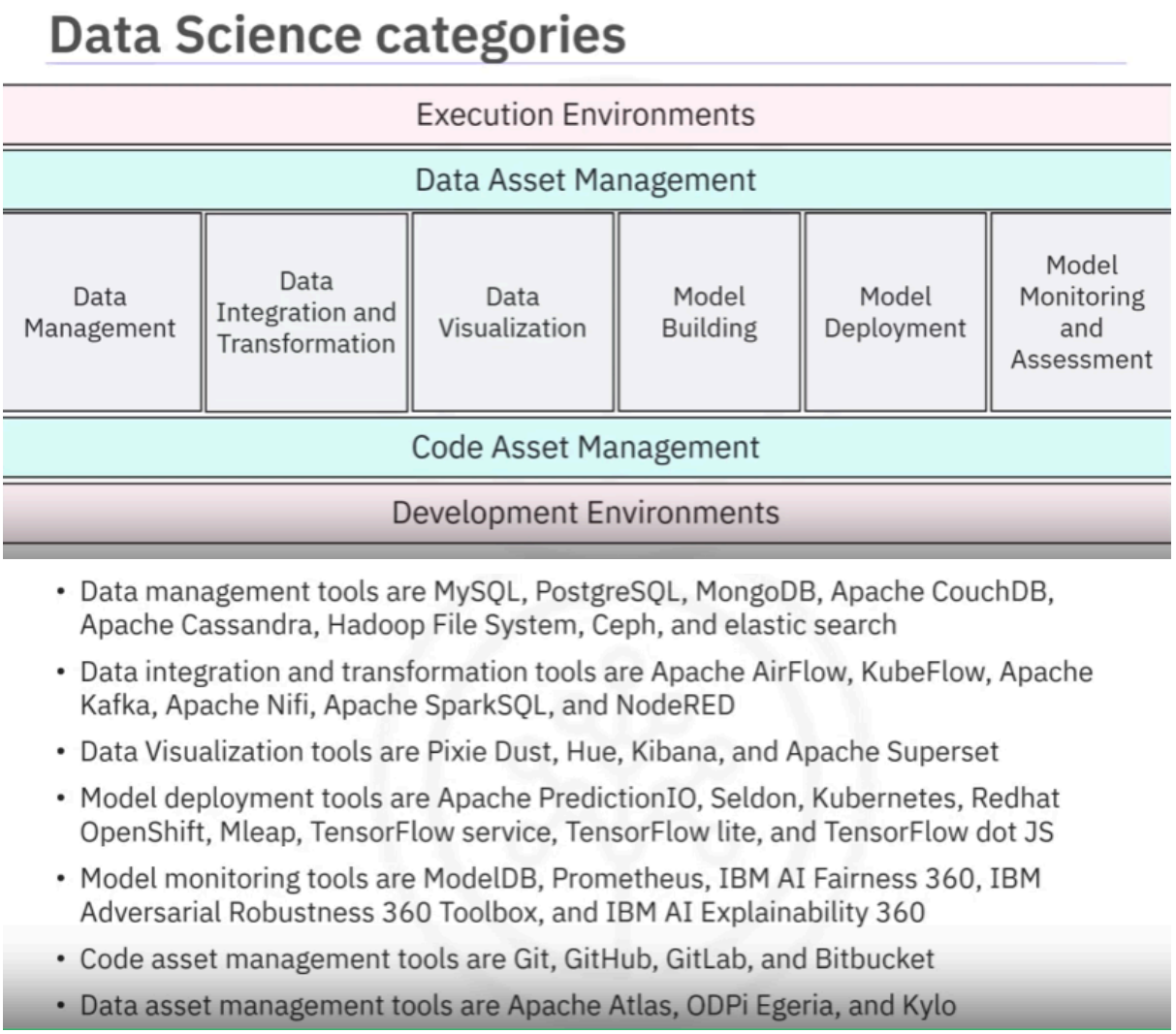
Categories of Data Science Tools:



1. **Data Management:**

Data management is the process of collecting, persisting, and retrieving data securely, efficiently, and cost-effectively. Data is collected from many sources, like Twitter, Flipkart, Media, Sensors, and more. Store collected data in persistent storage so it is available whenever you need it

Open Source Tools:

Most widely used open-source data management tools are relational databases like MySQL and PostgreSQL. Also, there are NoSQL Databases like MongoDB, Apache CouchDB, and Apache Cassandra. In addition, there are file-based tools like the Hadoop File System or Cloud File systems like Ceph. You also have an elastic search tool that stores text data, including the creation of a search index for fast document retrieval.

