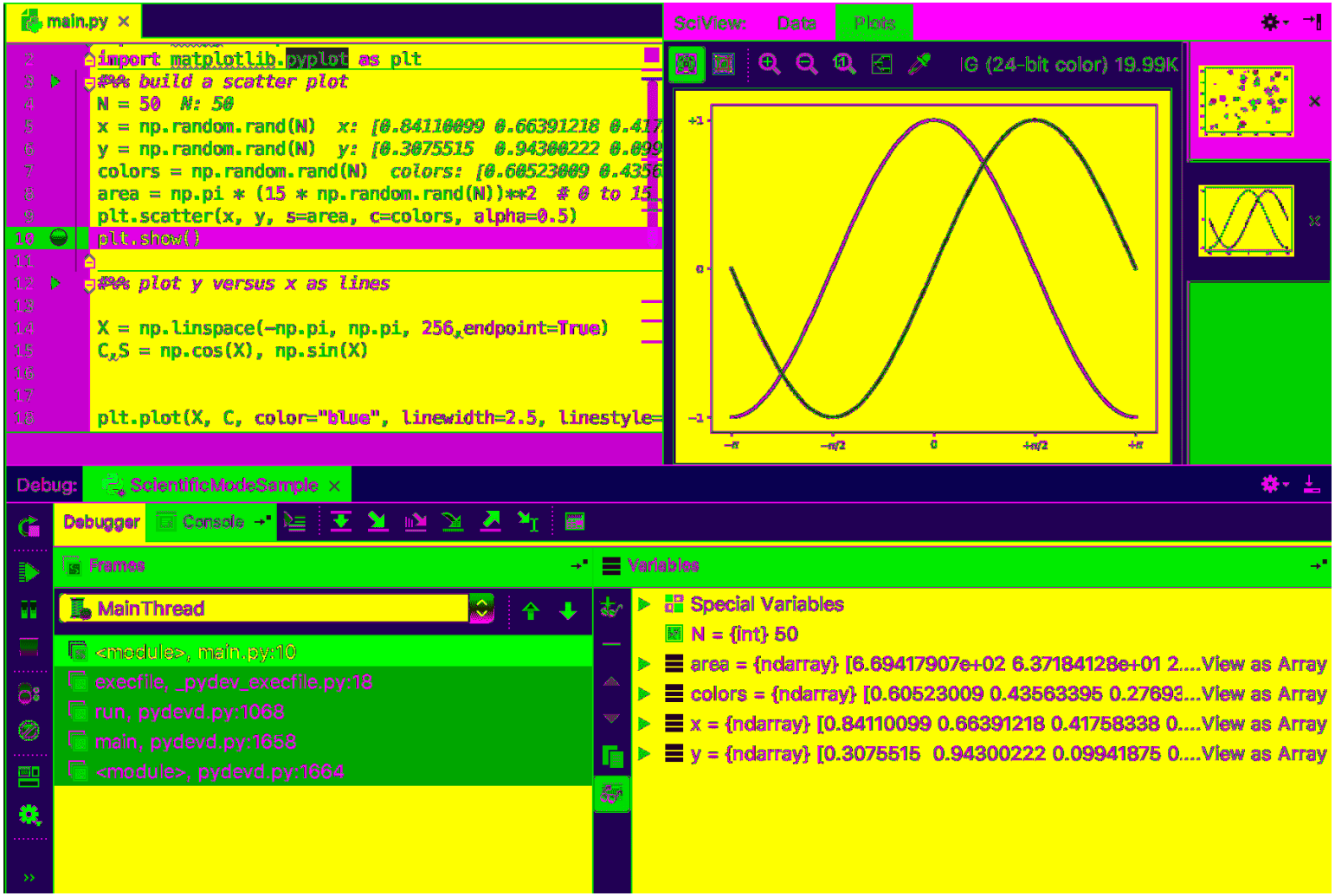
Commonly Used Programming Languages

- Java, Scala and C/C++: These mainstream programming languages for the development of large systems. Some Big Data platforms support Java and Scala languages.
- Matlab: Designed for the fast and efficient manipulation of matrices and solving numerical problems (many machine learning algorithms require these).
  - Matlab is a proprietary system. Much of its functionality is available in an open source alternative, GNU Octave.
- SQL: Designed for managing data held in relational database management systems. It is at least needed to retrieve data from databases.

Integrated Development Environments: PyCharm

- Powerful, general purpose Python IDE.
- Has both open source and professional editions, provides free license for academic work.
- With a feature called scientific mode in the professional edition, turns into a data science IDE supporting various built-in visualization features in the SciView.

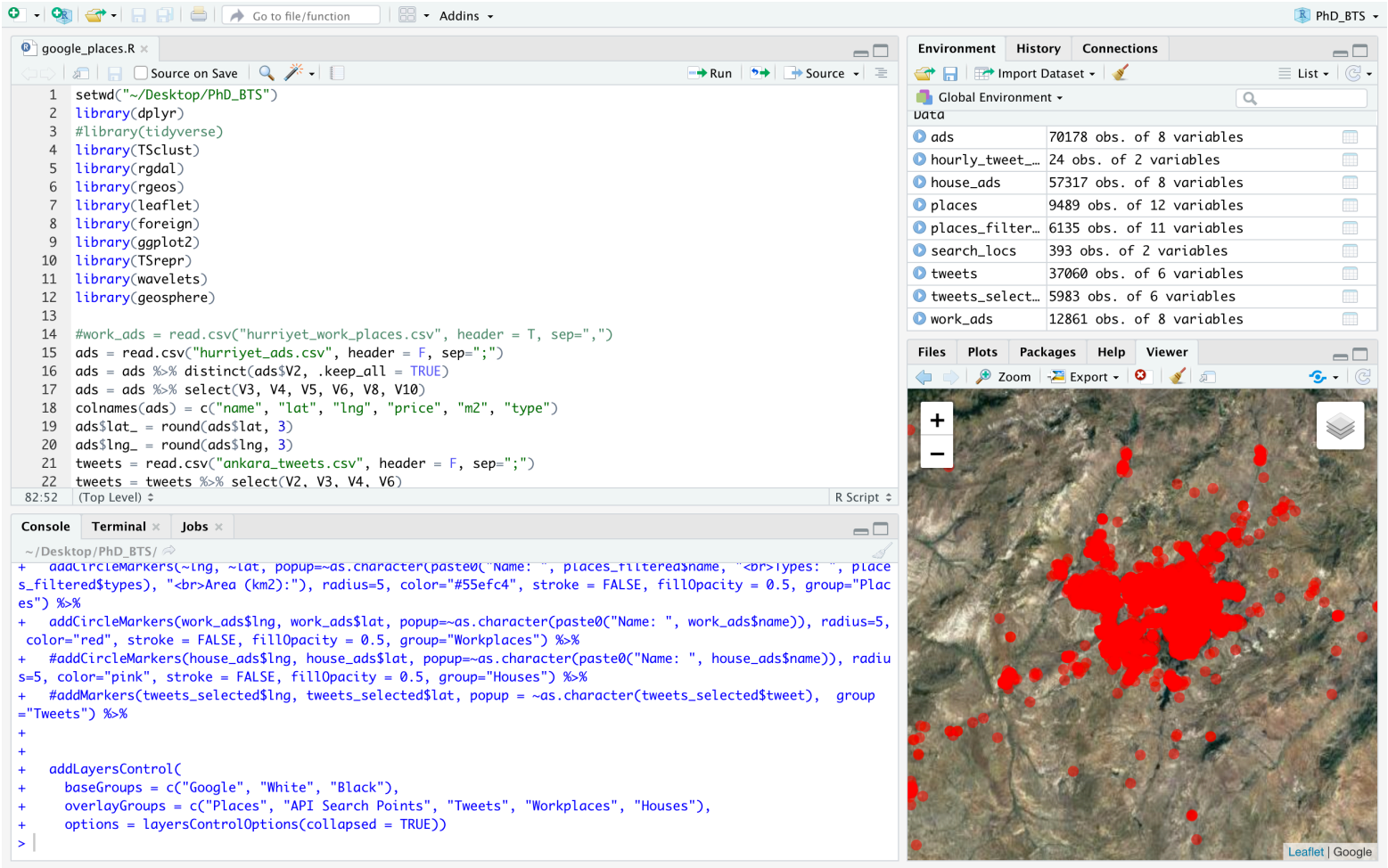
PyCharm IDE


main.py
import matplotlib.pyplot as plt
%% build a scatter plot
N = 50 N: 50
x = np.random.rand(N) x: [0.84110099 0.66391218 0.417
y = np.random.rand(N) y: [0.3075515 0.94300222 0.099
colors = np.random.rand(N) colors: [0.60523009 0.4356
area = np.pi * (15 * np.random.rand(N))**2 # 0 to 15
plt.scatter(x, y, s=area, c=colors, alpha=0.5)
plt.show()
%% plot y versus x as lines
X = np.linspace(-np.pi, np.pi, 256,endpoint=True)
C,S = np.cos(X), np.sin(X)
plt.plot(X, C, color="blue", linewidth=2.5, linestyle=
SciView
Data
Plots
IG (24-bit color) 19.99K
Debug:
ScientificModeSample
Debugger
Console
Frames
Variables
MainThread
<module>, main.py:10
execfile, _pydev_execfile.py:18
run, pydevd.py:1068
main, pydevd.py:1658
<module>, pydevd.py:1664
Special Variables
N = {int} 50
area = {ndarray} [6.69417907e+02 6.37184128e+01 2....View as Array
colors = {ndarray} [0.60523009 0.43563395 0.27693...View as Array
x = {ndarray} [0.84110099 0.66391218 0.41758338 0....View as Array
y = {ndarray} [0.3075515 0.94300222 0.09941875 0....View as Array

Integrated Development Environments: RStudio

- Integrated development environment for R language.
- Includes a console, syntax-highlighting editor that supports direct code execution, as well as tools for plotting, history, debugging and workspace management.

