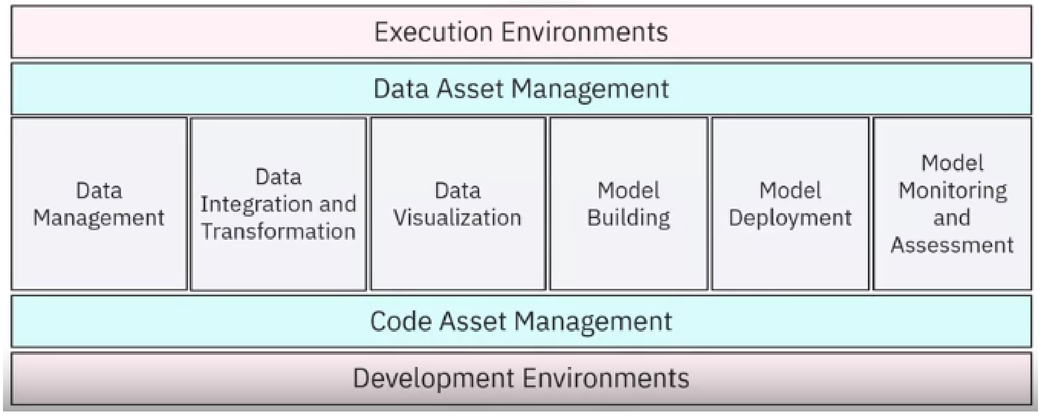
- Module 1: Overview of Data Science Tools
- Module 2: Languages of Data Science (Python, R, SQL, Java, Scala, C++, Javascript, Julia)
- Module 3: Packages, APIs, Data Sets and Models
- Module 4: Jupyter Notebooks and JupyterLab
- Module 5: RStudio and GitHub
- Module 6: Final Project and Assessment

- ❖ **MODULE 1: Overview of Data Science Tools**

Categories of Data Science Tools

- Raw data must pass through various Data Science task categories.
- 

  Execution Environments

  Data Asset Management

Data Management	Data Integration and Transformation	Data Visualization	Model Building	Model Deployment	Model Monitoring and Assessment

  Code Asset Management

  Development Environments

- Data management: process of collecting, persisting, and retrieving data securely, efficiently, and cost-effectively.
- 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. After extracting the data, the next step is to transform the data.
  - Data Transformation: process of transforming the values, structure, and format of data. Once the data is transformed, it’s time to load the data to the Data Warehouse.
- Data Visualization: graphical representation of data and information. You can use visualization to represent data in the form of charts, plots, maps, animations, etc. It’s a crucial step for understanding the data better. Model building is the next step.
- Model Building: This is where you train the data and analyze 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 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.

- Model Monitoring and Assessment: run continuous quality checks to ensure a model’s accuracy, fairness, and robustness.
    - Model monitoring: Tracks deployed models. Uses tools like Fiddler to track the performance of deployed models in a production environment.
    - Model assessment: checks for accuracy, fairness, and robustness monitoring. Uses evaluation metrics like the F1 score, true positive rate, or the sum of squared error to understand a model's performance.
- TOOLS
- Code asset management: unified view where you manage an inventory of assets
    - Developers use versioning to track and manage changes to a software project’s code
    - Collaboration allows diverse people to share and update the same project together
        - GitHub
- Data (or digital) asset management: Platform for organizing and managing data from different sources.
    - Supports replication, backup, and access right management
- Development Environments: Integrated development environments (IDEs) provide a workspace and tools to work on source code
    - They provide testing and simulation tools to emulate the real world so you can see how your code will behave after it is deployed
    - Develop, implement, execute, text and deploy
- Execution environment: has libraries for code compiling and system resources to execute and verify code
    - Cloud-based execution environments aren’t tied to specific hardware or software
    - They have tools for data preprocessing model training and deployment
- Fully-Integrated visual tools
- RECAP
    - Task categories
        - Data Management
        - Data Integration and Transformation
        - Data Visualization
        - Model Building
        - Model Deployment
        - Model Monitoring and Assessment
    - Data Science Tasks are supported by:
        - Data Asset Management
        - Code Asset Management
        - Execution Environments
        - Development Environments

Open Source Tools for Data Science - Part 1

- Data Management
    - Relational databases
        - MySQL
        - PostgreSQL
    - NoSQL databases
        - MongoDB
        - Apache CouchDB
        - Apache Cassandra
    - File-based tools
        - Hadoop file system
        - Ceph
    - Elastic search tool that stores text data
- Data integration and transformation (ETL): Also termed Data Refinery and Cleansing
    - Apache Airflow
    - KubeFlow
    - Apache Kafka
    - Apache Nifi
    - Apache SparkSQL
    - NodeRED
- Data visualization
    - Pixie Dust
    - Hue
    - Kibana
    - Apache Superset
- Model deployment: process of putting machine learning models into production
    - Apache PredictionIO
    - Seldon
    - Kubernetes
    - Redhat OpenShift
    - MLeap
    - TensorFlow
        - TensorFlow service
        - TensorFlow lite
        - TensorFlow dot JS
- Model monitoring and assessment: tools to keep track of machine learning model’s prediction performance to maintain outdated models
    - ModelDB
    - Prometheus
    - IBM AI Fairness 360
    - IBM Adversarial Robustness 360 Toolbox
    - IBM AI Explainability 360

- Code asset management: also known as version management or version control
    - Git
    - GitHub
    - GitLab
    - Bitbucket
- Data asset management: also known as data governance or data lineage
    - Apache Atlas
    - ODPi Egeria
    - Kylo

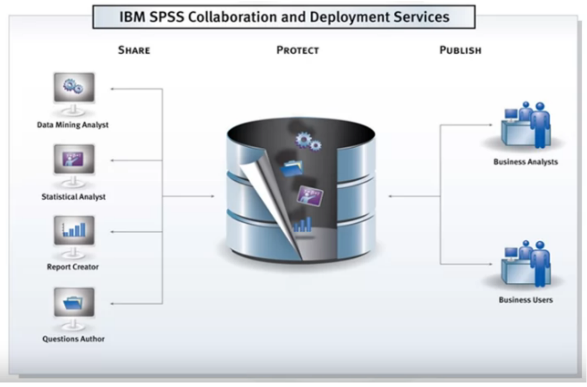
Open Source Tools for Data Science - Part 2