2. **Data Integration and Transformation:**

Data Integration and Transformation, is the process of Extracting, Transforming, and Loading data. This is called “ETL”. Some of this data is distributed in multiple repositories. For example, a database, a data cube, and flat files. Use the Extraction process to extract data from these numerous repositories and save to a central repository like a Data

Warehouse. Data Warehouses are primarily used to collect and store massive amounts of data for data analysis. Next, Data Transformation is the process of transforming the values, structure, and format of data. After extracting the data, the next step is to transform the data. Transformed data is loaded back to the Data Warehouse.

Open Source Tools:

The task of data integration and transformation in the classic data warehousing world is to Extract, Transform, and Load (ETL). Data scientists often propose Extract, Load, Transform (ELT) as data is dumped somewhere, and the data engineer or data scientist handles the transformation of the data. Another term for this process emerged: Data Refinery and Cleansing. The most widely used open-source data integration and transformation tools are the following: Apache AirFlow, which was created by Airbnb originally. KubeFlow, which allows the execution of data science pipelines on top of Kubernetes. Apache Kafka, which originated from LinkedIn. Apache Nifi, which delivers a very nice visual editor. Apache SparkSQL, lets you use ANSI SQL and scales up to compute clusters of thousands of nodes and NodeRED also brings a visual editor. In addition, NodeRED is so low in resource consumption that it even runs on tiny devices like a Raspberry Pi.

3. **Data Visualisation:**

Data visualisation is the graphical representation of data and information. You can use visualisation to represent data in the form of charts, plots, maps, animations, etc. And data visualisation conveys data more effectively for decision-makers. It is a crucial step in the data science process. Various forms of data visualisations include a bar chart, which compares the size of each component, a treemap, which displays hierarchy data, a line chart, which plots a series of data points over time, and a map chart, which displays data by location. Map charts can also be applied to other locations like websites.

Open Source Tools:

Let's discuss the most widely used open-source data visualisation tools. You must distinguish between programming libraries where you must use code or tools containing a user interface. Pixie Dust is also a library but has a user interface that facilitates plotting in Python. A similar approach uses Hue, which can create visualisations from SQL queries. Kibana, a data exploration, and visualisation web application, is limited to Elasticsearch (data provider). And finally, Apache Superset is a data exploration and visualisation web application.

4. **Model Building:**

This is where you train the data and analyse patterns with machine learning algorithms. The system ‘learns’ how to provide predictions or decisions by itself. You can then use this model to make predictions on new, unseen data. Model building can be done using a service called IBM Watson Machine Learning. It provides a full range of tools and services for building models.

5. **Model Deployment:**

The process of integrating a developed model into a production environment. In model deployment, a machine learning model is made available to third-party applications via APIs. Business users can access and interact with the data through these third-party applications. And this helps them make data-based decisions. As an example, the SPSS Collaboration and Deployment Services can be used to deploy any type of asset created by the SPSS software tools suite.

Open Source Tools:

Model deployment is a crucial step. Once you’ve created a machine learning model capable of predicting some critical aspects of the future, you should make it consumable by other

developers and turn it into an API. Apache PredictionIO currently only supports Apache Spark ML models for deployment, but support for all libraries is on the roadmap. Seldon is an interesting product since it supports nearly every framework including TensorFlow, Apache SparkML, R, and scikit learn. Interestingly, it can run on top of Kubernetes and Redhat OpenShift. Another way to deploy SparkML models is MLeap. Finally, TensorFlow can serve any tensor flow model using the TensorFlow service. It can be an embedded device like a Raspberry Pi or smartphone using TensorFlow lite and deployed to a web browser using TensorFlow dot JS.

6. **Model Monitoring and assessment:**

Model monitoring and assessment run continuous quality checks to ensure a model's accuracy, fairness, and robustness. Model monitoring uses tools like Fiddler to track the performance of deployed models in a production environment. Now, model assessment uses evaluation metrics like the F1 score, true positive rate, or the sum of squared error to understand a model's performance. A well-known example is the IBM Watson Open scale, which continuously monitors deployed machine learning and deep learning models. It will improve the accuracy and quality of your predictions.