RStudio IDE

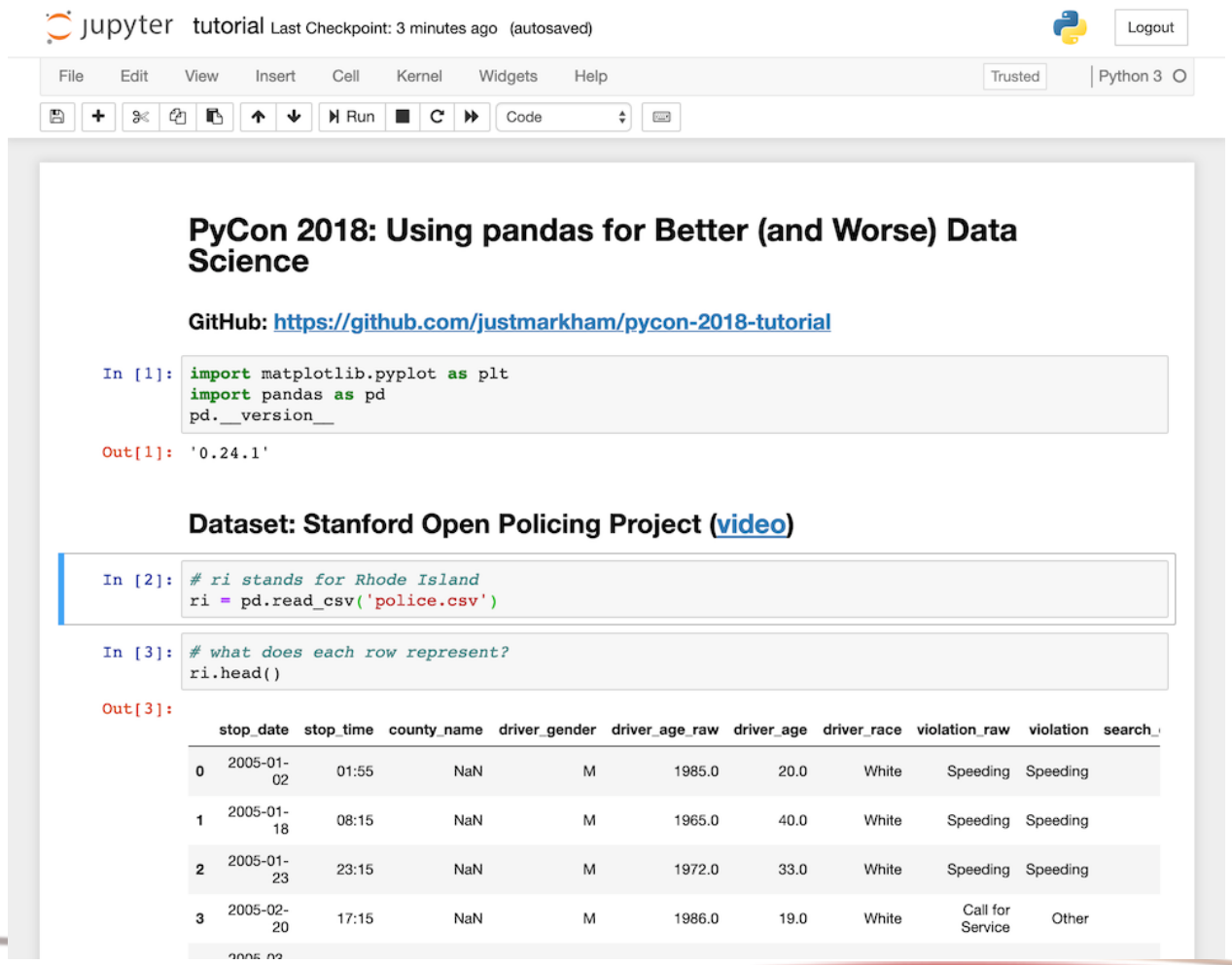

Go to file/function
Addins
PhD_BTS
google_places.R
Source on Save
Run
Source
setwd("~/Desktop/PhD_BTS")
library(dplyr)
#library(tidyverse)
library(TSclust)
library(rgdal)
library(rgeos)
library(leaflet)
library(foreign)
library(ggplot2)
library(TSrepr)
library(wavelets)
library(geosphere)
#work_ads = read.csv("hurriyet_work_places.csv", header = T, sep=",")
ads = read.csv("hurriyet_ads.csv", header = F, sep=";")
ads = ads %>% distinct(ads$V2, .keep_all = TRUE)
ads = ads %>% select(V3, V4, V5, V6, V8, V10)
colnames(ads) = c("name", "lat", "lng", "price", "m2", "type")
ads$lat_ = round(ads$lat, 3)
ads$lng_ = round(ads$lng, 3)
tweets = read.csv("ankara_tweets.csv", header = F, sep=";")
tweets = tweets %>% select(V2, V3, V4, V6)
82:52
(Top Level)
R Script
Console
Terminal
Jobs
~/Desktop/PhD_BTS/
addCircleMarkers(~lng, ~lat, popup=~as.character(paste0("Name: ", places_filtered$name, "
Types: ", places_filtered$types), "
Area (km2):"), radius=5, color="#55efc4", stroke = FALSE, fillOpacity = 0.5, group="Places") %>%
addCircleMarkers(work_ads$lng, work_ads$lat, popup=~as.character(paste0("Name: ", work_ads$name)), radius=5, color="red", stroke = FALSE, fillOpacity = 0.5, group="Workplaces") %>%
#addCircleMarkers(house_ads$lng, house_ads$lat, popup=~as.character(paste0("Name: ", house_ads$name)), radius=5, color="pink", stroke = FALSE, fillOpacity = 0.5, group="Houses") %>%
#addMarkers(tweets_selected$lng, tweets_selected$lat, popup = ~as.character(tweets_selected$tweet), group="Tweets") %>%
addLayersControl(
baseGroups = c("Google", "White", "Black"),
overlayGroups = c("Places", "API Search Points", "Tweets", "Workplaces", "Houses"),
options = layersControlOptions(collapsed = TRUE))
Environment
History
Connections
Import Dataset
List
Global Environment
Data
ads 70178 obs. of 8 variables
hourly_tweet_... 24 obs. of 2 variables
house_ads 57317 obs. of 8 variables
places 9489 obs. of 12 variables
places_filter... 6135 obs. of 11 variables
search_locs 393 obs. of 2 variables
tweets 37060 obs. of 6 variables
tweets_select... 5983 obs. of 6 variables
work_ads 12861 obs. of 8 variables
Files
Plots
Packages
Help
Viewer
Zoom
Export
Leaflet | Google

- Literate programming is a software development style pioneered by Stanford computer scientist, Donald Knuth.
- This type of programming emphasizes a prose first approach where exposition with human-friendly text is punctuated with code blocks.
- Literate programming allows users to formulate and describe their thoughts with prose, supplemented by mathematical equations, as they prepare to write code blocks.
- All data science notebooks are literate programming tools.

- An open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text.
- Use purposes: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning.

Data Science Notebooks

Jupyter Notebook



	stop_date	stop_time	county_name	driver_gender	driver_age_raw	driver_age	driver_race	violation_raw	violation	search_
0	2005-01-02	01:55	NaN	M	1985.0	20.0	White	Speeding	Speeding	
1	2005-01-18	08:15	NaN	M	1965.0	40.0	White	Speeding	Speeding	
2	2005-01-23	23:15	NaN	M	1972.0	33.0	White	Speeding	Speeding	
3	2005-02-20	17:15	NaN	M	1986.0	19.0	White	Call for Service	Other	

Data Science Notebooks

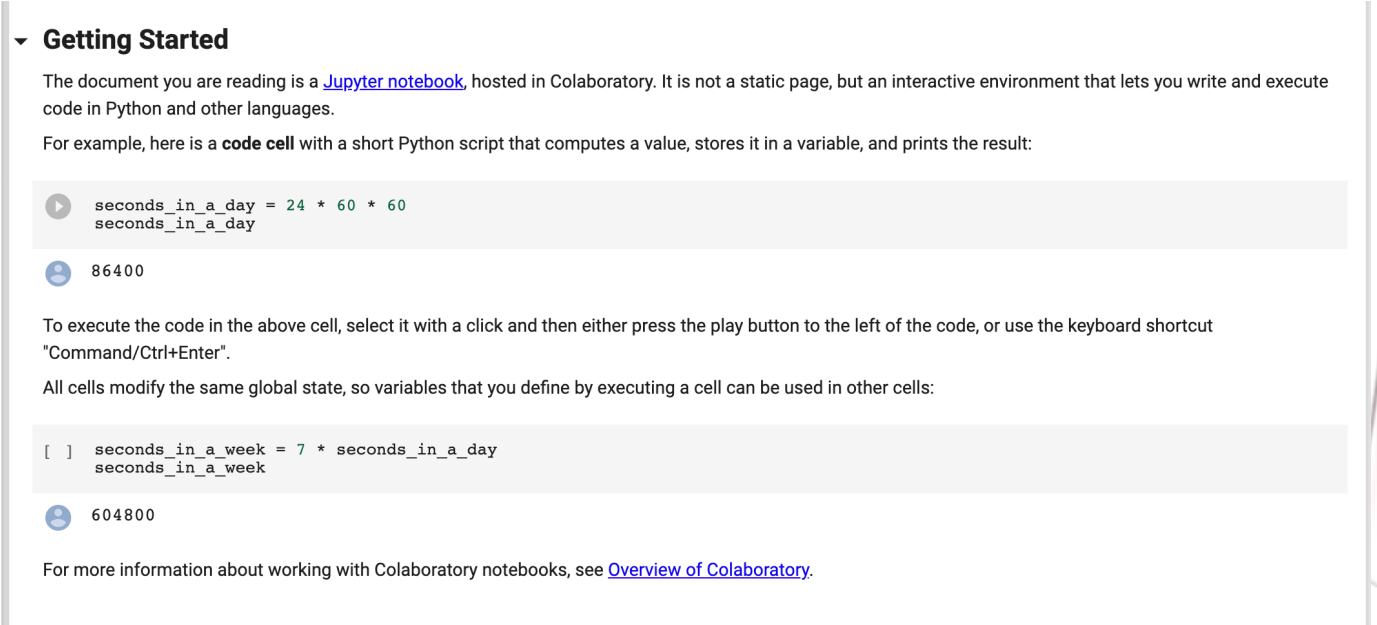
Jupyter Notebook

- They have LaTeX support for mathematical equations with MathJax, a web browser enhancement for display of mathematics.
- These notebooks can be saved and easily shared in .ipynb JSON format.
- They can also be committed to version control repositories such as git and the code sharing site GitHub.
- “Jupyter” is a loose acronym meaning Julia, Python and R, but today, the notebook technology supports many programming languages.

Data Science Notebooks

Google Colaboratory

- Colaboratory is a free Jupyter notebook environment that requires no setup and runs entirely in the cloud.
- With Colaboratory you can write and execute code, save and share your analyses, and access powerful computing resources, all for free from your browser.



Getting Started

The document you are reading is a Jupyter notebook, hosted in Colaboratory. It is not a static page, but an interactive environment that lets you write and execute code in Python and other languages.

For example, here is a **code cell** with a short Python script that computes a value, stores it in a variable, and prints the result:

```
seconds_in_a_day = 24 * 60 * 60
seconds_in_a_day
```

```
86400
```

To execute the code in the above cell, select it with a click and then either press the play button to the left of the code, or use the keyboard shortcut "Command/Ctrl+Enter".

All cells modify the same global state, so variables that you define by executing a cell can be used in other cells:

```
[ ]  seconds_in_a_week = 7 * seconds_in_a_day
     seconds_in_a_week
```

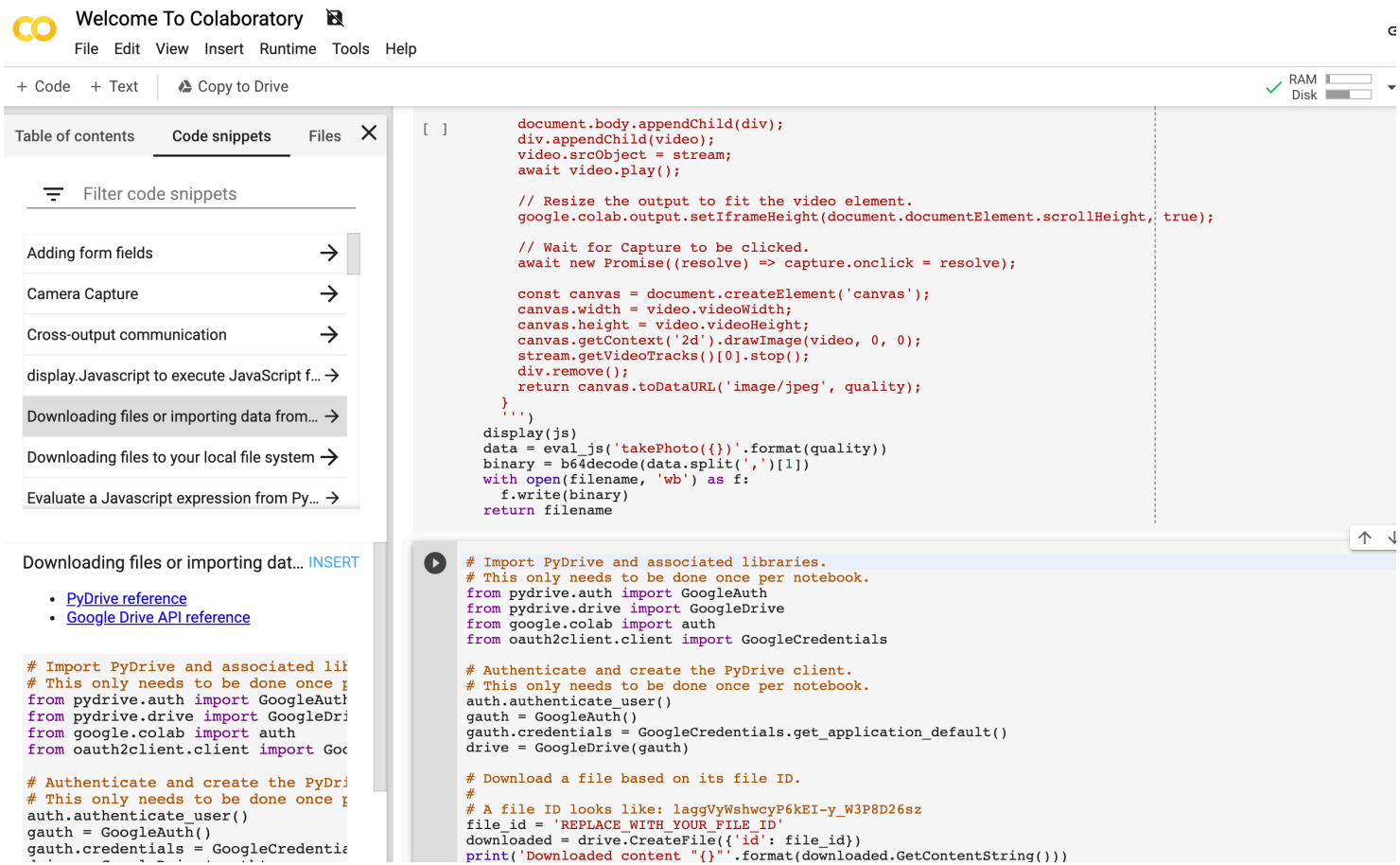
```
604800
```

For more information about working with Colaboratory notebooks, see Overview of Colaboratory.

https://colab.research.google.com/notebooks/welcome.ipynb#scrollTo=5fCEDCU_qrC0

Data Science Notebooks

Google Colaboratory



https://colab.research.google.com/notebooks/welcome.ipynb#scrollTo=5fCEDCU_qrC0

Data Science Notebooks

Azure Notebooks

- Hosted Jupyter Notebook alternative.
- Supports Python, R, F languages
- Backend runs on Azure Cloud.

Data Science Notebooks

Azure Notebooks

Microsoft Azure Notebooks Preview My Projects Help makyol

Powered by jupyter assignment-1 Last Checkpoint: 12 hours ago (unsaved changes) Assignment-1

File Edit View Insert Cell Kernel Widgets Help Trusted R

Run Code Enter/Exit RISE Slideshow Autosave interval (min): 2 (default)

Load a csv file

In [1]:
```
data = read.csv("OSMI Mental Health in Tech Survey 2018.csv", header = T, sep = ",")
```

Install R packages

In [3]:
```
if (!is.element("dplyr", installed.packages()[,1]))
    install.packages("dplyr", repos="http://cran.r-project.org")
```

In [4]:
```
if (!is.element("leaflet", installed.packages()[,1]))
    install.packages("leaflet", repos="http://cran.r-project.org")
```

Load R packages

In [5]:
```
library(dplyr)
```

...