- Jupyter Notebook: popular development environment tool
    - Supports more than a hundred different programming languages through “kernels”
    - Encapsulates the execution environment for the different programming languages
    - Used more in the Python world
    - Key properties in single document:
        - Unify documentation
        - Code
        - Output from code
        - Shell commands
        - Visualizations
- JupyterLab: will replace Jupyter Notebooks in the long term
    - Ability to open different types of files Jupyter Notebooks, data and terminals and arrange them on the canvas
- Apache Zeppelin: inspired by Jupyter Notebooks
    - Integrated plotting ability (doesn’t require coding)
- RStudio
    - Exclusively runs R and associated libraries
    - Enables Python development
    - Provides optimal user experience when tightly integrated in the tool
    - Unifies
        - Programming
        - Execution
        - Debugging
        - Remote data access
        - Data exploration
        - Visualization
- Spyder
    - Alternative for RStudio
    - Integrates:

        - Code
        - Documentation
        - Visualization
- Apache Spark
    - One of the most used Apache products
    - Provides cluster execution environment
    - Provides linear scalability - more the number of servers in a cluster more the performance
    - Difference between Spark and Flink is it’s a batch processing engine
    - Capable of processing huge amounts of data one by one or file by file
    - Choice for most use cases
- Apache Flink
    - Stream processing image
    - Capable of processing real-time data streams
- Ray
    - Enables large scale deep learning models
- Open-source tools for data scientists
    - Fully integrated and visual (no programming knowledge is necessary)
        - Data Integration and Transformation
        - Data Visualization
        - Model Building
- KNIME
    - Drag-and-drop capabilities
    - Built-in visualization capabilities
    - Can be extended by programming R and Python
    - Has connectors to Apache Spark
- Orange
    - Less flexible than KNIME but easier to use

Commercial Tools for Data Science

- Commercial tools support the most common tasks in data science
- Data management
    - Oracle Database
    - Microsoft SQL Server
    - IBM DB2
    - Commercial supports delivered by:
        - Software vendors
        - Influential partners
        - Support networks
- Data Integration and transformation (ETL) tools
    - Informatica PowerCenter
    - IBM InfoSphere DataStage
    - These are followed by

    - SAP
    - Oracle
    - SAS
    - Talend
    - Microsoft products
    - Watson Studio Desktop
- Data visualization
  - Tableau
  - Microsoft Power BI
  - IBM Cognos Analytics
- Model building
  - SPSS Modeler (available in Watson Studio Desktop)
  - SAS enterprise miner
- Model deployment
  - 

- Model monitoring and code asset management
  - Open source is the first choice
    - Git and GitHub
- Data asset management
  - Functions include
    - Data governance: crucial part of enterprise-grade data science
    - Data versioned and annotated
    - Data dictionary: facilitates the discovery of data assets
    - Data lineage: allowing tracking back the transformation steps in creating the data assets. Includes a reference to the actual source data
    - Data privacy and retention
  - Informatica and IBM provide tools for these tasks
- Development environments
  - Fully integrated development environment for data scientists
  - Combines Jupyter Notebooks with graphical tools
- Fully integrated visual tools
  - Watson Studio
  - Watson Open Scale
  - H20 Driverless AI

Cloud Based Tools for Data Science

- Fully integrated visual tools and platforms
    - Large-scale execution of data science workflows happens in compute clusters:
        - Composed of multiple server machines
    - Microsoft Azure Machine Learning
    - H2O Driverless AI
        - A product that you download and install
        - One-click deployment for the common cloud service providers
        - Cloud provider does not do operations and maintenance
- Data management
    - Comprises software as a service (SaaS) versions of existing open source and commercial tools
        - The cloud provider operates the tool for you in the cloud
    - Example: Amazon Web Services DynamoDB, which is a NoSQL database
        - Allows storage and retrieving data in a key-value or a document store format (like JSON)
    - Cloudant (in the background, it is based on the open-source Apache CouchDB)
        - The advantage is that complex operational tasks like updating, backup, restoring, and scaling are done by the cloud provider. However, the Cloudant service offering is compatible with CouchDB. Therefore, the application migrates to another CouchDB server without making any changes to the application.
- Data integration and transformation (ETL and ELT tools)
    - Transformation steps are pushed toward the domain of the data scientist or data engineer
    - Informatica is widely used
    - Data Refinery
        - Is part of IBM Watson Studio
        - Allows transforming large amounts of raw data into consumable quality information in a spreadsheet like interface
- Data visualization
    - An example of a smaller company offering a cloud-based data visualization tool is Datameer
    - IBM Cognos Business intelligence suite
    - Various visualizations depict data for better understanding
- Model building
    - Google Cloud: AI platform training
- Model deployment
    - Tightly integrated into the model-building process

    - Commercial software can export models in an open format, such as PMML
- Model monitoring and assessment
    - A cloud tool to monitor deployed machine learning and deep learning models continuously
    - Amazon SageMaker Model Monitor is an example

MODULE 1 SUMMARY

- The Data Science Task Categories include:
    - Data Management - storage, management and retrieval of data
    - Data Integration and Transformation - streamline data pipelines and automate data processing tasks
    - Data Visualization - provide graphical representation of data and assist with communicating insights
    - Modeling - enable Building, Deployment, Monitoring and Assessment of Data and Machine Learning models
- Data Science Tasks support the following:
    - Code Asset Management - store & manage code, track changes and allow collaborative development
    - Data Asset Management - organize and manage data, provide access control, and backup assets
    - Development Environments - develop, test and deploy code
    - Execution Environments - provide computational resources and run the code
- The data science ecosystem consists of many open source and commercial options, and include both traditional desktop applications and server-based tools, as well as cloud-based services that can be accessed using web-browsers and mobile interfaces.
- **Data Management Tools:** include Relational Databases, NoSQL Databases, and Big Data platforms:
    - MySQL, and PostgreSQL are examples of Open Source Relational Database Management Systems (RDBMS), and IBM Db2 and SQL Server are examples of commercial RDBMSes and are also available as Cloud services.
    - MongoDB and Apache Cassandra are examples of NoSQL databases.
    - Apache Hadoop and Apache Spark are used for Big Data analytics.
- **Data Integration and Transformation Tools:** include Apache Airflow and Apache Kafka.
- **Data Visualization Tools:** include commercial offerings such as Cognos Analytics, Tableau and PowerBI and can be used for building dynamic and interactive dashboards.
- **Code Asset Management Tools:** Git is an essential code asset management tool. GitHub is a popular web-based platform for storing and managing source

code. Its features make it an ideal tool for collaborative software development, including version control, issue tracking, and project management.

- **Development Environments:** Popular development environments for Data Science include Jupyter Notebooks and RStudio.
    - Jupyter Notebooks provides an interactive environment for creating and sharing code, descriptive text, data visualizations, and other computational artifacts in a web-browser based interface.
    - RStudio is an integrated development environment (IDE) designed specifically for working with the R programming language, which is a popular tool for statistical computing and data analysis.