Open Source Tools:

Model monitoring is an important step as well. Once you've deployed a machine learning model, you want to track its prediction performance while new data arrives to maintain outdated models. Some examples are the following: ModelDB is a machine model metadata base where information about the models is stored and queried. It natively supports Apache Spark ML Pipelines and scikit-learn. A generic, multi-purpose tool called Prometheus is widely used as well. Although it is not specifically made for machine learning model monitoring, it is used for this purpose. Model performance is measured by more than accuracy. Model bias against protected groups like gender or race is important as well. The IBM AI Fairness 360 open-source toolkit detects and mitigates bias in machine learning models. These models, especially neural network-based deep learning models, can be subject to adversarial attacks where an attacker tries to mislead the model with manipulated data or by controlling it. The IBM Adversarial Robustness 360 Toolbox detects vulnerability against adversarial attacks and leverages the model to be more robust. Finally, machine learning modes are often considered as a black box applying some magic. The IBM AI Explainability 360 toolkit addresses that problem by finding similar examples in a dataset to be presented to an end-user for manual comparison. IBM AI Explainability 360 toolkit can also address the training of a simpler machine learning model to explain the responsibility of different input variables directed toward the final decision of the model.

7. **Code and Data asset management:**

Code asset management provides a unified view where you manage an inventory of assets. When you want to develop a model, you may need to update it, fix bugs, or improve code features incrementally. All of this requires version control. Developers use versioning to track and manage changes to a software project's code. When working on a model, teams have a centralised repository where everyone can upload, edit, and manage the code files simultaneously. Collaboration allows diverse people to share and update the same project together. GitHub is a good example of a code asset management platform. It's web-based and provides sharing, collaboration, and access control features. As a data scientist, you want to properly store and organise all your images, videos, text, and other data in a central location. You also want control over who can access, edit, and manage your data. Data asset management, also called digital asset management (DAM), is the organising and managing of important data collected from different sources. DAM is performed on a DAM platform that allows versioning and

collaboration. DAM platforms also support replication, backup, and access right management for the stored data.

Open Source Tools:

Git is now the de facto standard for code asset management, also known as version management or version control. Around Git emerged several services. The most prominent is GitHub, but the runner-up is GitLab, with the advantage that the platform is entirely open source and can be hosted and managed on your own. Another choice is Bitbucket. Data asset management, also known as data governance or data lineage, is a crucial part of enterprise-grade data science. Data has to be versioned and annotated with metadata. Apache Atlas is such a tool supporting this task. Another interesting project is ODPi Egeria, managed through the Linux Foundation, is an open ecosystem that offers a set of open APIs, types, and interchange protocols that metadata repositories use to share and exchange data. And finally, Kylo is an open-source data management software platform, with extensive support for data asset management tasks

8. **Development Environments:**

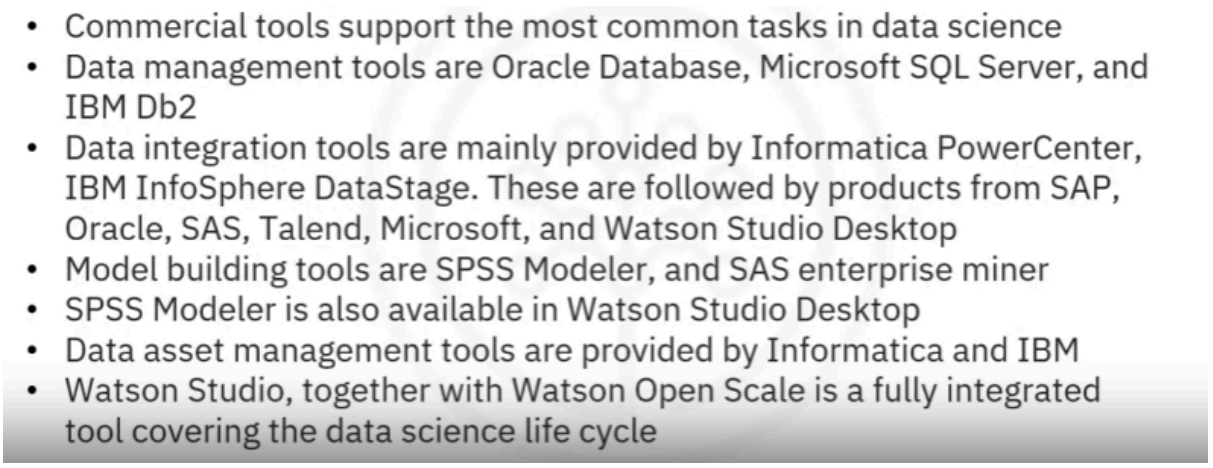
Development Environments, also called Integrated Development Environments, or "IDEs", provide a workspace and tools to develop, implement, execute, test, and deploy source code. IDEs like IBM Watson Studio provide testing and simulation tools to emulate the real world so you can see how your code will behave after it is deployed. An execution environment has libraries to compile the source code and system resources that execute and verify the code. Cloud-based execution environments are not tied to any specific hardware or software, and offer tools like IBM Watson Studio for data preprocessing, model training, and deployment.