In [6]:
```
library(leaflet)
```

In [8]:
```
leaflet() %>% addTiles() %>% addMarkers(lng=174.768, lat=-36.852, popup="The birthplace of R")
```

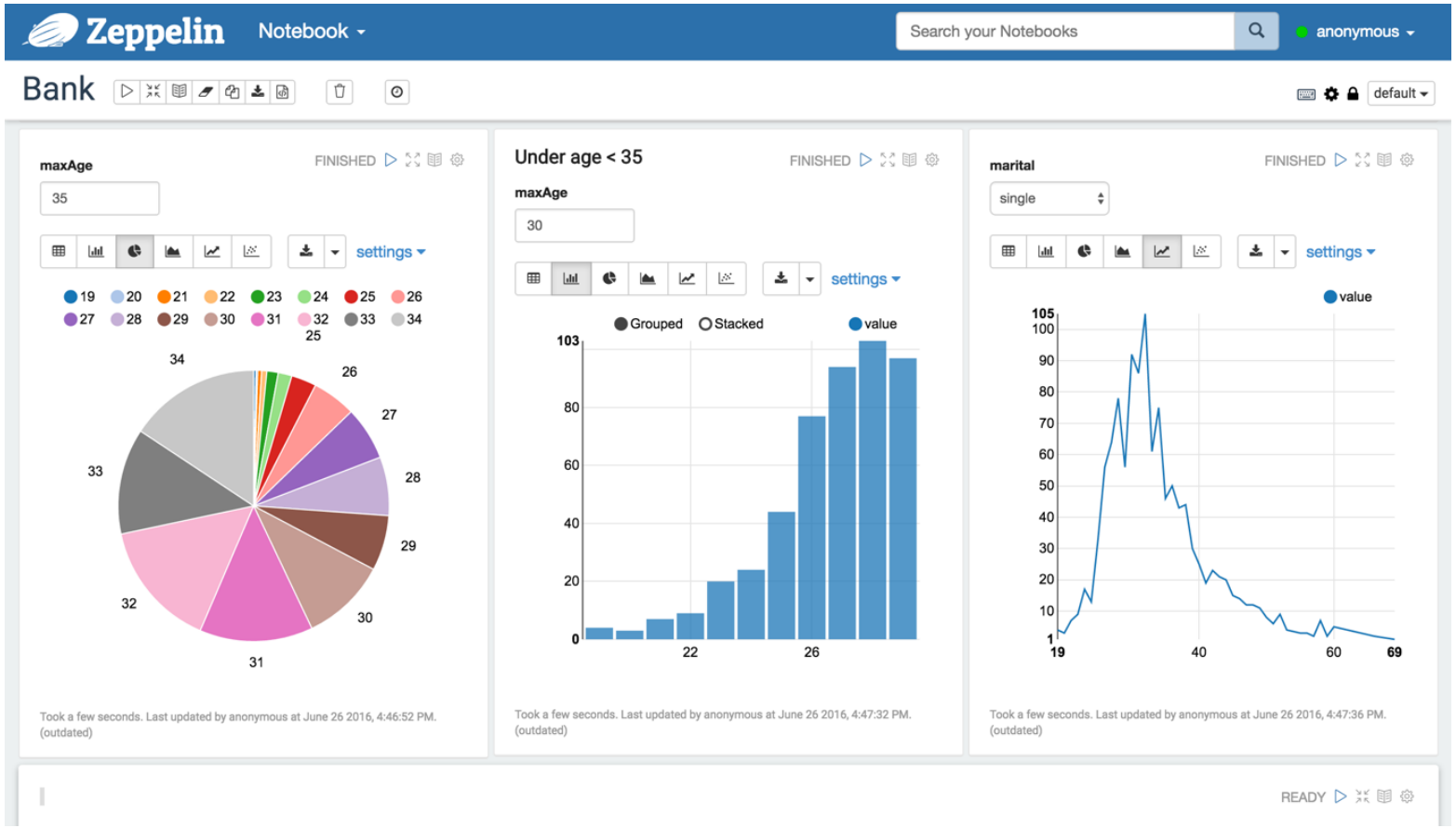
Data Science Notebooks

Apache Zeppelin Notebooks

- Web-based notebook that enables data-driven, interactive data analytics and collaborative documents with SQL, Scala and more.
- Other than enabling you to write vanilla code in R, Python, Scala, and SQL; integrates with various open source big data tools such as Flink, Spark, Cassandra and HBase.

Data Science Notebooks

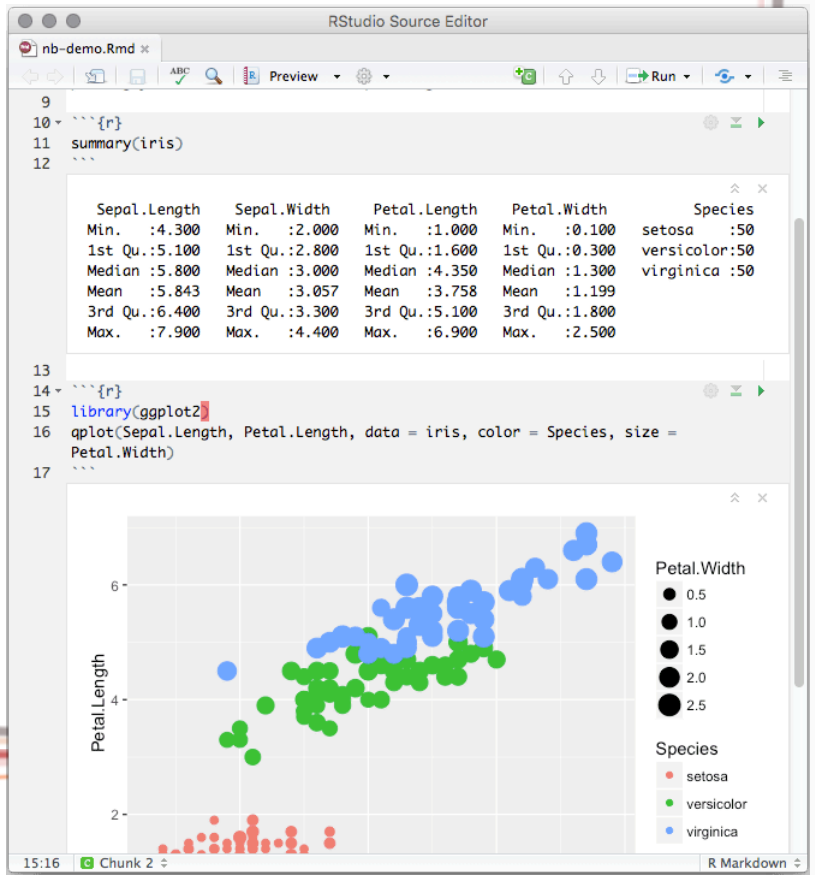
Apache Zeppelin Notebooks



Data Science Notebooks

R Notebooks

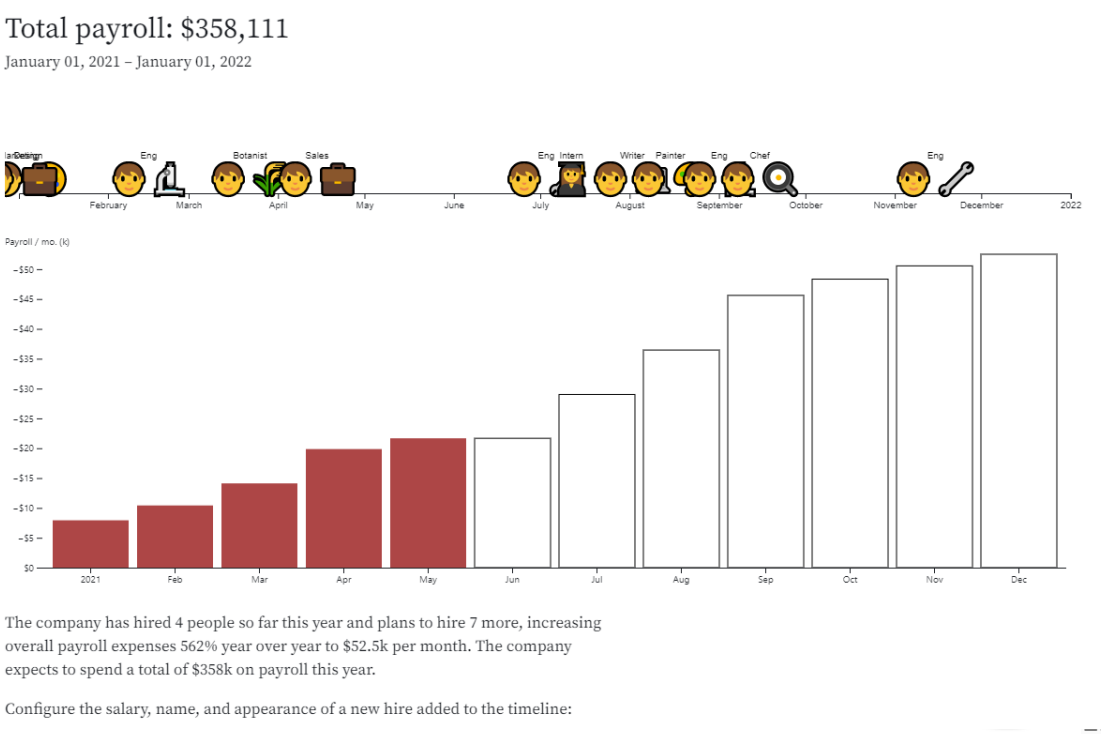
- An R Notebook is an R Markdown document with chunks that can be executed independently and interactively, with output visible immediately beneath the input.



Data Science Notebooks

Observable

- An oversimplified description would be that this is a Jupyter Notebook specifically for JavaScript.
- Libraries like D3 are pre-loaded and immediately accessible.
  - D3.js is a JavaScript Library for manipulating documents based on data – it provides powerful visualization for your data using HTML, SVG, and CSS.



https://observablehq.com/@observablehq/hiring-timeline

Data Science Notebooks

Observable

```
3
```

```
1 + 2 // Edit me!
```

For more complex definitions such as loops, hug your code with curly braces {…}.
Whatever value you return is shown.

```
4950
```

```
{
  let sum = 0;
  for (let i = 0; i < 100; ++i) {
    sum += i;
  }
  return sum;
}
```

```
data = ▶ Array(12) [Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object]
```

```
data = (await FileAttachment("payroll.csv").csv()).map(autoTypeExcel)
```

id	date	name	salary	emoji
2	2021-02-01	Sales	50,000	🧑💼
3	2021-02-15	Eng	50,000	🧑🔬
4	2021-03-15	Design	50,000	🧑🎨
5	2021-04-15	Marketing	50,000	🧑💼
6	2021-06-01	Eng	70,000	🧑🔧
7	2021-06-15	Intern	30,000	🤵
8	2021-08-01	Writer	50,000	🧑💻
9	2021-08-15	Painter	50,000	🧑🎨
10	2021-09-01	Eng	50,000	🧑🔬
11	2021-09-15	Chef	70,000	🧑🔍
12	2021-11-15	Eng	50,000	🧑🔧

https://observablehq.com/

Data Science Notebooks

Observable

You can load data by attaching files or using the Fetch API.

```
cars = ▶ Array(406) [Object, Object, Object, Object, Object, Object, Object, Object, Object, Object, Object,
```

```
cars = fetch("https://raw.githubusercontent.com/vega/vega/v4.3.0/docs/data/cars.json")
  .then(response => response.json())
```

You can yield DOM elements for animation. Click the play button ▷ in the top-right corner of the cell below to restart the animation.

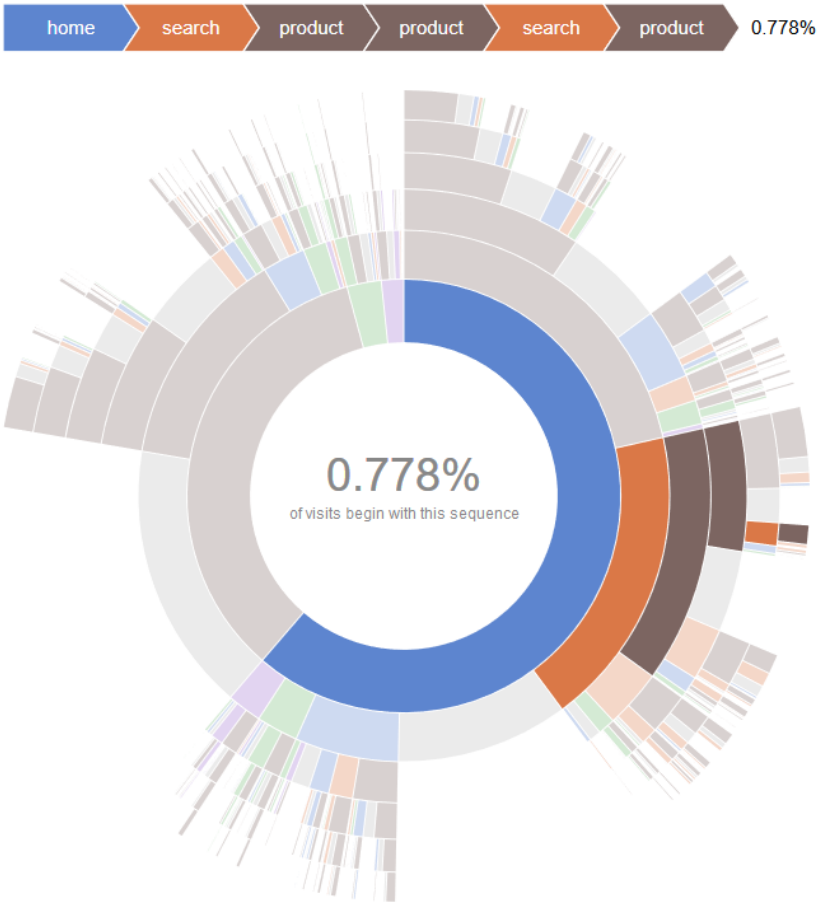
Run cell (shift-enter)