- ❖ **MODULE 2: Languages of Data Science**

Languages

- Select a language to learn depending on your needs
- Python, R, SQL, Scala, Java, C++ and Julia are the most popular languages
- JavaScript, PHP, Go, Ruby and Visual Basic all have their own unique use cases as well
- Roles In Data Science:
    - Business Analyst, Database Engineer, Data Analyst, Data Engineer, Data Scientist, Research Scientist, Software Engineer, Statistician, Product Manager, Project Manager, and so on.

Introduction to Python

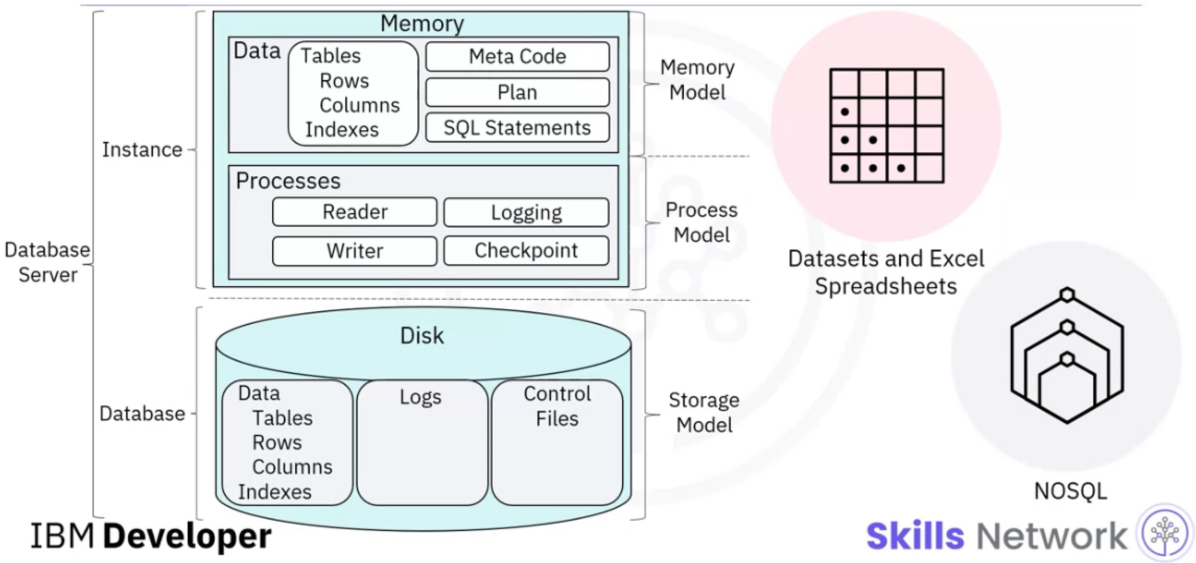
- The most used language in Data Science
- Who is Python for?
    - People who already know how to program
    - People who want to learn to program
    - Over 80% of data professionals worldwide
    - Areas like data science, AI and machine learning, web development, and Internet of Things (IoT) with devices like Raspberry Pi
    - Large organizations like IBM, Wikipedia, Google, Yahoo!, CERN, NASA, Facebook, Amazon, Instagram, Spotify, and Reddit
- What makes Python great?
    - Is a general-purpose language
    - Has a large standard library (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 AI, it has TensorFlow, PyTorch, Keras, and Scikit-learn.
    - Python can also be used for Natural Language Processing (NLP) using the Natural Language Toolkit (NLTK).

Introduction to R language

- R is free software
- Open source vs. free software
    - Similarities
        - ■ Both are free to use
        - ■ Both commonly refer to the same set of licenses
        - ■ Both support collaboration
        - ■ In many cases these terms can be used interchangeably (but not all)
    - Differences:

        - Open Source Initiative (OSI) champions open source while the Free Software Foundation (FSF) defines free software
        - Open Source is more business focused while free software is more focused on a set of values
- Who is R for?
    - Statisticians, mathematicians, and data miners use R to develop statistical software, graphing, and data analysis.
    - Someone with no or minimal programming background
    - For learners with a data science career
    - R is popular in academia
    - Companies like IBM, Google, Facebook, Microsoft, Bank of America, Ford, TechCrunch, Uber and Trulia
- What makes R great?
    - Is the largest repository of statistical knowledge
    - Has more than 15,000 publicly released packages to conduct complex exploratory data analysis
    - Integrates well with other computer languages like C++, Java, C, .Net and Python
    - Common mathematical operations like matrix multiplication give immediate results
    - Has stronger object-oriented programming facilities

Introduction to SQL

- SQL = Structured Query Language
- Simple and powerful
- Relational databases
- 

- It is subdivided into several language elements, including
    - Clauses, Expressions, Predicates, Queries and Statements

- What makes SQL great?
    - Helps you get jobs in data science and data engineering
    - Speeds up workflow executions
    - Acts as an interpreter between you and the database
    - Is an American National Standards Institute standard
    - Enables you to apply your SQL knowledge with other databases
- Many SQL databases available
    - MySQL, IBM DB2, PostgreSQL, Apache Open Office Base, SQLite, Oracle, MariaDB, Microsoft SQL Server, and more.
- If you want to learn SQL
    - You should focus on a specific relational database and then plug into the community for that specific platform.
    - In addition, there are many available great introductory courses on SQL!

Other Languages for Data Science

- 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
- Scala
    - A general-purpose programming language that provides support for functional programming
    - Designed as an extension to Java, it is interoperable 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
- 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 redefined parallelism
    - A young language with a lot of promise
    - For Data Science:
        - JuliaDB, a package for working with large persistent data sets

MODULE 2 SUMMARY

- You should select a language to learn depending on your needs, the problems you are trying to solve, and whom you are solving them for.
- The popular languages are Python, R, SQL, Scala, Java, C++, and Julia.
- For data science, you can use Python's scientific computing libraries like Pandas, NumPy, SciPy, and Matplotlib.
- Python can also be used for Natural Language Processing (NLP) using the Natural Language Toolkit (NLTK).
- Python is open source, and R is free software.
- R language’s array-oriented syntax makes it easier to translate from math to code for learners with no or minimal programming background.
- SQL is different from other software development languages because it is a non-procedural language.
- SQL was designed for managing data in relational databases.
- If you learn SQL and use it with one database, you can apply your SQL knowledge with many other databases easily.

- Data science tools built with Java include Weka, Java-ML, Apache MLlib, and Deeplearning4.
- For data science, a popular program built with Scala is Apache Spark which includes Shark, MLlib, GraphX, and Spark Streaming.
- Programs built for Data Science with JavaScript include TensorFlow.js and R-js.
- One great application of Julia for Data Science is JuliaDB.