9. **Fully Integrated visual tools:**

Finally, fully integrated visual tools like IBM Watson Studio and IBM Cognos Dashboard Embedded cover all the previous tooling components, and can be used to develop deep learning and machine learning models.

https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0130EN-SkillsNetwork/storyline/Open%20Source%20Tools/story.html?origin=www.coursera.org

Commercial Tools for Data Science:



In data management, most of an enterprise's relevant data is stored in an Oracle Database, in Microsoft SQL Server, or an IBM Db2. Although open-source databases are coming to the forefront, these three data management products are considered industry-standard and will be around for a while. In addition, it's not only about functionality. Since data is the heart of every organisation, commercial support availability plays a major role. Commercial support is delivered directly from software vendors, influential partners, and support networks.

commercial data integration tools that comprise extract, transform, and load (ETL) tools. According to a Gartner Magic Quadrant, Informatica PowerCenter and IBM InfoSphere DataStage are the leaders. These are followed by SAP, Oracle, SAS, Talend, and Microsoft products. These tools support the design and deployment of ETL data processing pipelines through a graphical interface. They bring along connectors to most of the commercial and open-source target information systems. Finally, Watson Studio Desktop includes a component called Data Refinery, which enables definition and execution of data integration processes in a spreadsheet-style.

In the commercial environment, data visualisations use business intelligence (BI) tools. The focus of these tools is to create visual reports and live dashboards. The most prominent commercial representatives are: Tableau, Microsoft Power BI, and IBM Cognos Analytics. Another type of visualisation targets data scientists rather than end users. For example, the visualisation can show relationships between different columns in a table. This functionality is contained in Watson Studio Desktop.

If you want to build a machine learning model with a commercial tool, you should use a data mining product. The most prominent products in that space are: SPSS Modeler and SAS enterprise miner. In addition, SPSS Modeler is also available in Watson Studio Desktop, based on the tool's cloud version. Now, Model deployment in commercial software is tightly integrated into the model-building process. Here is an example of the SPSS Collaboration and Deployment Services, which is used to deploy any type of asset created by the SPSS software tools suite. The same holds for other vendors. Also, commercial software can export models in an open format. For example, SPSS Modeler supports exporting models as predictive model markup language (PMML), which an abundance of other commercial and open software packages can read.

Model monitoring is a very new discipline. Currently, relevant commercial tools are not available. Therefore, open source is the first choice. The same is true for code asset management. Open source with Git and GitHub is the de facto standard.

Data asset management, often called data governance or data lineage, is a crucial part of enterprise-grade data science. Data must be versioned and annotated with metadata. Vendors, including Informatica Enterprise Data Governance and IBM, provide tools for these specific tasks. The Information Governance Catalog covers functions like a data dictionary, which facilitates the discovery of data assets. Each data asset is assigned to a data steward or the data owner. The data owner is responsible for that data asset and can be contacted. Then, data lineage is covered, allowing tracking back the transformation steps in creating the data assets. The data lineage also includes a reference to the actual source data. Rules and policies can be added to reflect complex regulatory and business requirements for data privacy and retention.

Watson Studio is a fully integrated development environment for data scientists. Most people consume it through the cloud. And there is also a desktop version available. Watson Studio

Desktop combines Jupyter Notebooks with graphical tools to maximise the performance of data scientists. Watson Studio, together with Watson Open Scale, is a fully integrated tool covering the data science life cycle involving all tasks discussed previously. They can be deployed in a local data centre, on top of Kubernetes / RedHat OpenShift. Another example of a fully integrated commercial tool is H2O Driverless AI, which covers the complete data science life cycle.

Cloud Based Tools for Data Science:

- Watson Studio and Watson OpenScale cover the complete development life cycle for all data science, machine learning, and AI tasks
- In data management, with some exceptions, there exists a software-as-a-service (SaaS) version of existing open source and commercial tools
- Two commercial data integration tools widely used are Informatica Cloud Data Integration and IBM's Data Refinery
- An example of a cloud-based data visualization tool is Datameer and IBM's Congos Business intelligence suite
- Model building can be done using a service such as Watson Machine Learning
- Amazon SageMaker Model Monitor is an example of a cloud tool to monitor deployed machine learning and deep learning models continuously

Languages of Data Science:

Choosing the language depends on the problem to solve and who you are solving them for.

The popular languages are Python, R, SQL, Scala, Java, C++, and Julia. JavaScript, PHP, Go, Ruby, and Visual Basic all have their own unique use cases as well. The problems you need to solve can be related to your company, role, and age of the existing application.