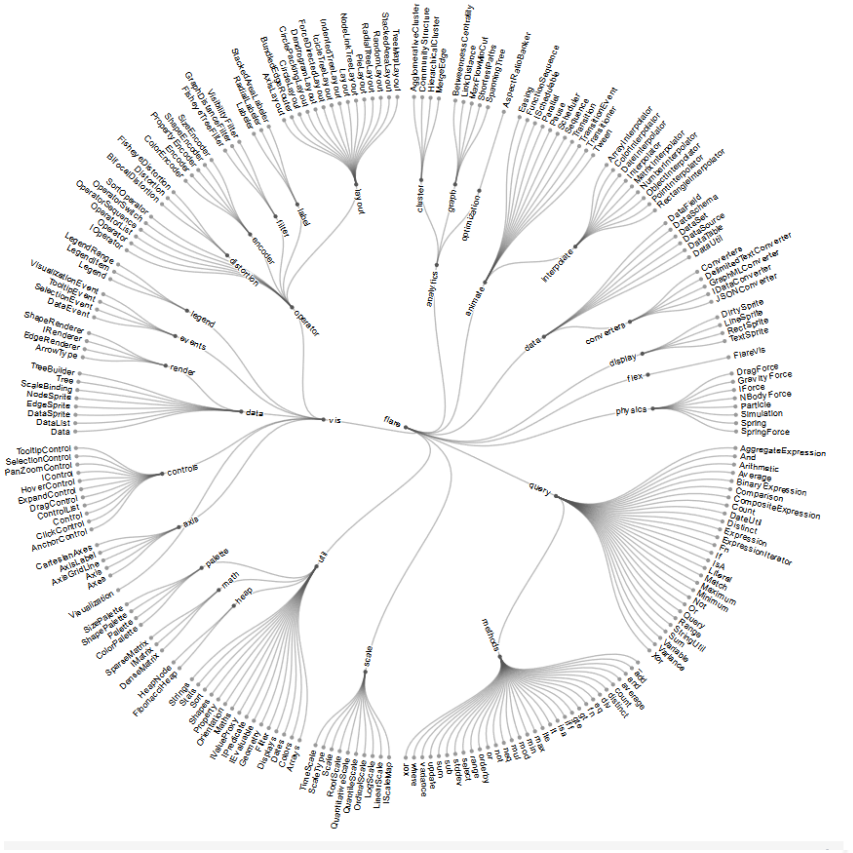
```
{
  const height = 33;
  const context = DOM.context2d(width, height);
  for (let i = 0; i < width; ++i) {
    const t = i / width;
    const r = Math.floor(255 * Math.sin(Math.PI * (t + 0 / 3)) ** 2);
    const g = Math.floor(255 * Math.sin(Math.PI * (t + 1 / 3)) ** 2);
    const b = Math.floor(255 * Math.sin(Math.PI * (t + 2 / 3)) ** 2);
    context.fillStyle = `rgb(${r},${g},${b})`;
    context.fillRect(i, 0, 1, height);
    yield context.canvas;
  }
}
```

https://observablehq.com/

Data Science Notebooks

Observable, D3.js





https://observablehq.com/@d3/gallery

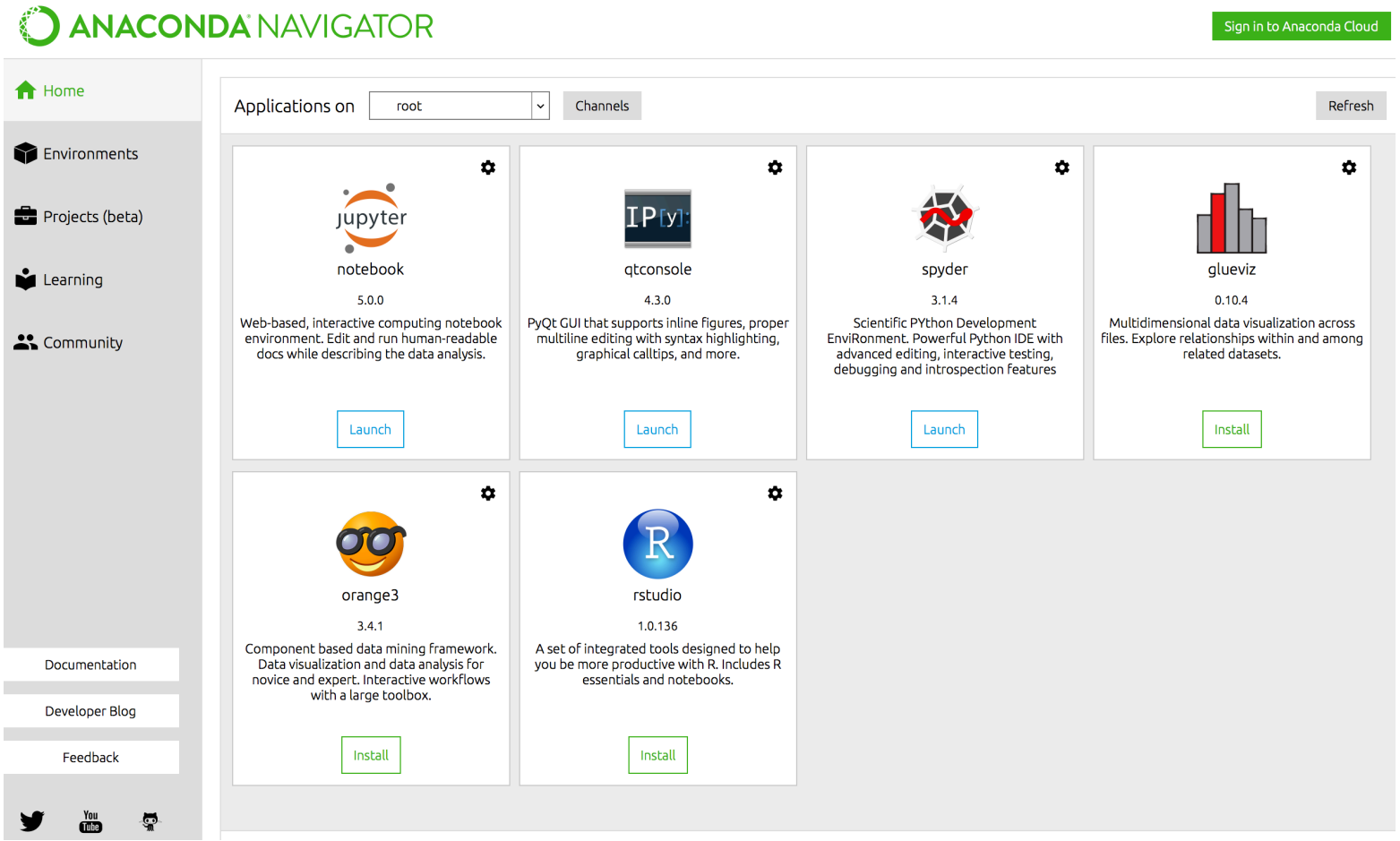
Platforms & Workbenches

Anaconda

- Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.)
- Aims to simplify package management and deployment.
- Helps you easily install both Python libraries and R packages.
- Comes with different tools like Jupyter Notebook, Rstudio, and Spyder (Python IDE, alternative to PyCharm)

Platforms & Workbenches

Anaconda



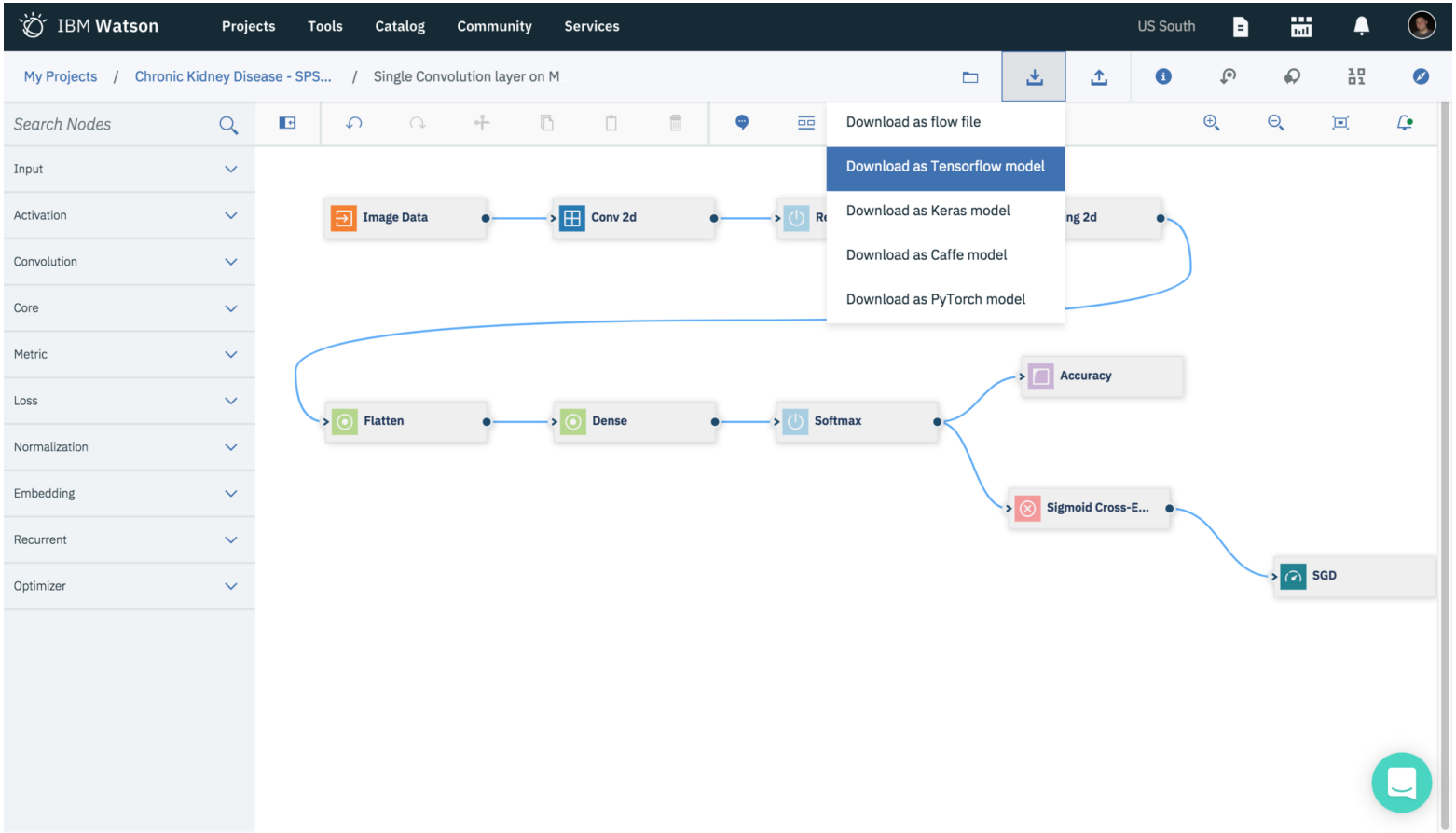
Platforms & Workbenches

IBM Watson Studio

- A suite of tools for data scientists, application developers and subject matter experts, allowing them to collaboratively connect to data, wrangle that data and use it to build, train and deploy models at scale.
- Provides both open source and IBM tools to create end-to-end data science project workflow.
- Has a no-code and drag&drop visual modelling feature.
- Supports various ML/DL libraries such as Spark MLlib, Scikit learn, SPSS modeller, Torch, Caffe, Tensorflow, and Keras.

Platforms & Workbenches

IBM Watson Studio



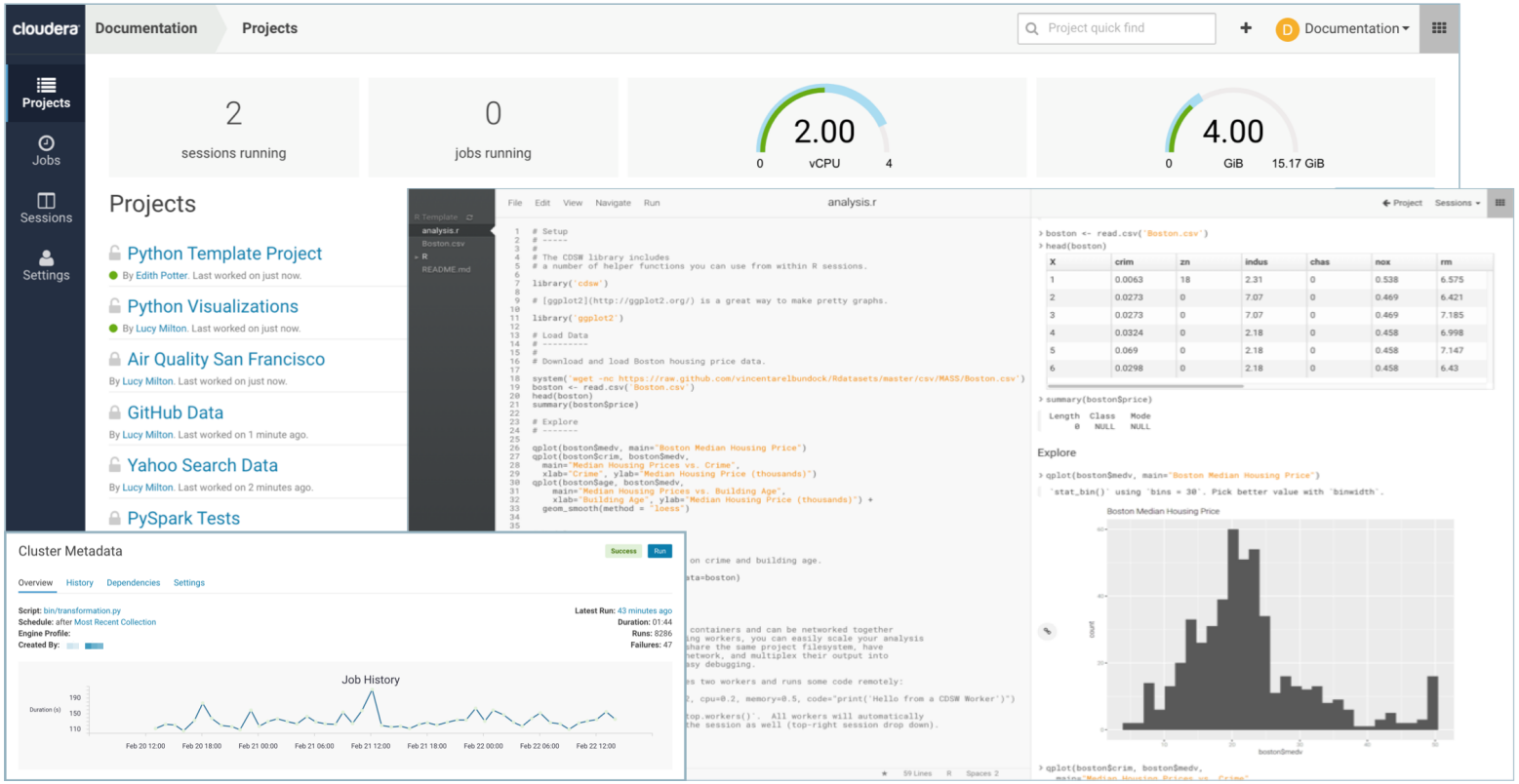
Platforms & Workbenches

Cloudera Data Science Workbench

- An application that empowers data scientists to use their preferred technologies, languages and libraries in an environment that can run on your local computer or on the cloud.
- Runs on top of the Hadoop/Spark clusters, helps you distribute your data science projects.
- Enables you to easily deploy your models.