- ❖ **MODULE 3: Packages, APIs, Datasets and Models**

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
- Scientifics Computing Libraries in Python
  - Libraries contain built-in modules providing different functionalities, which you can use directly, also called frameworks
  - Examples:
    - Pandas (Data structures & tools)
    - NumPy (Arrays & matrices)
- Visualization Libraries in Python
  - Use data visualization libraries to communicate with others and display meaningful results of an analysis
  - Examples:
    - Matplotlib (plots & graphs, most popular)
    - Seaborn (plots: heat maps, time series, and violin plots)
- Machine Learning and Deep Learning Libraries in Python
  - Scikit-learn (Machine Learning: regression, classification, clustering)
  - Keras (Deep Learning Neural Networks)
- Deep Learning Libraries in Python
  - TensorFlow (Deep Learning: Production and Deployment)
  - PyTorch (Deep Learning: regression classification)
- Apache Spark
  - General-purpose cluster-computing framework:
    - Pandas
    - NumPy
    - Scikit-learn
  - Data processing jobs:
    - Python
    - R
    - Scala
    - SQL
- Scala libraries
  - Vegas: for statistical data visualizations
  - Big DL for deep learning

- R libraries
    - ggplot2 for data visualization in R
    - Libraries that allow you to interface with Keras and TensorFlow

Application Programming Interfaces (APIs)

- What is an API?
    - It is the part of the library you see while the library contains all the components of the program
    - Allows communication between two pieces of softwares
        - Your program | Inputs (API) and outputs (Data) | Other software component
- API library
    - Your program | Inputs (Pandas Object) and outputs (Data) | Other software component
- Other languages API
    - TensorFlow (C++) with Python, JavaScript, C++, Java and Go
    - Julia, Matlab, R, Scala, etc.
- REST APIs (Representational State Transfer APIs)
    - Allow you to communicate through the internet
    - Enable you to use resources like storage, data, and artificially intelligent algorithms
    - Used to interact with web services
    - Have a set of rules regarding:
        - Communication
        - Input or Request
        - Output or Response
- Common terms
    - You or your code is the client.
    - The web service is the resource, and the client finds the service via an endpoint.
    - The client sends requests to the resource and receives a response from the resource.
- Example: HTTP
    - Data is transmitted over the internet using HTTP methods. The REST APIs get all the information from the request sent by the client.
    - The request is sent using an HTTP message that contains a JSON file. The file contains instructions for what operation is to be performed by the web service.
    - This operation is transmitted to the web service via the internet and the server performs the operation, and the service performs the operation.
    - Similarly, the web service returns a response through an HTTP message, where the information is returned using a JSON file. And this information is transmitted back to the client.

Data Sets - Powering Data Science

- What is a data set?
    - Collection of data
    - Data structures
        - Tabular data (CSV or Comma Separated Values)
        - Hierarchical data, network data
        - Raw files (images or audio, MNIST)
- Data ownership
    - Private data
        - Confidential
        - Private or personal information
        - Commercially sensitive
    - Open data: enables data scientists, researchers, analysts and others to uncover previously unknown and valuable insights
        - Publicly available
        - Companies
        - Scientific institutions
        - Government
        - Organizations
- Where to find open data
    - Open data portal list from around the world
        - datacatalogs.org
    - Governmental, intergovernmental and organization websites
        - data.un.org
        - data.gov
        - europeandataportal.eu/en/
    - Kaggle
        - kaggle.com/data_sets
    - Google dataset search
- Community Data License Agreement
    - Collaborative licenses to enable access, sharing and use of data openly among individuals and organizations
    - cdla.io (Linux)

Additional Sources of Datasets: When you select a dataset, it is necessary to look into the license. A license explains whether you can use that dataset or not; or explains if you have to accept certain guidelines to use that dataset. The different license types are listed below.

- **PUBLIC DOMAIN MARK - PUBLIC DOMAIN**

    - When a dataset has a Public Domain license, all the rights to use, access, modify and share the dataset are open to everyone. Here there is technically no license.
- **OPEN DATA COMMONS PUBLIC DOMAIN DEDICATION AND LICENSE – PDDL**
    - Open Data Commons license has the same features as the Public Domain license, but the difference is the PDDL license uses a licensing mechanism to give the rights to the dataset.
- **CREATIVE COMMONS ATTRIBUTION 4.0 INTERNATIONAL CC-BY**
    - This license allows users to share and modify a dataset, but only if they give credit to the creator(s) of the dataset.
- **COMMUNITY DATA LICENSE AGREEMENT – CDLA PERMISSIVE-2.0**
    - Like most open-source licenses, this license allows users to use, modify, adapt, and share the dataset, but only if a disclaimer of warranties and liability is also included.
- **OPEN DATA COMMONS ATTRIBUTION LICENSE - ODC-BY**
    - This license allows users to share and adapt a dataset, but only if they give credit to the creator(s) of the dataset.
- **CREATIVE COMMONS ATTRIBUTION-SHAREALIKE 4.0 INTERNATIONAL - CC-BY-SA**
    - This license allows users to use, share, and adapt a dataset, but only if they give credit to the dataset and show any changes or transformations they made to the dataset. Users might not want to use this license because they have to share the work they did on the dataset.
- **COMMUNITY DATA LICENSE AGREEMENT – CDLA-SHARING-1.0**
    - This license uses the principle of ‘copyleft’: users can use, modify, and adapt a dataset, but only if they don’t add license restrictions on the new work(s) they create with the dataset.
- **OPEN DATA COMMONS OPEN DATABASE LICENSE - ODC-ODBL**
    - This license allows users to use, share, and adapt a dataset but only if they give credit to the dataset and show any changes or transformations they make to the dataset. Users might not want to use this license because they have to share the work they did on the dataset.
- **CREATIVE COMMONS ATTRIBUTION-NONCOMMERCIAL 4.0 INTERNATIONAL - CC BY-NC**
    - This license is a restrictive license. Users can share and adapt a dataset, provided they give credit to its creator(s) and ensure that the dataset is not used for any commercial purpose.
- **CREATIVE COMMONS ATTRIBUTION-NO DERIVATIVES 4.0 INTERNATIONAL - CC BY-ND**

    - This license is also a restrictive license. Users can share a dataset if they give credit to its creator(s). This license does not allow additions, transformations, or changes to the dataset.
- **CREATIVE COMMONS ATTRIBUTION-NONCOMMERCIAL-SHAREALIKE 4.0 INTERNATIONAL - CC BY-NC-SA**
    - This license allows users to share a dataset only if they give credit to its creator(s). Users can share additions, transformations, or changes to the dataset, but they cannot use the dataset for commercial purposes.
- **CREATIVE COMMONS ATTRIBUTION-NONCOMMERCIAL-NODERIVATIVES 4.0 INTERNATIONAL - CC BY-NC-ND**
    - This license allows users to share a dataset only if they give credit to its creator(s). Users are not allowed to modify the dataset and are not allowed to use it for commercial purposes.