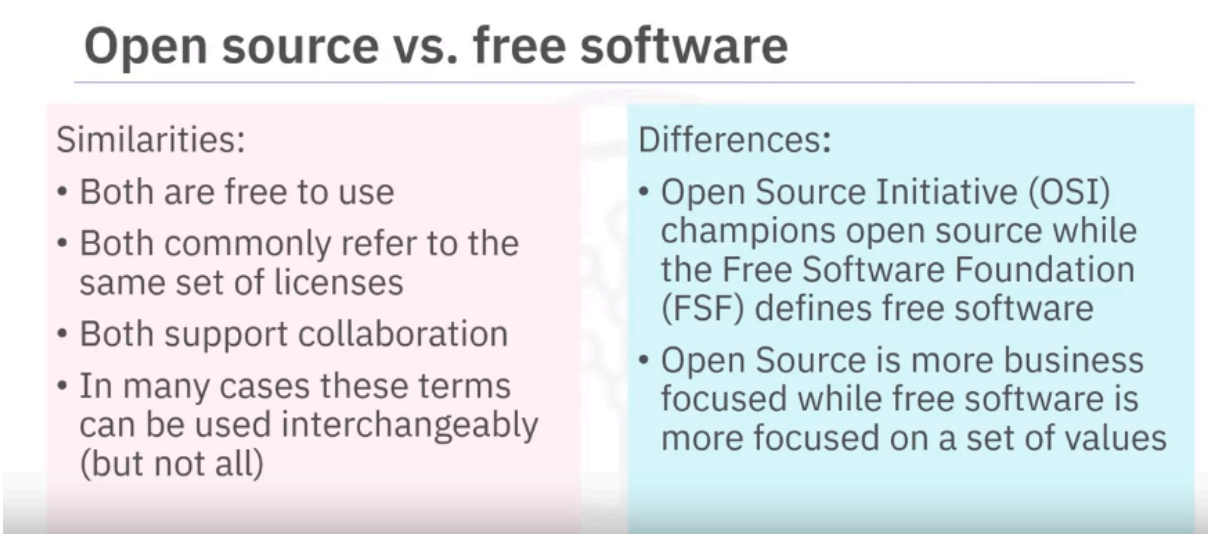
Python:

Most widely used language in Data Science.
Python is used in many areas including data science, AI and machine learning, web development, and Internet of Things (IoT) devices, like the Raspberry Pi.

Large organisations that heavily use python include IBM, Wikipedia, Google, Yahoo!, CERN, NASA, Facebook, Amazon, Instagram, Spotify, and Reddit. Python is widely supported by a global community and shepherded by the Python Software Foundation.

Python is a high-level, general-purpose programming language that can be applied to many different classes of problems. It has a large, standard library that provides tools suited to many different tasks including but not limited to Databases, Automation, Web scraping, Text processing, Image processing, Machine learning, and Data analytics. For data science, you can use Python's scientific computing libraries like Pandas, NumPy, SciPy, and Matplotlib. For artificial intelligence, it has TensorFlow, PyTorch, Keras, and Scikit-learn. Python can also be used for Natural Language Processing (NLP) using the Natural Language Toolkit (NLTK).

R Language:
R is a free software and Python is Open Source.



Statisticians, mathematicians, and data miners use R to develop statistical software, graphing, and data analysis. R Language's array-oriented syntax makes it easier to translate from math to code for learners with no or minimal programming background.

Companies that use R include IBM, Google, Facebook, Microsoft, Bank of America, Ford, TechCrunch, Uber, and Trulia.
R has become the world's largest repository of statistical knowledge.(more than 15000 publicly released packages to conduct complex data analysis)

R integrates well with other computer languages like C++, Java, C, .Net, and Python. Using R, common mathematical operations like matrix multiplication give immediate results. And R has stronger object-oriented programming facilities than most statistical computing languages.

SQL:
Useful in handling Structured data.
When performing operations with SQL, the data is accessed directly, without needing to copy the data separately, which can considerably speed up workflow executions. SQL behaves like an interpreter between you and the database.
SQL is an American National Standards Institute (or ANSI) standard, which means if you learn SQL and use it with one database, you can apply your SQL knowledge to many other databases easily.

Now, many different SQL databases are available, including the following: MySQL, IBM DB2, PostgreSQL, Apache Open Office Base, SQLite, Oracle, MariaDB, Microsoft SQL Server, and more. The syntax of the SQL you write may change based on the relational database management system you are using.

Other Languages:
Scala, Java, C++, and Julia are probably the most traditional data science languages.
However, JavaScript, PHP, Go, Ruby, Visual Basic and many others have found their place in the data science community.