Platforms & Workbenches

Cloudera Data Science Workbench



Platforms & Workbenches

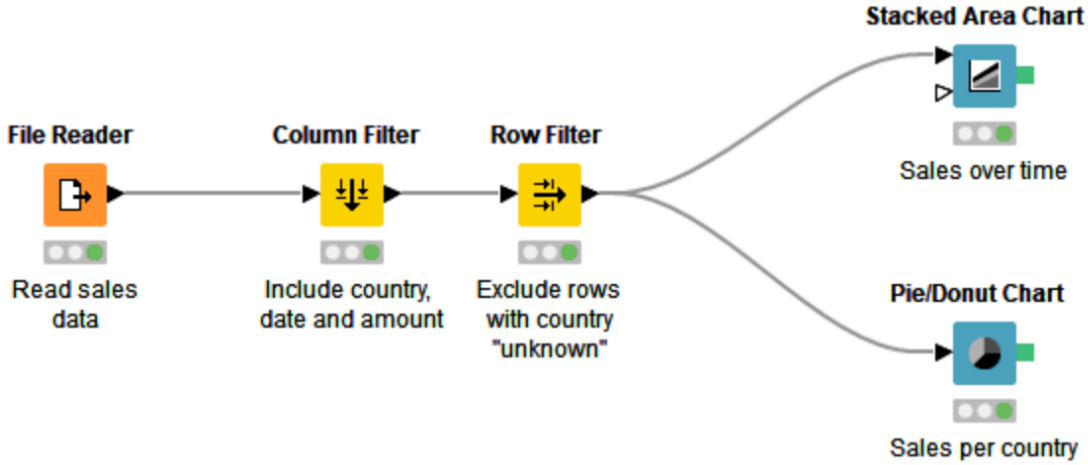
KNIME Analytics Platform

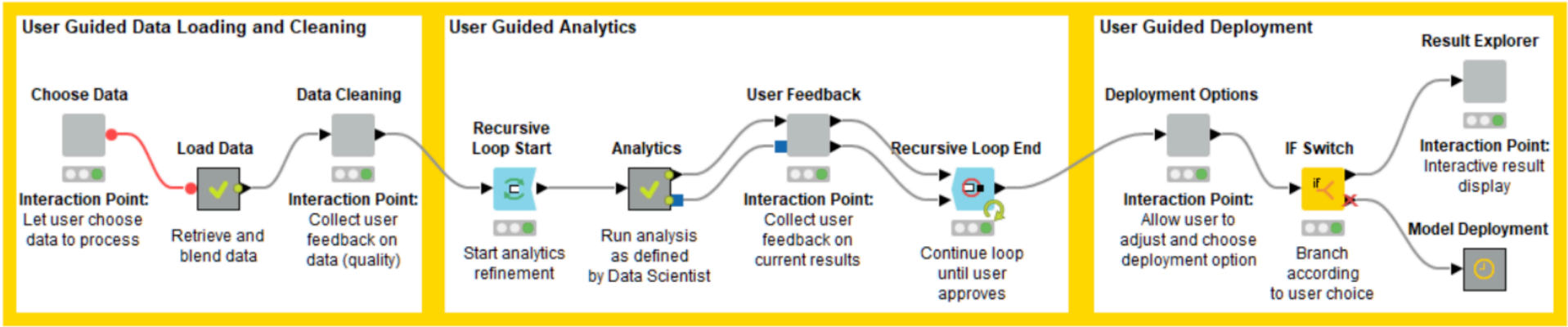
- KNIME Analytics Platform is open source software for creating data science applications and services. Intuitive, open, and continuously integrating new developments, KNIME makes understanding data and designing data science workflows and reusable components accessible to everyone.
- With KNIME Analytics Platform, you can create visual workflows with an intuitive, drag and drop style graphical interface, without the need for coding.

https://docs.knime.com/latest/analytics_platform_quickstart_guide/index.html

Platforms & Workbenches

KNIME Analytics Platform





https://docs.knime.com/latest/analytics_platform_quickstart_guide/index.html

Platforms & Workbenches

KEEL (Knowledge Extraction based on Evolutionary Learning)

- KEEL is an open source (GPLv3) Java software tool that can be used for a large number of different knowledge data discovery tasks.
- KEEL provides a simple GUI based on data flow to design experiments with different datasets and computational intelligence algorithms (paying special attention to evolutionary algorithms) in order to assess the behavior of the algorithms.

https://sci2s.ugr.es/keel/

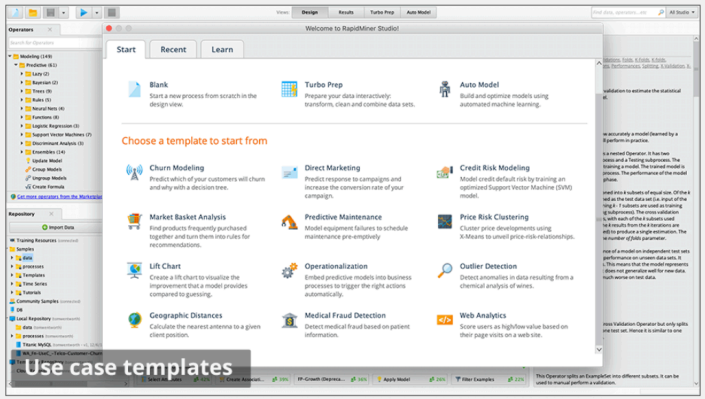
Platforms & Workbenches

KEEL (Knowledge Extraction based on Evolutionary Learning)

- KEEL has extremely extensive support for discretization algorithms,
  - but has limited support for other methods for engineering new features out of existing features.
- It has excellent support for feature selection, with a wider range of algorithms than any other package.
- It also has extensive support for imputation of missing data, and considerable support for data re-sampling.

Platforms & Workbenches

RapidMiner



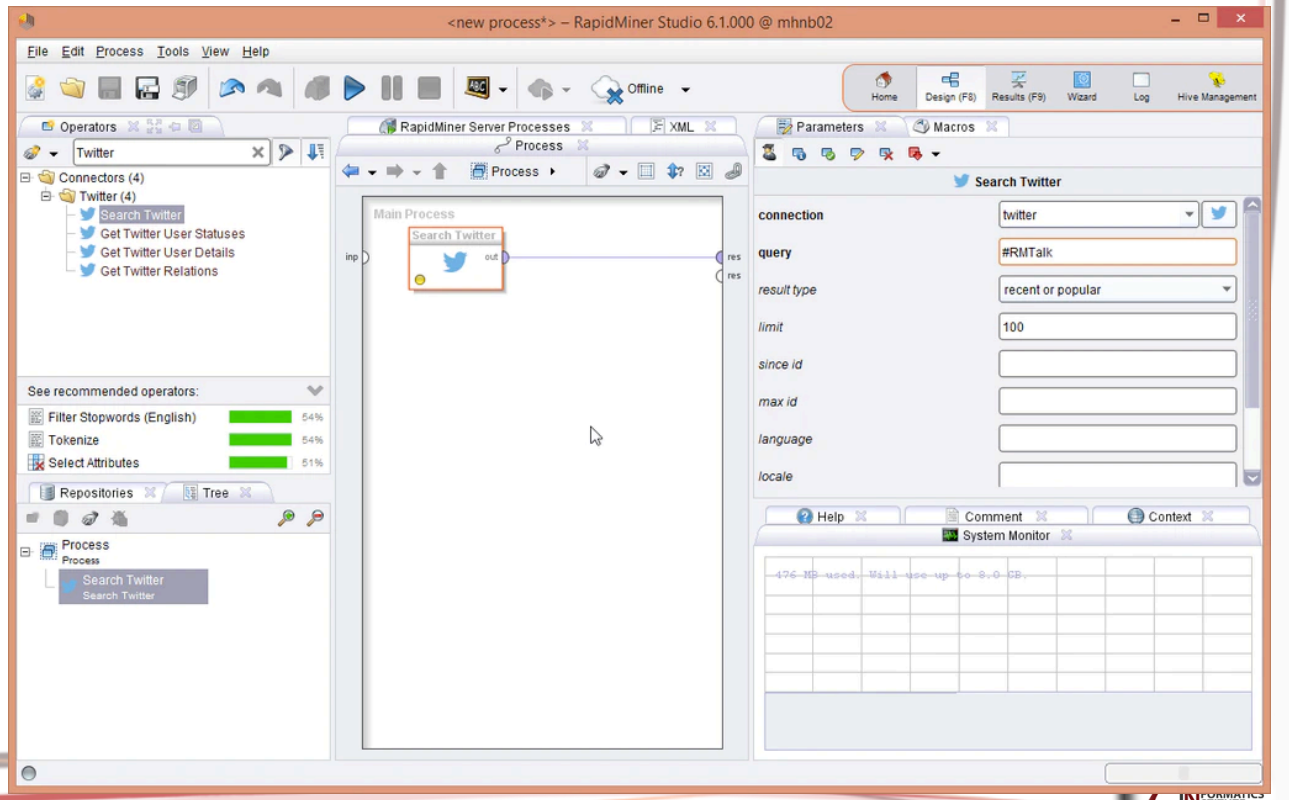
- It provides several features:
  - Visual workflow designer
  - Connecting to any data source
  - Automated in database processing
    - Run data prep and ETL inside databases to keep your data optimized for advanced analytics
  - Data visualization and exploration
  - Data preparation and blending
  - Visual and automated machine learning
  - Model validation
  - R and Python codes can be integrated
  - Automation and process control (scheduling)

https://rapidminer.com/products/studio/

Platforms & Workbenches

RapidMiner

- Free license provides a very limited availability
  - RapidMiner Studio: Processing 10,000 data rows, 30 day trial of enterprise

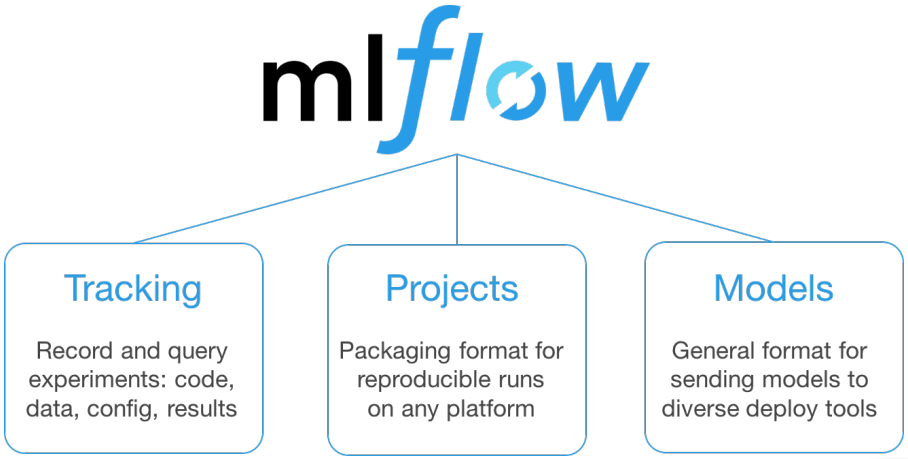


https://rapidminer.com/products/studio/

Platforms & Workbenches

DataBricks

- Databricks Unified Analytics Platform, from the original creators of Apache Spark™, unifies data science and engineering across the Machine Learning lifecycle from data preparation, to experimentation and deployment of ML applications.



https://databricks.com/

Platforms & Workbenches

Other

- There are also alternatives to aforementioned ones, such as:
  - TIBCO
  - H20.ai
  - WEKA
  - MATLAB
- When choosing a platform:
  - Your platform needs to conform to the needs of your organization or business.
  - Consider the licensing scheme!
- Your data science platform should be:
  - flexible
  - supportive
  - inclusive of all the tools (and team members) you need
  - integrative
  - It should foster collaboration and encourage each member of the team to produce high-quality work.

Platforms & Workbenches

Comparison

Table 1 Summary of characteristics of software tools

Software	Open source	language	Run Type		Graphical Interface			Input / output		
			On-line	Off-line	Graph Representation	Data visualization	Data management	ARFF data format	Other data format	Data base connection
KEEL	Y	Java	Y	Y	Y	Y	Y	Y	Y	Y
KNIME	Y	Java	Y	N	Y	A	A	Y	Y	Y
WEKA	Y	Java	Y	N	Y	A	A	Y	Y	Y
ORANGE	Y	C/C++/ Python	N	Y	Y	A	A	N	Y	N
RAPIDMINER	Y	Java	Y	N	N	A	A	Y	Y	Y
AZURE MACHINE	Y	–	Y	Y	Y	Y	A	Y	Y	Y
IBM SPSS MODELER	N	Java	N	Y	Y	Y	Y	N	Y	Y
R	Y	C / Fortran/ R	Y	Y	Y	Y	Y	Y	Y	Y
SCIKIT-LEARN	Y	Python	Y	Y	Y	Y	Y	Y	Y	Y

None (N), basic support (B), intermediate support (I) and advanced support (A).
The notation Yes (Y) is used for supporting, and No (N) is used for no-supporting, if characteristics do not have intermediate levels of support. The (+) specifies that the tools implement the algorithm, apply an external add-on (A) to support it; (S) indicates some degree of support for the method, or do not (−). Since most tools are upgraded in a constant state, the data in Tables should be considered temporarily.

Ref: Hosseini, S., & Sardo, S. R. (2020). Data mining tools-a case study for network intrusion detection. *Multimedia Tools and Applications*, 1-21.

Platforms & Workbenches

Comparison

Table 2 Summary of characteristics software tools: Pre-processing variety

Pre-processing variety					
Software	Discretization	Feature Selection	instance selection	Training Set Select	Missing Value Imputation
KEEL	A	A	A	A	A
KNIME	I	A	B	N	B
WEKA	A/I	I/A	B	N	B
ORANGE	A	I	B	N	B
RAPIDMINER	I	A	B	A	B
AZURE MACHINE	B	A	A	N	A
IBM SPSS MODELER	B	A	A	A	A
R	N	N	N	N	B
SCIKIT-LEARN	A	A	B	A	A

None (N), basic support (B), intermediate support (I) and advanced support (A).
The notation Yes (Y) is used for supporting, and No (N) is used for no-supporting, if characteristics do not have intermediate levels of support. The (+) specifies that the tools implement the algorithm, apply an external add-on (A) to support it; (S) indicates some degree of support for the method, or do not (−). Since most tools are upgraded in a constant state, the data in Tables should be considered temporarily.

Ref: Hosseini, S., & Sardo, S. R. (2020). Data mining tools-a case study for network intrusion detection. *Multimedia Tools and Applications*, 1-21.