Sharing Enterprise Data - Data Asset eXchange

- Curated collection of data sets:
    - IBM Research data sets
    - Trusted 3rd party data sets
    - Ready for use in enterprise applications
- Data Science friendly licenses
- Tutorial notebooks
    - Data cleaning
    - Pre-processing
    - Exploratory analysis
- Advanced notebooks
    - Creating charts
    - Training machine learning models
    - Integrating deep learning (MAX)
    - Statistical analysis
    - Time-series analysis
- developer.ibm.com
    - Open source at IBM —> Data Asset eXchange
- Recap
    - Contains high-quality open data sets
    - DAX open data sets include tutorial notebooks that provide basic and advanced walk thoughts for developers
    - DAX and MAX are available on the IBM Developer website
    - You can get, run, and preview data sets and notebooks on DAX
    - DAX notebooks are opened in Watson Studio

Machine Learning Models – Learning from Models to Make Predictions

- 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 Learning, Unsupervised and Reinforcement
    - 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
    - Unsupervised Learning
        - Data is not labeled
        - Model tries to identify patterns without external help
            - Examples: 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 the way the human brain solves problems
        - Applications
            - Natural Language Processing
            - Image, audio and video analysis
            - Time series forecasting
        - Requires large datasets of labeled data and is computation intensive
        - Requires special purpose hardware
        - 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” like TensorFlow, PyTorch, Keras, and ONNX
- Using models to solve a problem
    - Prepare data: First, you collect and prepare data that will be used to train a model. Data preparation can be a time-consuming and labor-intensive process. In order to train a model to detect objects in

    images, you need to label the raw training data. For example, you can draw bounding boxes around objects and label them.
  - Build model: Next, you build a model from scratch or select an existing model that might be well suited for the task from a public or private resource.
  - Train model: You can then train the model on your prepared data. During training, your model learns from the labeled data how to identify objects that are depicted in an image.
    - Iterative process: requires data, expertise, time and resources
  - Deploy model: Once training has commenced, you analyze the training results and repeat the process until the trained model performance meets your requirements.
  - Use model: When the trained model performs as desired, you deploy it to make it available to your applications.

The Model Asset eXchange

- The Model Asset eXchange is a free open source repository for ready-to-use and customizable deep learning microservices.
- To reduce time to value, consider taking advantage of pre-trained models for certain types of problems.
- MAX model-serving microservices are built and distributed on GitHub as open source Docker images.
- Hat OpenShift is a Kubernetes platform used to automate deployment, scaling, and management of microservices.
- Ml-exchange.org has multiple predefined models. The CodePen tool lets users edit front-end languages.
- The CodePen tool lets users edit front-end languages.

<u>MODULE 3 SUMMARY</u>

- Libraries usually contain built-in modules that provide different functionalities.
- You can use data visualization methods to communicate with others and display meaningful results of an analysis.
- For machine learning, the Scikit-learn library contains tools for statistical modeling, including regression, classification, clustering, and so on.
- Large-scale production of deep-learning models use TensorFlow, a low-level framework.
- Apache Spark is a general-purpose cluster-computing framework that allows you to process data using compute clusters.
- An application programming interface (API) allows communication between two pieces of software.
- API is the part of the library you see while the library contains all the components of the program.

- REST APIs allow you to communicate through the internet and take advantage of resources like storage, data, artificially intelligent algorithms, and much more.
- Open data is fundamental to Data Science.
- Community Data License Agreement makes it easier to share open data.
- The IBM Data Asset eXchange (DAX) site contains high-quality open data sets.
- DAX open data sets include tutorial notebooks that provide basic and advanced walk-throughs for developers.
- DAX notebooks open in Watson Studio.
- Machine learning (ML) uses algorithms – also known as "models" – to identify patterns in the data.
- Types of ML are Supervised, Unsupervised, and Reinforcement.
- Supervised learning comprises two types of models, regression and classification.
- Deep learning refers to a general set of models and techniques that loosely emulate the way the human brain solves a wide range of problems.
- The Model Asset eXchange is a free, open-source repository for ready-to-use and customizable deep-learning microservices.
- MAX model-serving microservices are built and distributed on GitHub as open-source Docker images.
- You can use Red Hat OpenShift, a Kubernetes platform, to automate deployment, scaling, and management of microservices.
- Ml-exchange.org has multiple predefined models.

<u>GRADED QUIZ</u>

- **Which library is used for Machine Learning?** Scikit-learn.
- **Which deep learning library in Python is used for experimentation?** PyTorch.
- **Which API can be used with TensorFlow?** Julia
- **What does T stand for in REST?** Transfer
- **Which of the following data sets is considered open data?** Government data.
- **Which license stipulates that the modified version of the data should be published under the same license terms as the original data?** CDLA-Sharing
- **Which tab on the IBM developer web page enables you to open the Data Asset eXchange page?** Open Source at IBM.
- **Which tab in the Data Asset eXchange project page enables you to view all the Jupyter Notebooks?** Assets.
- **Which machine learning model is used to solve regression and classification problems?** Supervised Learning

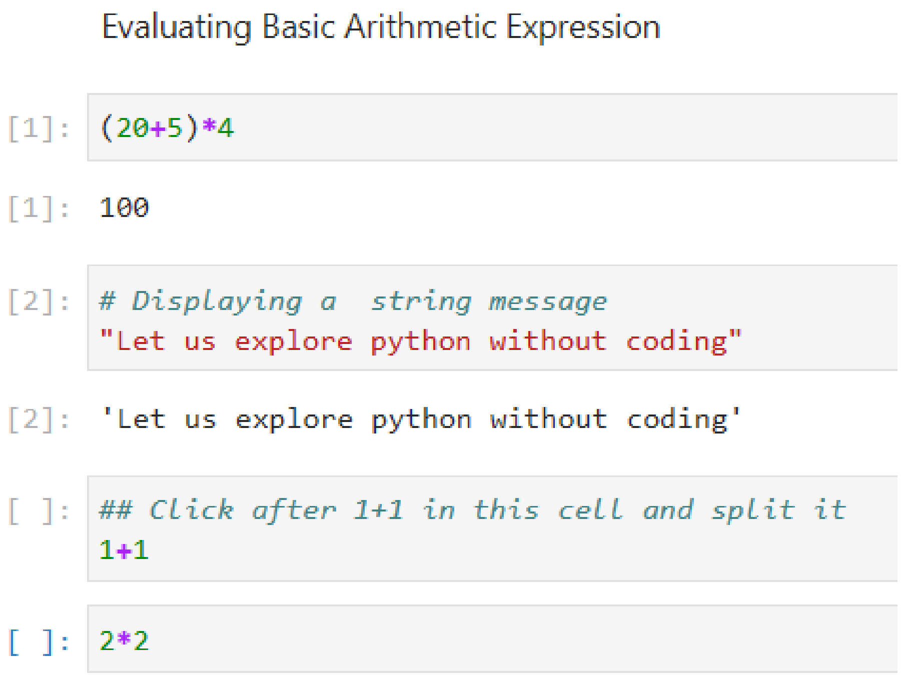
- **On the MAX object detector page, which online tool is used by developers to edit front-end languages?** CodePen