Scala

- A general-purpose programming language that provides support for functional programming
- Designed as an extension to Java, it is inter-operable with Java as it also runs on JVM
- The name Scala comes from “Scalable Language”

For Data Science, Apache Spark:

- Provides APIs that make parallel jobs easy to write
- Optimized engine that supports computation graphs
- Includes Shark, MLlib, GraphX, and Spark Streaming
- Designed to be faster than Hadoop

Java

- A general-purpose object-oriented programming language
- Huge adoption in the enterprise space, designed to be fast and scalable
- Applications are compiled to bytecode and run on JVM

For Data Science, Java tools are:

- Weka (data mining), Java-ML (ml library), Apache MLlib (scalable ml), and Deeplearning4
- Hadoop manages data processing and storage for big data applications running in clustered systems

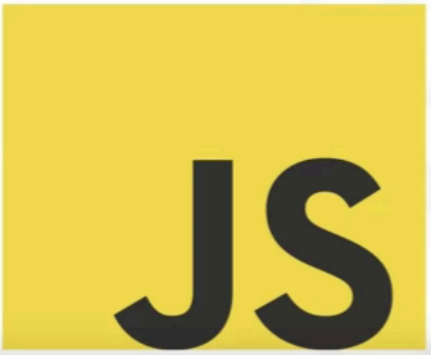
C++

- A general-purpose language, an extension of C
- Improves processing speed, enables system programming, and gives you broader control over the application
- Develops programs that feed data to customers in real-time

For Data Science, C++ applications are:

- TensorFlow a Deep Learning library
- MongoDB a NoSQL database for big data management
- Caffe a deep learning algorithm repository

JavaScript



- A core technology for the world wide web
- A general-purpose language that extended beyond the browser with Node.js and other server-side approaches
- NOT related to the Java language

For Data Science, JavaScript applications are :

- TensorFlow.js makes machine learning and deep learning possible in Node.js and in the browser
  - Adopted by other open-source libraries including brain.js and machinelearn.js
- R-js makes linear algebra possible in typescript (superset of JavaScript)

Julia

- Designed for high-performance numerical analysis and computational science
- Provides speedy development and fast programs
- Executes directly on the processor
- Calls C, Go, Java, MATLAB, R, Fortran, and Python libraries with refined parallelism
- A young language with a lot of promise

For Data Science, JuliaDB, a package for working with large persistent data sets

Libraries for Data Science:
Libraries are a collection of functions and methods that allow you to perform many actions without writing the code.

Python Libraries:
Scientific Computing Libraries in Python Visualization Libraries in Python High-Level- Machine Learning and Deep Learning Libraries (High-level means you don't have to worry about details making studying or improving difficult.) And finally, Deep Learning Libraries in Python, and Libraries used in other languages.

Scientific Computing Libraries in Python:
Scientific computing libraries contain built-in modules providing different functionalities, which you can use directly. They are also called frameworks.

1. Pandas (Data structures and tools)
2. Numpy (Arrays and matrices)

Visualisation Libraries in Python:
We use data visualisation methods to communicate with others and display meaningful results of an analysis. These libraries enable you to create graphs, charts, and maps.

1. Matplotlib(plots and graphs)
2. Seaborn (Plots: heat maps,time series,Violin plots)

Machine learning and Deep Learning Libraries in Python:

1. Scikit-learn (Machine Learning: Regression, classification, clustering)
2. Keras(Deep Learning Neural networks)
3. TensorFlow (Deep Learning: Production and Deployment)
4. PyTorch(Deep LEarning: regression and classification)

Scala libraries:

1. Vegas (for statistical data visualisations)
2. Big DL (for Deep Learning)

R libraries:

1. ggplot2 (for data visualisation)

API's:
An application programming interface (API) allows communication between two pieces of software. For example, in a program, you have some data and other software components. You use the API to communicate using inputs and outputs without knowing what happens at the backend. The API only refers to the interface. It is the part of the library you see while it contains all the program components.

REST API's:
The RE stands for Representational. The S stands for State. The T stands for Transfer. They allow you to communicate through the internet and take advantage of resources like storage, data, artificially intelligent algorithms, and much more. In Rest API, your program is the client. The API communicates with a web service you can call through the internet. Though there are rules regarding Communication, Input or Request, and Output or Response.

Data-Set:
Structure of collection of data.
CSV, Tabular model

https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-DS0105EN-SkillsNetwork/labs/Labs_V4/Additional_Sources_Of_DataSeta.md.html?origin=www.coursera.org?origin=www.coursera.org

What is a machine learning model?