Platforms & Workbenches

Comparison

Table 3 Summary of characteristics of software tools: learning variety

Learning Variety								
Software	Classification	Regression	Clustering	Association Rules	Subgroup Discovery	Imbalanced Classification	SSL	MIL
KEEL	A	A	A	A	A	A	A	I
KNIME	A	A	A	A	N	N	N	N
WEKA	A	A	A	A	N	N	N	I
ORANGE	I	N	I	I	N	N	N	N
RAPIDMINER	A	A	A	B	B	A	Y	N
AZURE MACHINE	A	A	B	N	A	A	N	N
IBM SPSS MODELER	A	A	A	A	N	N	A	N
R	A	A	A	B	I	B	I	N
SCIKIT-LEARN	A	A	A	B	I	B	N	N

SSL: Semi supervised learning, multiple instance learning (MIL)

None (N), basic support (B), intermediate support (I) and advanced support (A).
The notation Yes (Y) is used for supporting, and No (N) is used for no-supporting, if characteristics do not have intermediate levels of support. The (+) specifies that the tools implement the algorithm, apply an external add-on (A) to support it; (S) indicates some degree of support for the method, or do not (−). Since most tools are upgraded in a constant state, the data in Tables should be considered temporarily.

Ref: Hosseini, S., & Sardo, S. R. (2020). Data mining tools-a case study for network intrusion detection. *Multimedia Tools and Applications*, 1-21.

Platforms & Workbenches

Comparison

Table 3 Summary of characteristics of software tools: learning variety

Learning Variety								
Software	Classification	Regression	Clustering	Association Rules	Subgroup Discovery	Imbalanced Classification	SSL	MIL
KEEL	A	A	A	A	A	A	A	I
KNIME	A	A	A	A	N	N	N	N
WEKA	A	A	A	A	N	N	N	I
ORANGE	I	N	I	I	N	N	N	N
RAPIDMINER	A	A	A	B	B	A	Y	N
AZURE MACHINE	A	A	B	N	A	A	N	N
IBM SPSS MODELER	A	A	A	A	N	N	A	N
R	A	A	A	B	I	B	I	N
SCIKIT-LEARN	A	A	A	B	I	B	N	N

SSL: Semi supervised learning, multiple instance learning (MIL)

The tools change rapidly. For instance, SCIKIT-Learn now supports ”Label propagation” in semi supervised learning:
Ref: https://scikit-learn.org/stable/modules/label_propagation.html
but not MIL yet.

Ref: Hosseini, S., & Sardo, S. R. (2020). Data mining tools-a case study for network intrusion detection. *Multimedia Tools and Applications*, 1-21.

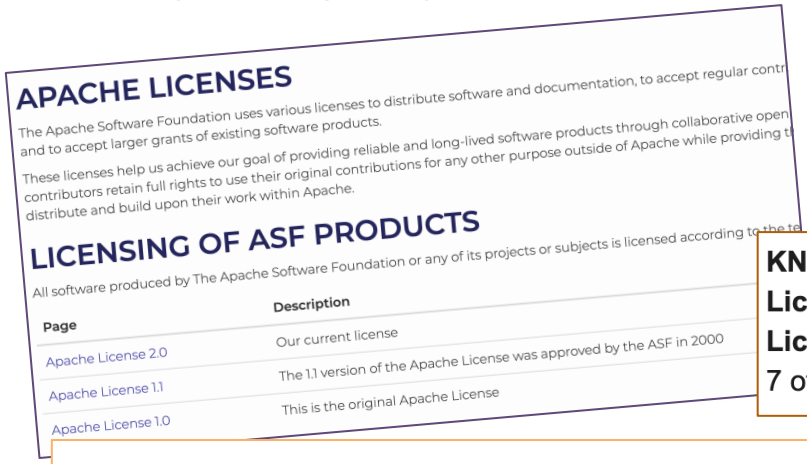
DI501 Introduction to Data Science

Lecture 2 – Part3 Licensing Issues and Important Concepts in Data Science

Platforms & Workbenches

Note on Licensing Issues

- When you develop a data product, you should always take into consideration the licensing terms of the software and packages you have been using.
- Otherwise, you may fail to commercialize your product in your own terms.



APACHE LICENSES

The Apache Software Foundation uses various licenses to distribute software and documentation, to accept regular contr
and to accept larger grants of existing software products.

These licenses help us achieve our goal of providing reliable and long-lived software products through collaborative open
contributors retain full rights to use their original contributions for any other purpose outside of Apache while providing t
distribute and build upon their work within Apache.

LICENSING OF ASF PRODUCTS

All software produced by The Apache Software Foundation or any of its projects or subjects is licensed according to the te

Page	Description
Apache License 2.0	Our current license
Apache License 1.1	The 1.1 version of the Apache License was approved by the ASF in 2000
Apache License 1.0	This is the original Apache License

- KEEL is an open source (GPLv3) Java software tool that can be used for a large number of different knowledge data discovery tasks.

KNIME is a bundle containing Eclipse Software **licensed** under the Eclipse Public **License** (EPL) and separate **KNIME** plug-ins **licensed** under the General Public **License** (GPL), Version 3 (including certain additional permissions according to Sec. 7 of the GPL).

Project Jupyter is a non-profit, open-source project, born out of the IPython Project in 2014 as it evolved to support interactive data science and scientific computing across all programming languages. Jupyter will always be 100% open-source software, free for all to use and released under the liberal terms of the modified BSD license.

Platforms & Workbenches

Note on Licensing Issues

- ***Open Source Definition:***
  - Open source doesn't just mean access to the source code.
  - It is a specification of what is permissible in a license in order to call a software open source.
- Licenses which meet the definition may contain the legend “OSI Certified” or use the OSI Certified logo. These mean that it’s an open source license.
  - OSI: Open Source Initiative (OSI)

Platforms & Workbenches

Note on Licensing Issues

- These OSI Certified license are not all the same, because licensors can use licenses with additional rights beyond the minimums.
- Anyone is free to create their own open source license; they don't have to use an existing license. New licenses may be submitted to the Open Source Initiative for certification.

Comparing the Open Source Licenses

Terminology in the Licenses:

- The Mozilla Public License (MPL) divides a software work into an Open Source part (called "Covered Code (CC)") and anything a contributor adds (aka Secondary Code (SC)).

Possibilities:

- You use an open source software (CC) and modify it. Then you release it -> (changes in the CC but not in the SC)
- You release an software bundle (SB), which comprises CC and SC you developed. ->(no changes in the CC but a new SC by you)
- All open source license terms are defined based on SB, CC and SC.

Comparing the Open Source Licenses

Strong copyleft licenses

- One of the mostly used implementation of copyleft is GPL (GNU General Public License).
- A programmer cannot combine GPL-licensed program code with other codes and then release it to public under non-GPL terms. The intention here is that GPL-licensed software remains freely usable.

Comparing the Open Source Licenses

- If you develop a SC and release it with CC together as a software bundle, both SB and SC should have the same license.
- If you modify CC and revise the code, the new version should have the same license.

- ***Example***: You want to release software for security camera surveillance. You made use of existing image processing algorithms packages (released under GPL) and developed another package for detecting suspicious packages.
  Your final software should have GPL license.
  Example: You developed a new image processing algorithm and embedded it in the same package having GPL license.
  The final software should have GPL license.

SB: software bundle
SC: secondary code
CC: covered code (main code)

Comparing the Open Source Licenses

Strong copyleft licenses

Copyleft is a category of license requirements that govern how modifications to the original open source software must be legally treated when they are publicly distributed.

If a license contains a strong copyleft provision, anyone who modifies the software and distributes it to the public must license the resulting work back to the public under the same terms as the original software.

R as a package is licensed under `GPL-2 | GPL-3`. File `doc/COPYING` is the same as `GPL-2`.

Some files are licensed under 'GPL (version 2 or later)', which includes `GPL-3`. See the comments in the files to see if this applies.

Just like in GPL!!

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

Weak Copyleft licenses

- Mozilla Public License (MPL): most popular weak copyleft license.
  - Created to distribute Mozilla Web browser (open source version of Netscape)
  - Source code included with the modified software, or available for download under the terms of MPL
  - Weak because it's copylefting per file, not the entire work
    - MPL can be loosely regarded as a hybrid of ideas between the GPL and the MIT/BSD licenses.

https://www.mozilla.org/en-US/MPL/2.0/FAQ/

Weak Copyleft licenses

- A distinguishing difference with the other licenses analyzed is that the MPL divides a software work into an Open Source part (called “Covered Code”) and anything a contributor adds.
- This arrangement ***allows developers to add their own files and distribute them with the covered code***, provided they do not modify the covered code. However ***if they modify the covered code, they must distribute the modified code under MPL***.

Weak Copyleft Licenses

Eclipse Public License (EPL)

- Based on the Common Public License v1.0
- Weak copyleft licenses requires you to disclose your source on source code, but not on binaries and therefore you can compile covered sources with others and distribute the resulting (merged) binaries under the licence of your choice.
  - If you modify an EPL'ed component and distribute it in the source code form as part of your program, you're required to disclose the modified code under the EPL.
  - If you distribute such a program in its object code form, you're required to state that the source code can be made available to the recipient upon request.
  - You're also required to share the method for requesting the source code.

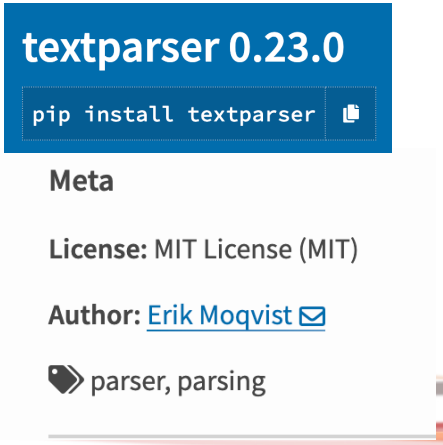
You can't avoid sharing source code!

Permissive Licenses

- Non-copyleft licenses (including “BSD-style” and “MIT-style” licenses):
  - No requirement for all software to be copylefted.
  - The best known no-copyleft license is the Berkeley Software Distribution License (BSD, the earliest non-proprietary license).
- Others:
  - Best known examples are Artistic License and the Academic Free License.

Permissive Licenses

- The MIT and BSD Licenses are the two earliest open source software licenses.
- The MIT License is a simple license that basically grants all of the rights of a copyright holder including ***the exclusive right to commercially exploit and create derivatives from the software***. The only two conditions imposed are that the copyright and permission notices must be included in the copies of the software and a general disclaimer of warranty.



Permissive Licenses

- The BSD License is only slightly more restrictive. Originally it carried a provision that the University of California, Lawrence Berkeley Laboratory must be acknowledged in all advertising and use of the software, but was this was removed later on.
- The only other restriction that is different from the MIT License is that ***the name of the organization that created the software or its contributors cannot be used to endorse or promote the software without prior written permission***.
- The MIT and BSD licenses are one of the most popular open source licenses partly because of they have been around for a long time.

scikit-learn

scikit-learn is a Python module for machine learning built on top of SciPy and is distributed under the 3-Clause BSD license.

The project was started in 2007 by David Cournapeau as a Google Summer of Code project, and since then many volunteers have contributed. See the About us page for a list of core contributors.

It is currently maintained by a team of volunteers.

Permissive Licenses

Apache

- The Apache License version 2.0 is a comprehensive license.
  - It includes provisions for patent rights granted by the license and the use of other licenses for derivative software based on the original software.
  - It also explicitly defines ‘Contributions’ that are special modifications of the software provided to the licensor of the software for its inclusion into the original software. If accepted, the modifications will become part of the original software and will fall under the same license.

Permissive Licenses

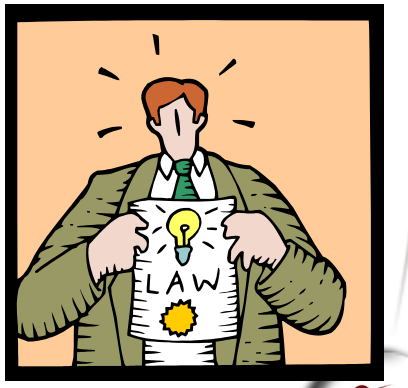
Apache

- The Apache License version 2.0 is a comprehensive license.
  - You can give your modified code away for free, or sell it, or keep it to yourself, or whatever you like. Just remember that the original code is still covered by the Apache license and you must comply with its terms.
  - Even if you change every single line of the Apache code you're using, the result is still based on the Foundation's licensed code.

http://www.apache.org/foundation/license-faq.html#IsItFree

Considerations cont.

- Using an open source program on a day-to-day basis ordinarily doesn't have any illegal implications for the user.
- You can use the modified version as long as the new work is only used internally.