❖ MODULE 4: Jupyter Notebooks and JupyterLab

Introduction to Jupyter Notebooks

- Jupyter Notebooks
  - Is a browser-based application that allows you to create and share documents containing code, equations, visualizations, narrative text links, and more.
  - Records Data Science experiments
  - Allows combining text, code blocks, and code output in a single file
    - When you run the code, it generates the output, including plots and tables, within the notebook file.
  - Exports the notebook to a PDF or HTML file format
- JupyterLab
  - Allows access to multiple Jupyter Notebooks files, other code, and data files
  - Enables working in an integrated manner
  - Is compatible with several file formats like CSV, JSON, PDF, Vega, and so on
  - Is an open source
  - Can be used with cloud-based services like IBM and Google Colab (they don't require any installation on your local machine)

Getting Started with Jupyter Notebooks



Evaluating Basic Arithmetic Expression

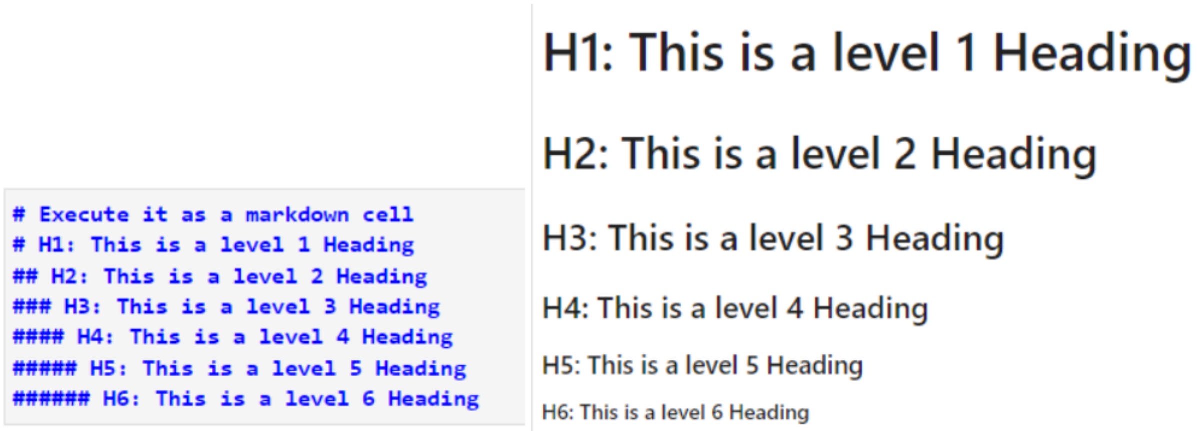
```
[1]: (20+5)*4

[1]: 100

[2]: # Displaying a  string message
     "Let us explore python without coding"

[2]: 'Let us explore python without coding'

[ ]: ## Click after 1+1 in this cell and split it
     1+1

[ ]: 2*2
```

Jupyter Kernels

- What is a kernel?
  - Is a computational engine that executes the code contained in a Notebook file
  - Exists for many languages
  - Launches when a Jupyter Notebook is opened
  - Performs the computation when the Notebook is executed
- Python kernel
  - You may need to install other notebook languages in your Jupyter environment
  - You can switch to a different kernel as per your requirement

Using Markdowns in Jupyter Notebooks

Headings



```
# Execute it as a markdown cell
# H1: This is a level 1 Heading
## H2: This is a level 2 Heading
### H3: This is a level 3 Heading
#### H4: This is a level 4 Heading
##### H5: This is a level 5 Heading
###### H6: This is a level 6 Heading
```

H1: This is a level 1 Heading

H2: This is a level 2 Heading

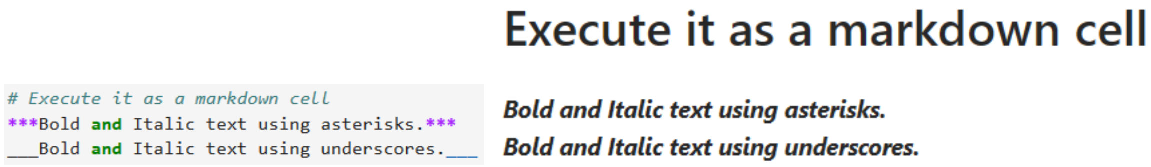
H3: This is a level 3 Heading

H4: This is a level 4 Heading

H5: This is a level 5 Heading

H6: This is a level 6 Heading

Bold and italic



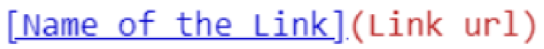
```
# Execute it as a markdown cell
***Bold and Italic text using asterisks.***
___Bold and Italic text using underscores.___
```

Execute it as a markdown cell

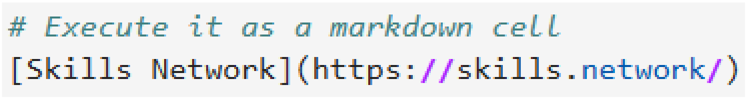
Bold and Italic text using asterisks.

Bold and Italic text using underscores.