Data contains a wealth of information

Machine Learning (ML) models identify patterns in data

Model training is the process by which the model learns the data patterns

After a model is trained it can be used to make predictions

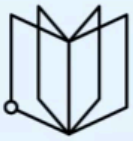
Types of ML are Supervised, Unsupervised, and Reinforcement

,

Supervised Learning

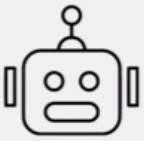
The Supervised Learning model identifies relationships and dependencies between the input data and the correct output.

- Regression — To predict real numerical values
  - Examples: home sales prices, stock market prices
- Classification — To classify data into categories
  - Examples: email spam filters, fraud detection, image classification



Other learning types

- Unsupervised Learning
  - Data is not labeled
  - Model tries to identify patterns without external help
    - Clustering divides each record of a dataset into one of similar group
    - Anomaly detection identifies outliers in a dataset
- Reinforcement Learning
  - Conceptually similar to human learning processes
  - Examples: Mouse and maze, robot learning to walk, chess, Go, and other board games of skill



Deep Learning

- Tries to loosely emulate show the human brain works
- Applications
  - Natural Language Processing
  - Image, audio, and video analysis
  - Time series forecasting
- Requires large datasets of labeled data and is compute intensive
- Requires special purpose hardware

Deep Learning Models

- Build from scratch or download from public model repositories
- Built using frameworks, such as
  - TensorFlow
  - PyTorch
  - Keras
- Provide Python API and support C++ and JavaScript
- Popular model repositories
  - Most frameworks provides a “model zoo”
  - TensorFlow, PyTorch, Keras, and ONNX model zoos

Using models to solve a problem

What is this?

Prepare Data → Build model → Train model → Deploy model → Use model

Iterative process:

Requires data, expertise, time, and resources

Time to Value

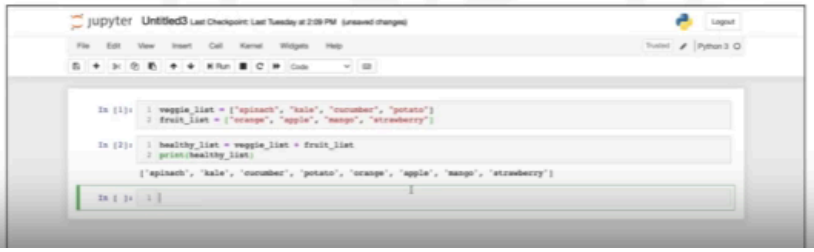
This is a teddy bear.

Pre-trained models reduce time to value

Jupyter Notebook:

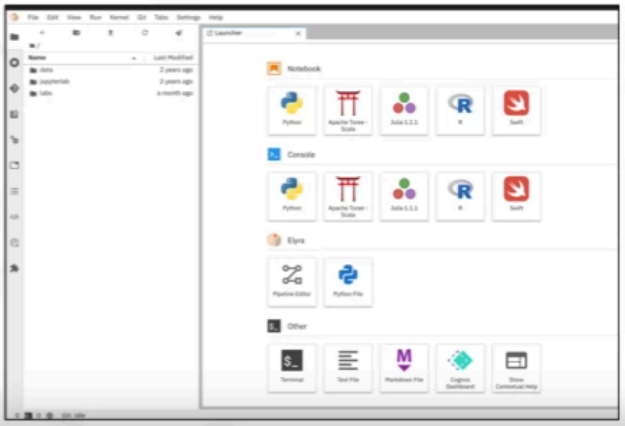
Jupyter Notebook:

- Records Data Science experiments
- Allows combining text, code blocks, and code output in a single file
- Exports the notebook to a PDF or HTML file format



JupyterLab

- Allows access to multiple Jupyter Notebook files, other code, and data files
- Enables working in an integrated manner
- Is compatible with several file formats
- Is an open source



Anaconda

- A free and open-source distributor for Python and R
- Has 1500+ libraries
- Free to install
- Free community support
- Installs new packages without a CLI
- Download **Anaconda Navigator** at https://www.anaconda.com/

VisualStudio Code (VS Code)

- A free, open-source code editor for operations like debugging and task running
- Works on Linux, Windows, and MacOS
- Supports:
  - multiple languages
  - syntax highlighting
  - auto-indentation
- One of the most popular development environment tools

JupyterLite

JupyterLite:

- Lightweight tool built from Jupyterlab components
- Executes in the browser
- Dedicated Jupyter server not required
- Can deploy as a static website
- Can create interactive graphics and visualizations
- Supports visualization libraries like Altair, Plotly, and ipywidgets
- Includes JupyterLab's latest improvements and features

https://jupyter.org/try-jupyter/lab/

Google Colaboratory (GoogleColab)

Google Colaboratory (GoogleColab) is a free Jupyter notebook environment that runs entirely in the cloud.