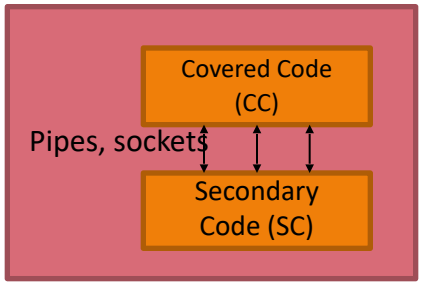


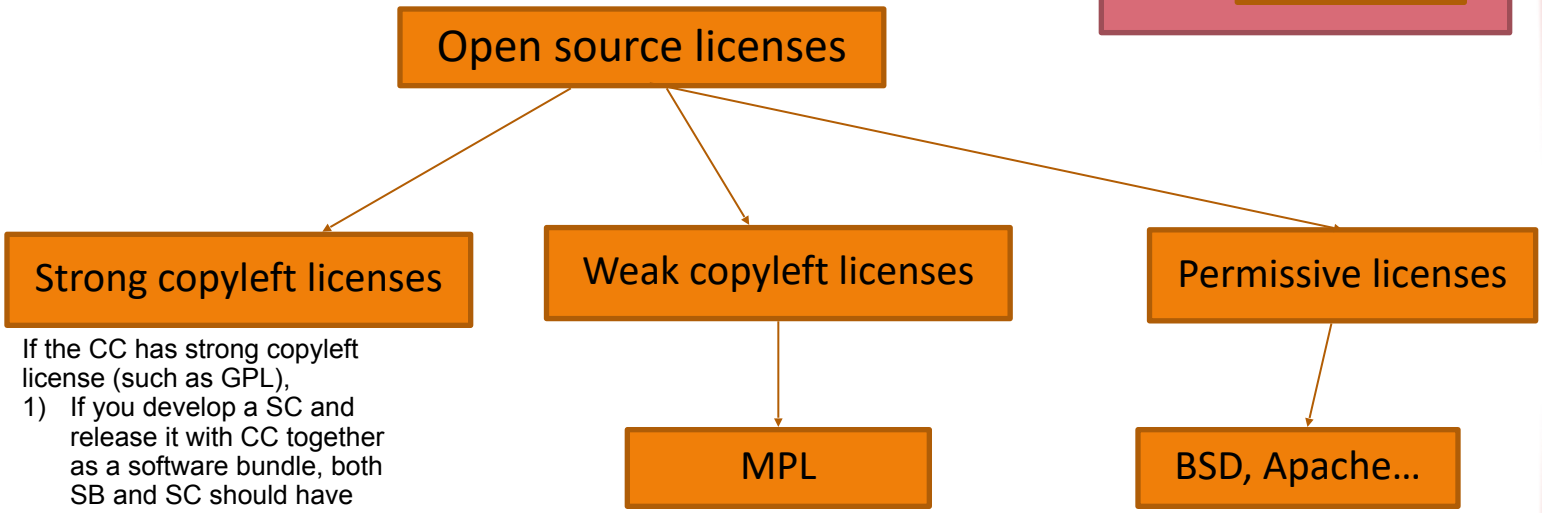
The factors you should consider while you look for a license for your software

- You should stick with one of well known open source licenses such as the GPL, MPL or BSD licenses.
- If you're developing an application that falls within a family or preexisting open source software, you should choose the license that is commonly used for that software.
  - E.g. you should ordinarily use the GPL for Linux kernel code, since Linux is licensed under GPL.
- If you want to prevent end-users from taking their modifications private, use a strong copyleft license such as the GPL.
- If you don't care if users take their modifications private, use a co-copyleft license such as the BSD license.
- If you want copyleft protection for some core functionality, but are willing to grant broader latitude to others integrating your work into third-party systems, choose a weak copyleft license, such as the MPL.

Summary

A software bundle (SB), where CC and SC are interacting.





If the CC has strong copyleft license (such as GPL),

1) If you develop a SC and release it with CC together as a software bundle, both SB and SC should have the same license.
2) If you modify CC and revise the code, the new version should have the same license.

If the CC has weak copyleft license (such as MPL),

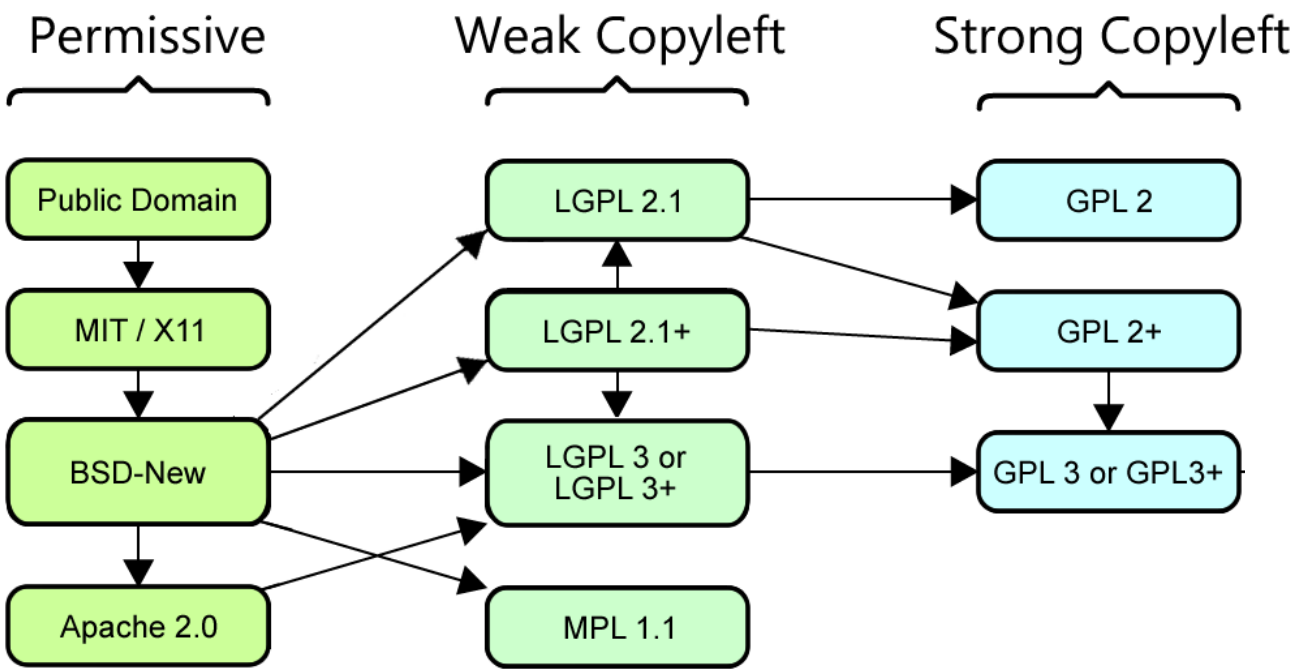
1) If you develop a SC and release it with CC together as a software bundle, both SB and SC can have different licenses.
2) If you modify CC and revise the code, the new version should have the same license (if MPL, then MPL).

If the CC has weak copyleft license (such as BSD),

1) If you develop a SC and release it with CC together as a software bundle, both SB and SC can have different licenses.
2) If you modify CC and revise the code, the new version can have different license.

These are called permissive licenses.

Summary
Compatibility Relationships



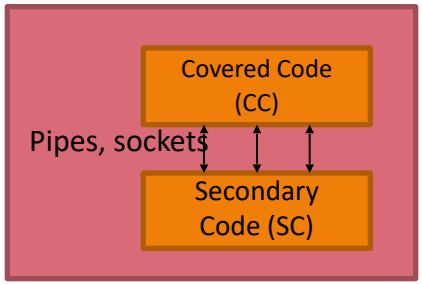
Example: You can combine BSD-New with LGPL 2.1. The final product can have LGPL 2.1 license but not BSD-New.

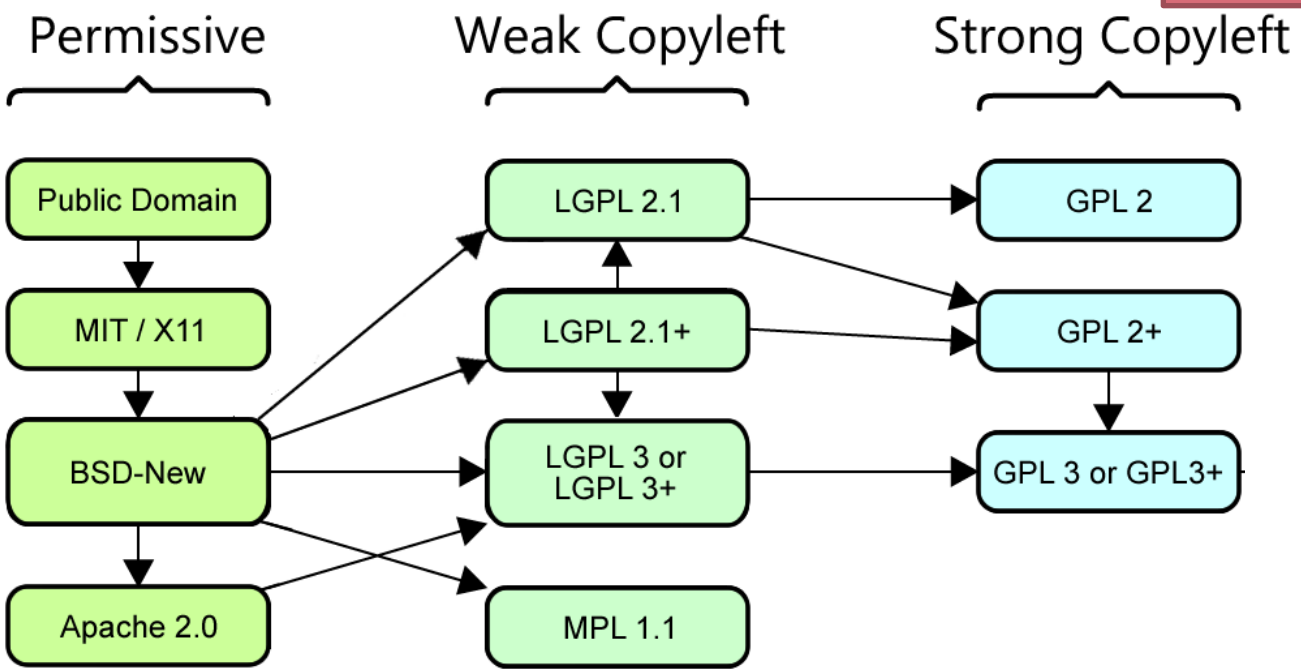
Ref: https://timreview.ca/article/416

Summary

Compatibility Relationships

A software bundle (SB), where CC and SC are interacting.





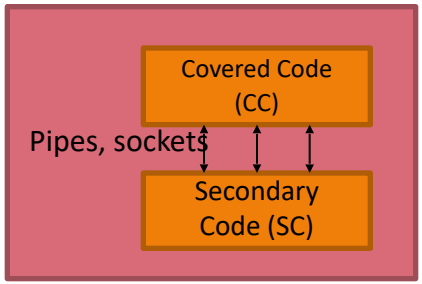
Example: CC is released under MIT license. You develop your SC and license it under GPL 3. You can release both SC and SB as GPL 3.

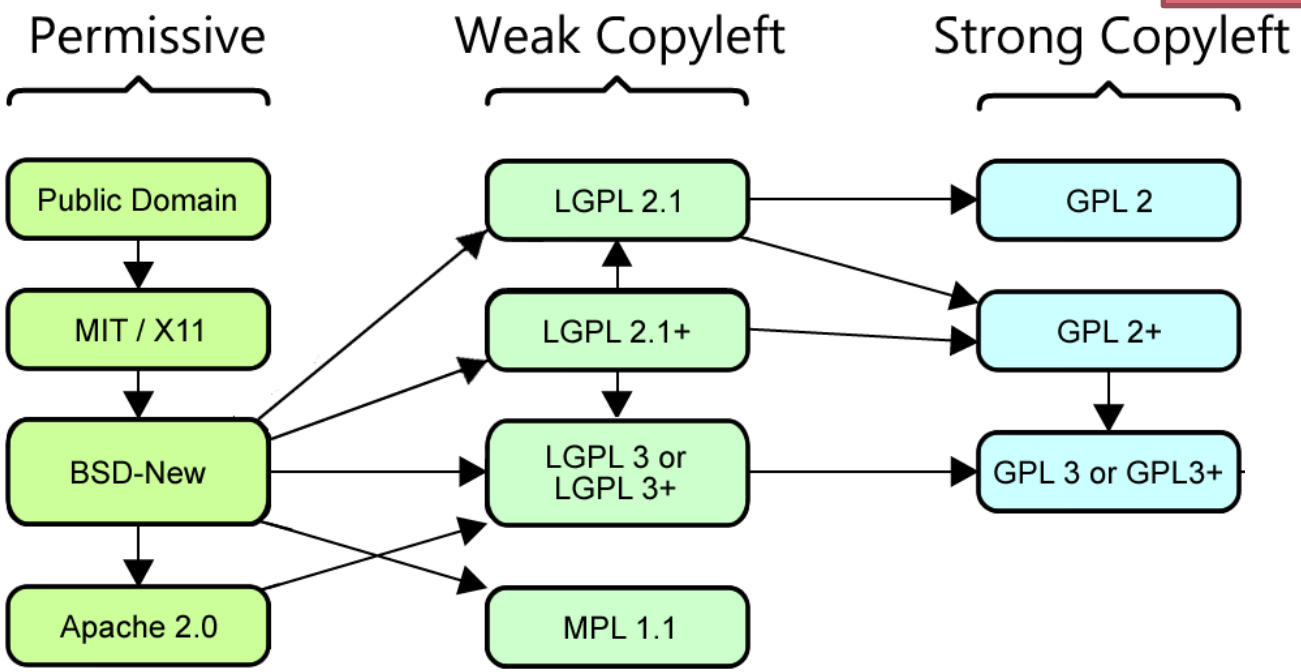
Ref: https://timreview.ca/article/416

Summary

Compatibility Relationships

A software bundle (SB), where CC and SC are interacting.





Example: If CC is Apache 2.0 and SC is LGPL, what license should I use for SB?

Ref: https://timreview.ca/article/416

Important Concepts in Data Science

Cloud Computing

- Cloud computing is internet-based computing, in which shared resources, software and information are provided to computers and other devices on-demand “as a service”.
- In cloud computing the service is run on “The Cloud” and take up very little of your computer’s resources.
- "Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.“ –
  - The National Institute of Standards and Technology (NIST)

Important Concepts in Data Science

Cloud Computing

- One of the main uses of the cloud is collaborative working and distributed access, meaning accessing the same data on multiple devices.
- Service Models:
  - Software as a Service (SaaS): Applications are hosted by a vendor or service provider and made available to customers
  - Platform as a Service (PaaS): Delivers a computing platform (hardware architecture and software framework that allows software to run).
  - Infrastructure as a Service (IaaS): Delivers computer infrastructure, typically a platform virtualization environment.

Important Concepts in Data Science

Software as a Service (SaaS)

- aka “Software on Demand”
- It is a software distribution model in which applications are hosted by a vendor or service provider and made available to customers through the internet.
- Largely free services
- Examples
  - Email (Gmail, Yahoo, etc)
  - Productivity (Google Docs, Microsoft Live, SharePoint)
  - Google Maps, MapQuest, Online Banking, etc.