Hyperlinks

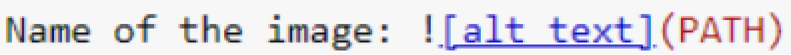


```
[Name of the Link](Link url)
```

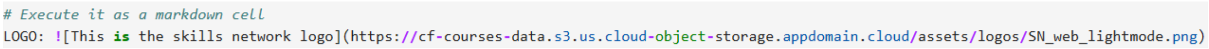


```
# Execute it as a markdown cell
[Skills Network](https://skills.network/)
```

Images

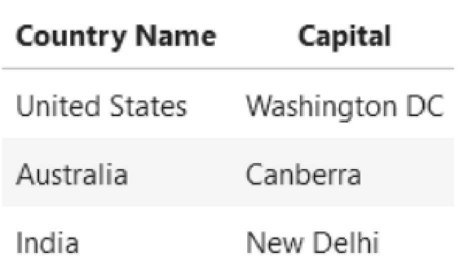


```
Name of the image: ![alt text](PATH)
```

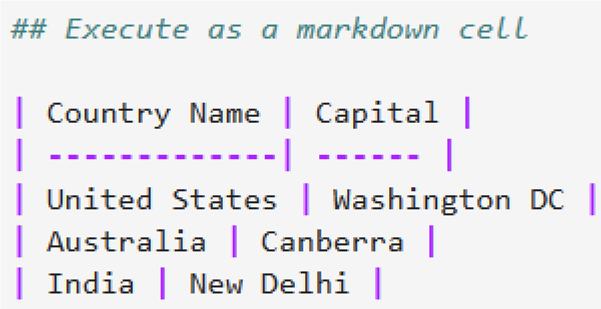


```
# Execute it as a markdown cell
LOGO: ![This is the skills network logo](https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/assets/logos/SN_web_lightmode.png)
```

Tables

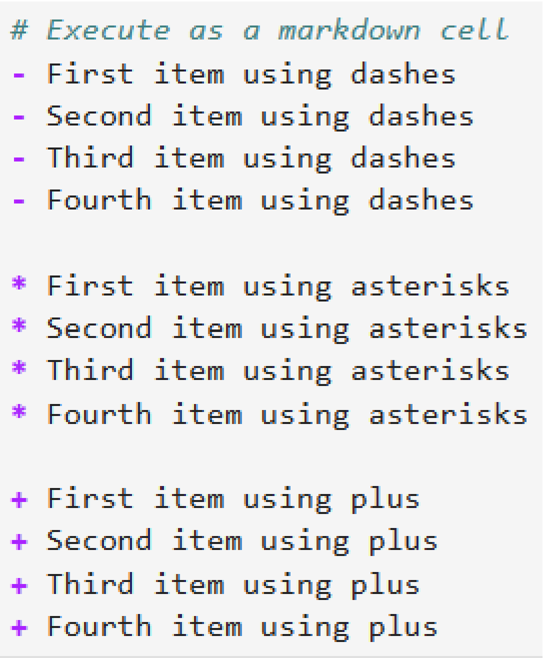


Country Name	Capital
United States	Washington DC
Australia	Canberra
India	New Delhi

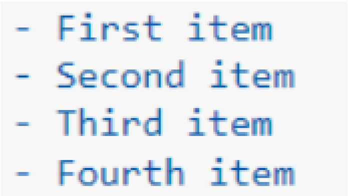


```
## Execute as a markdown cell

| Country Name | Capital |
| -------------| ------ |
| United States | Washington DC |
| Australia | Canberra |
| India | New Delhi |
```

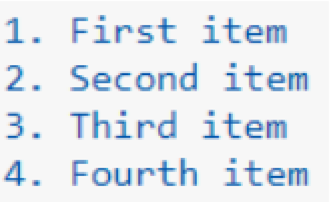
Ordered and unordered (bulleted) lists



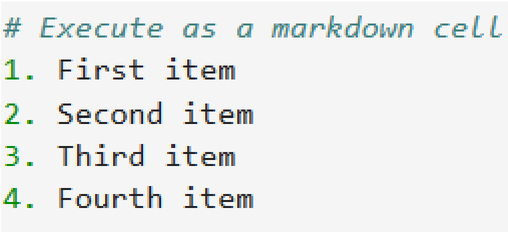
```
# Execute as a markdown cell
- First item using dashes
- Second item using dashes
- Third item using dashes
- Fourth item using dashes

* First item using asterisks
* Second item using asterisks
* Third item using asterisks
* Fourth item using asterisks

+ First item using plus
+ Second item using plus
+ Third item using plus
+ Fourth item using plus
```



```
- First item
- Second item
- Third item
- Fourth item
```



```
1. First item
2. Second item
3. Third item
4. Fourth item
```



```
# Execute as a markdown cell
1. First item
2. Second item
3. Third item
4. Fourth item
```

Jupyter Architecture

- Architecture

  - Implements a two-process model with a kernel and a client.
  - The Notebook server is responsible for saving and loading the notebooks
  - The kernel executes the cells of code contained in the Notebook
  - The Jupyter architecture uses the NB convert tool to convert files to other formats

Additional Anaconda Jupyter Environments

- Computational notebooks
  - Combine code, computational output, explanatory text, and multimedia resources in a single document
- JupyterLab
  - An open-source, web-based application
  - Enables the creation of code, interactive visualizations, text, and equations
  - Includes pre-installed Python libraries
    - NumPy
    - Pandas
    - Matplotlib
- 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 command line interface (CLI)
  - Download Anaconda Navigator at 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 environments tools

Additional Cloud Based Jupyter Environments

- Computational notebooks
  - They combine code, computational output, explanatory text, and multimedia resources in a single document
- JupyterLite
  - Lightweight tool build 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
- Google Collaboratory (GoogleColab)
    - Free Jupyter notebook environment that runs environment 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)

MODULE 4 SUMMARY

- Jupyter Notebooks are used in Data Science for recording experiments and projects.
- Jupyter Lab is compatible with many files and Data Science languages.
- There are different ways to install and use Jupyter Notebooks.
- How to run, delete, and insert a code cell in Jupyter Notebooks.
- How to run multiple notebooks at the same time.
- How to present a notebook using a combination of Markdown and code cells.
- How to shut down your notebook sessions after you have completed your work on them.
- Jupyter implements a two-process model with a kernel and a client.
- The notebook server is responsible for saving and loading the notebooks.
- The kernel executes the cells of code contained in the Notebook.
- The Jupyter architecture uses the NB convert tool to convert files to other formats.
- Jupyter implements a two-process model with a kernel and a client.
- The Notebook server is responsible for saving and loading the notebooks.
- The Jupyter architecture uses the NB convert tool to convert files to other formats.
- The Anaconda Navigator GUI can launch multiple applications on a local device.
- Jupyter environments in the Anaconda Navigator include JupyterLab and VS Code.
- You can download Jupyter environments separately from the Anaconda Navigator, but they may not be configured properly.
- The Anaconda Navigator GUI can launch multiple applications.
- Additional open-source Jupyter environments include JupyterLab, JupyterLite, VS Code, and Google Colaboratory.
- JupyterLite is a browser-based tool.