- Execute on a browser
- Store on Google drive and GitHub
- Upload and share without setup and installation
- Clone from GitHub and execute in GoogleColab
- Most libraries are pre-installed (scikit-learn, matplotlib)

R and RStudio:

What is R?

- Statistical programming language
- Used for data processing and manipulation
- Statistical, data analysis, and machine learning
- R is used most by academics, healthcare and the government
- R supports importing of data from different sources: Flat files, databases, web, statistical software

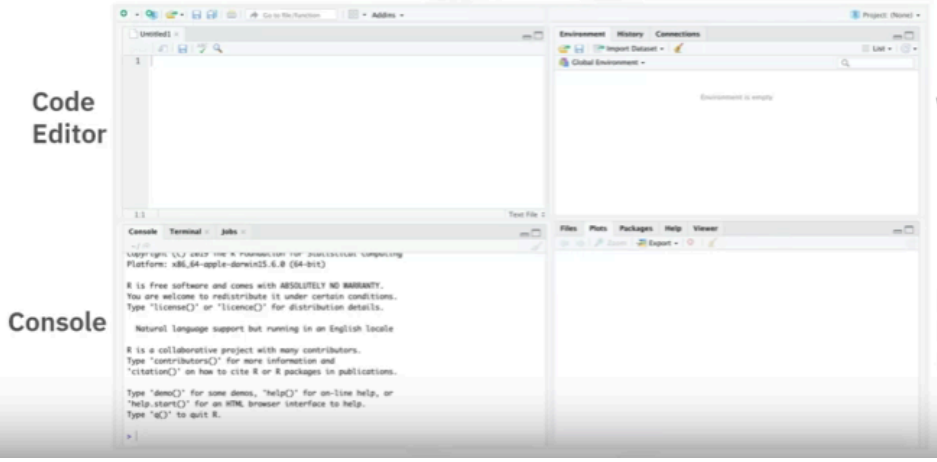
R Capabilities

- It is easy to use compared to other data science tools
- Great tool for visualization
- Basic data analysis doesn't require installing packages

What is RStudio

- RStudio is an Integrated Development Environment (IDE)
- It increases productivity in running **R** programming language

RStudio



Popular R Libraries for Data Science

dplyr	Data Manipulation
stringr	String Manipulation
ggplot	Data Visualization
caret	Machine Learning

Using data visualization in R

To install packages, use the command: install.packages <package name>

Packages are:

- ggplot: Histograms, bar charts, scatterplots
- Plotly: Web-based data visualizations
- Lattice: Complex, multi-variable data sets
- Leaflet: Interactive plots

Git/Github:

SSH protocol – A method for secure remote login from one computer to another.

Repository - The folders of your project that are set up for version control.

Fork - A copy of a repository.

Pull request – The process you use to request that someone reviews and approves your changes before they become final.

Working directory – A directory on your file system, including its files and subdirectories, that is associated with a git repository.

Git Repository Model

- What is special about the Git Repository model?
  - Distributed version-control system
  - Tracks source code
  - Coordinates among programmers
    - Tracks changes
    - Supports non-linear workflows
  - Created in 2005 by Linus Torvalds

What is Git?

- Git is a distributed version-control system
  - Tracks changes to content
  - Provides a central point for collaboration
- Git allows for centralized administration
  - Teams have controlled access scope
  - The main branch should always correspond to deployable code
- IBM Cloud is built around open-source tools including Git repositories

What is GitHub?

- GitHub is an online hosting service for Git repositories
  - Hosted by a subsidiary of Microsoft
  - Offers free, professional and enterprise accounts
  - As of August 2019, GitHub had over 100M repositories
- What is a Repository?
  - A data structure for storing documents including application source code
  - A repository can track and maintain version-control

What is GitLab?

GitLab is:

- A DevOps platform, delivered as a single application
- Provides access to Git Repositories
- Provides source code management

GitLab

GitLab enables developers to:

- Collaborate
- Work from a local copy
- Branch and merge code
- Streamline testing and delivery with CI/CD

1. **create a new local repository using `git init`**
2. **create and add a file to the repo using `git add`**
3. **commit changes using `git commit`**
4. **create a branch using `git branch`**
5. **switch to a branch using `git checkout`**
6. **check the status of files changed using `git status`**

7. review recent commits using `git log`
8. revert changes using `git revert`
9. get a list of branches and active branch using `git branch`
10. merge changes in your active branch into another branch using `git merge`