Important Concepts in Data Science

Infrastructure as a Service (IaaS)

- delivers computer infrastructure, typically a platform virtualization environment, as a service.
- This means rather than purchasing the actual equipment(hardware and/or software) clients instead buy those resources as a fully outsourced service.
- IaaS evolved from virtual private server offerings.
- Examples:
  - Storage space as a service(Humyo, OpenDrive, Mozy, Dropbox, etc)
  - Infrastructure as a service(Google App Engine)
  - *“Super computing” as a Service (Amazon Web Services (AWS), Azure)

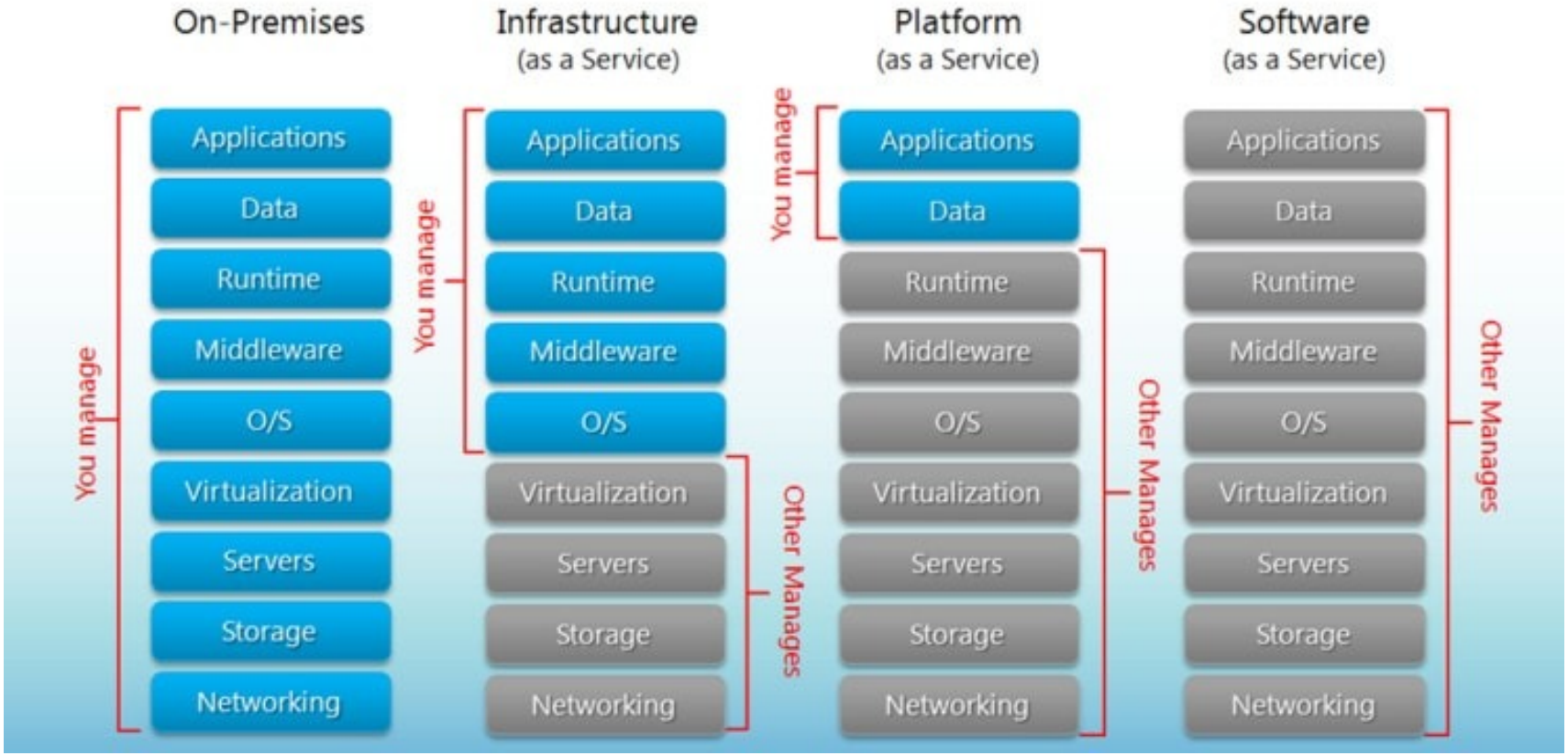
Important Concepts in Data Science

Platform as a service (PaaS)

- is the delivery of a computing platform (hardware architecture and software framework that allows software to run) as a service.
- Google ChromeOS
  - Google is offering users an OS that is running entirely over the internet.
  - Computers no longer need to run BIOS or start up applications, which translates into much faster boot times.
  - You just turn on you computer and are connected to your personal cloud space where you can do everything you would normally do on your computer except now its being run at a remote location.
  - This could allow users to own slower, outdated machine and as long as you have a good internet connection you have a powerful computer.

Important Concepts in Data Science

Cloud Computing: Differences



SaaS examples: BigCommerce, Google Apps, Salesforce, Dropbox, MailChimp, ZenDesk, DocuSign, Slack, Hubspot.

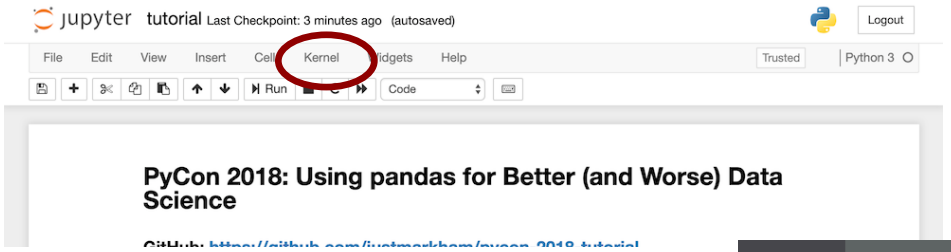
PaaS examples: AWS Elastic Beanstalk, Heroku, Windows Azure (mostly used as PaaS), Force.com, OpenShift, Apache Stratos, Magento Commerce Cloud.

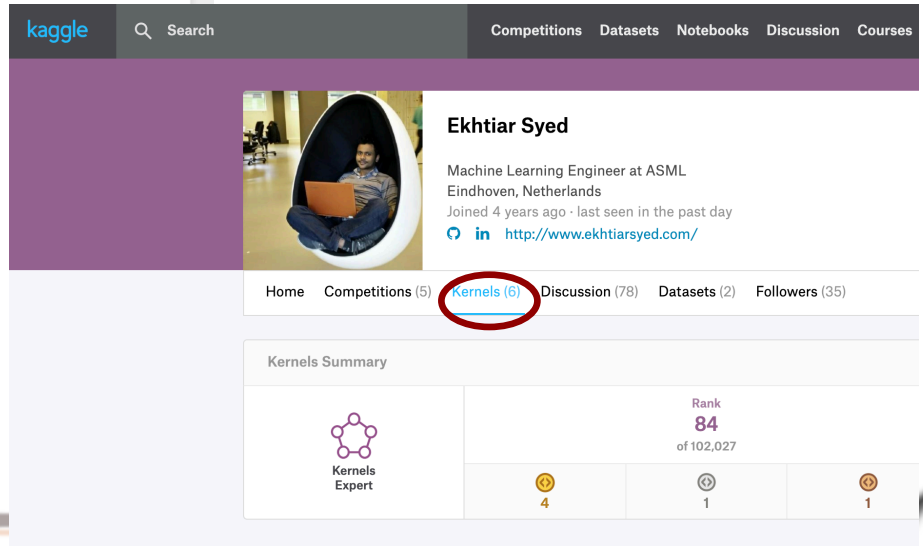
IaaS examples: AWS EC2, Rackspace, Google Compute Engine (GCE), Digital Ocean

Important Concepts in Data Science

Containers, dockers, kernels

- Containers, dockers and kernels are widely used in many data science environments.





Important Concepts in Data Science

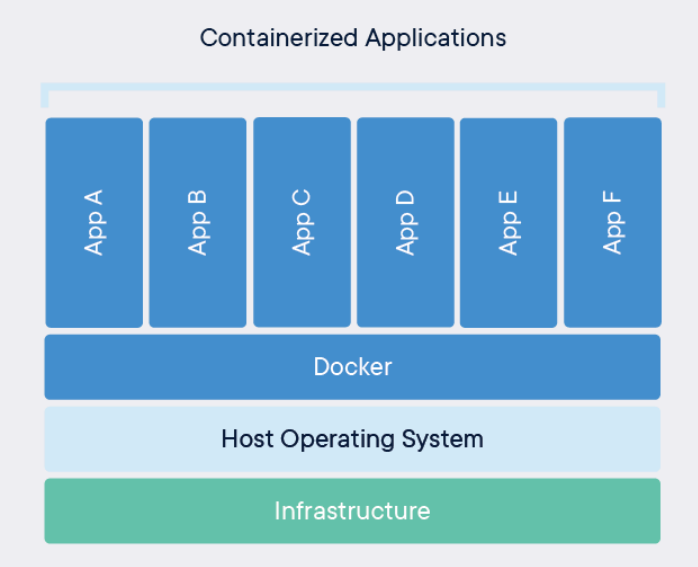
Containers, dockers, kernels

- A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another.
- A Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings.
  - Docker container technology was launched in 2013 as an open source Docker Engine.
  - There are alternatives to Docker container: Mesos by Apache, LXC Linux Containers, OpenVZ, CoreOS rkt…

Important Concepts in Data Science

Containers, dockers, kernels

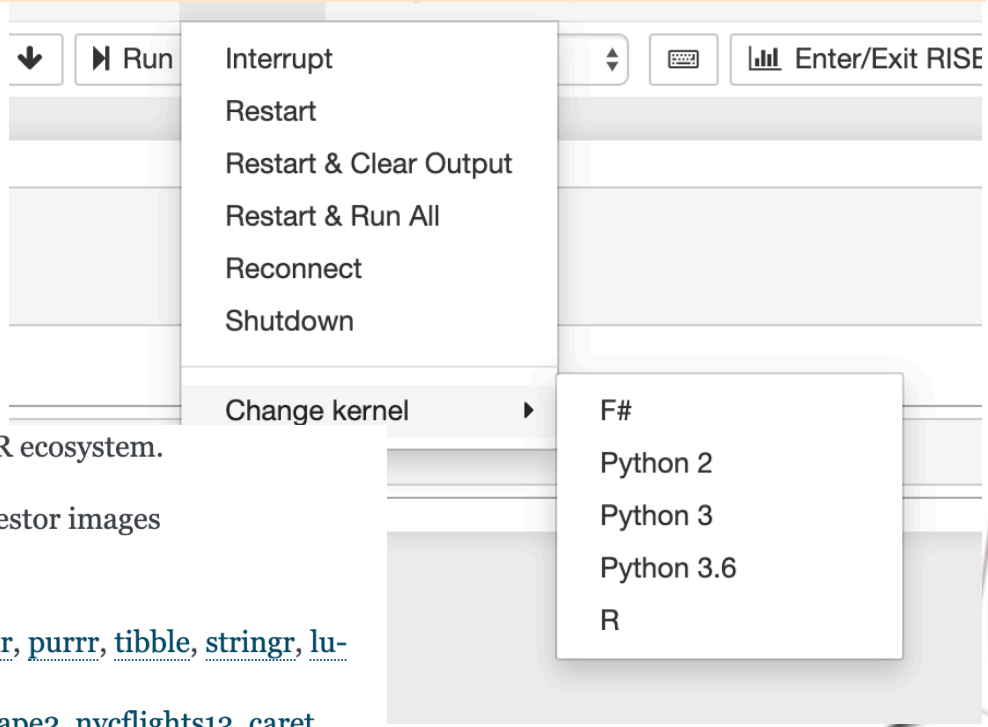
- Containers are an abstraction at the app layer that packages code and dependencies together.
- Multiple containers can run on the same machine and share the OS kernel with other containers, each running as isolated processes in user space.



Important Concepts in Data Science

Containers, dockers, kernels

Jupyter Project offers various docker images in their Github repo.
https://github.com/jupyter/docker-stacks



`jupyter/r-notebook` includes popular packages from the R ecosystem.

- Everything in `jupyter/minimal-notebook` and its ancestor images
- The R interpreter and base environment
- IRKernel to support R code in Jupyter notebooks
- tidyverse packages, including ggplot2, dplyr, tidyr, readr, purrr, tibble, stringr, lubridate, and broom from conda-forge
- plyr, devtools, shiny, rmarkdown, forecast, rsqlite, reshape2, nycflights13, caret, rcurl, and randomforest packages from conda-forge

Important Concepts in Data Science

Containers, dockers, kernels

Jupyter Project offers various docker images in their Github repo.

h[illegible]s

jupyter/datascience-notebook

Source on GitHub | Dockerfile commit history | Docker Hub image tags

`jupyter/datascience-notebook` includes libraries for data analysis from the Julia, Python, and R communities.

- Everything in the `jupyter/scipy-notebook` and `jupyter/r-notebook` images, and their ancestor images
- The Julia compiler and base environment
- IJulia to support Julia code in Jupyter notebooks
- HDF5, Gadfly, and RDatasets packages

jupyter/scipy-notebook

Source on GitHub | Dockerfile commit history | Docker Hub image tags

`jupyter/scipy-notebook` includes popular packages from the scientific Python ecosystem.

- Everything in `jupyter/minimal-notebook` and its ancestor images
- pandas, numexpr, matplotlib, scipy, seaborn, scikit-learn, scikit-image, sympy, cython, patsy, statsmodel, cloudpickle, dill, numba, bokeh, sqlalchemy, hdf5, vincent, beautifulsoup, protobuf, and xlrd packages
- ipywidgets for interactive visualizations in Python notebooks
- Facets for visualizing machine learning datasets

Important Concepts in Data Science

Example: Kaggle& Kernels

- **KAGGLE** is an online community of data scientists and machine learners, owned by Google LLC.
- Kaggle allows users to find and publish data sets, explore and build models in a web-based data-science environment, work with other data scientists and machine learning engineers, and enter competitions to solve data science challenges.
- In June 2017, Kaggle announced that it passed 1,000,000 registered users, or Kagglers. The community spans 194 countries.

https://www.kaggle.com/

Important Concepts in Data Science

Example: Kaggle& Kernels

- **Kaggle Kernels:** a cloud-based workbench for data science and machine learning.
  - Allows data scientists to share code and analysis in Python and R.
  - Over 150K "kernels" (code snippets) have been shared on Kaggle covering everything from sentiment analysis to object detection.
- They use Docker containers at the heart of Kaggle Scripts. Playing around with Scripts can give you a sense of what you can do with data science containers.
- To run Kaggle Scripts, they put together three Docker containers: kaggle/rstats has an R installation with all of CRAN and a dozen extra packages, kaggle/julia has a recent build of Julia 0.5 with a set of data science libraries installed, and kaggle/python is an Anaconda Python setup with a large set of libraries.