- ❖ **MODULE 5: RStudio & GitHub**

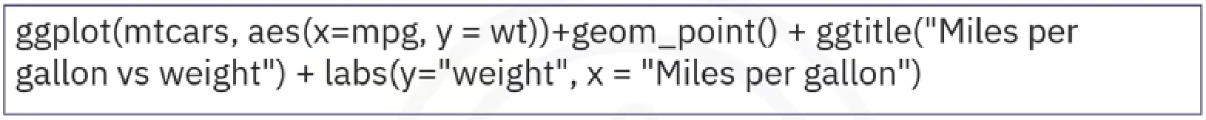
Introduction to R and RStudio

- 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 (SPSS and SATA)
- 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
    - Is an Integrated Development Environment (IDE)
    - It increases productivity in running R programming language
    - It includes:
        - Code editor
        - Console
        - Workspace History Tab
        - Files, Plots, Packages and Help
- Popular R Libraries for Data Science
    - dplyr for Data Manipulation
    - stringr for String Manipulation
    - ggplot for Data Visualization
    - caret for Machine Learning

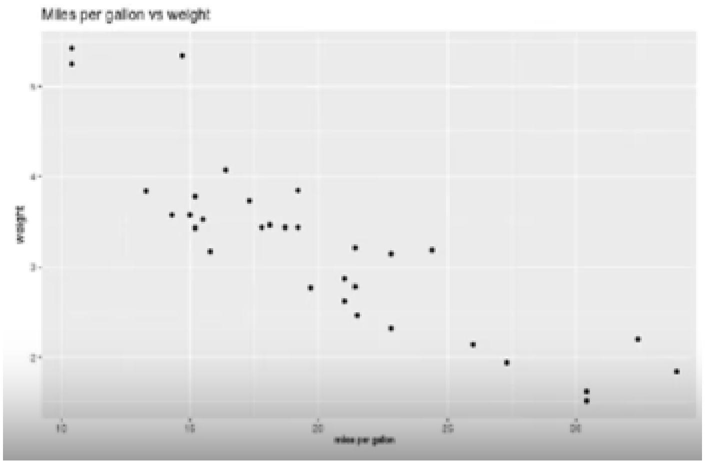
Plotting in RStudio

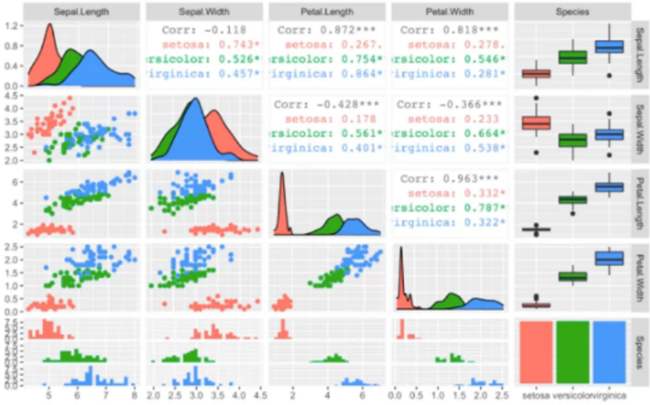
- Using data visualization in R
    - ggplot: Histograms, bar charts, scatterplots
    - Plotly: Web-based data visualizations
    - Lattice: Complex, multivariable data sets
    - Leaflet: Interactive plots

Using ggplot

- ggplot adds layers of functions and arguments
- 

```
ggplot(mtcars, aes(x=mpg, y = wt))+geom_point() + ggtitle("Miles per
gallon vs weight") + labs(y="weight", x = "Miles per gallon")
```

- 

- GGally extends ggplot by adding several functions to reduce the complexity of combining geometric objects with transformed data.
- 

Overview of Git/GitHub

- Git
  - Free and open source software
  - Distributed version control system
  - Accessible anywhere in the world
  - One of the most common version control systems available
  - Can also version control images, documents, etc.
- SHORT Glossary of Terms
  - 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

- Basic Git Commands
    - init: When starting out with a new repository, you only need to create it once: either locally, and then push to GitHub, or by cloning an existing repository by using the command "git init".
    - add: moves changes from the working directory to the staging area
    - status: allows you to see the state of your working directory and the staged snapshot of your changes.
    - commit: takes your staged snapshot of changes and commits them to the project
    - reset: undoes changes that you've made to the files in your working directory.
    - log: enables you to browse previous changes to a project
    - branch: lets you create an isolated environment within your repository to make changes.
    - checkout: lets you see and change existing branches.
    - merge: lets you put everything back together again.
    - clone: clone an existing repository
- Go to try.github.io to download the cheat sheets and run through the tutorials

Introduction to GitHub

- Git Repository Model
    - Distributed version-control system
    - Tracks source code
    - Coordinates among programmers
        - Track 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 build around open-source tools including Git repositories
- 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
- GitLab
    - Is a DevOps platform, delivered as a single application
    - Provides access to Git Repositories
    - Provides source code management
    - Enables developers to:
        - Collaborate
        - Work from a local copy
        - Branch and merge code
        - Streamline testing and delivery with CI/CD

GitHub - Working with Branches

- A branch is a snapshot of your repository
    - Master branch is the official version of the project
- Why create a branch?
    - Edits and changes are made in the child branch
    - Tests are done to ensure quality before merging with the Master branch
- Merging multiple branches
    - Branches allow for simultaneous development and testing by multiple team members
- Pull Request (PR)
    - They are a way of proposing changes to the main branch
    - Other team members review the change and approve the merging to the master branch

GitHub Branches

- Branches store all files in GitHub
- The master branch stores the deployable code
- Create a new branch for planned changes
- Merging Branches
    - Start with a common base
    - The code is branched while new features are developed
    - Both branches are undergoing changes
    - When the two streams of work are ready to merge, each branch's code is identified as a tip and the two tips are merged into a third, combined branch
- Make a commit
    - Saved changes are called commits
    - To change the contents of a file
        - Select file
        - Click pencil icon
        - Make changes

        - Scroll down to find commit changes
    - In the Commit changes box, add a comment that describes the changes
    - Choose to commit directly to the current branch or to create a new branch
    - Click Commit changes
- What is Pull Request?
    - A pull request makes the proposed (committed) changes available for others to review and use
    - A pull can follow any commits, even if code is unfinished
    - Pull requests can target specific users
    - GitHub automatically makes a pull request if you make a change on a branch you do not own
    - Log files record the approval of the merge
    - Open a Pull Request
        - Click Pull request and select New pull request
        - Select the new branch from the compare box
        - Confirm that changes are what you want to assess
        - Add a title and description to the request
        - Click Create pull request
- Merging into the Master Branch
    - The master branch should be the only deployed code
    - Developers can change source files in a branch but the changes are not released until
        - They are committed
        - A pull command is issued
        - The code is reviewed and approved
        - The approved code is merged back into the master code
    - Merge a Pull Request
        - Click Merge pull request
        - Click Confirm merge

MODULE 5 SUMMARY

- The capabilities of R and its uses in Data Science.
- The RStudio interface for running R codes.
- Popular R packages for Data Science.
- Popular data visualization packages in R.
- Plotting with the inbuilt R plot function.
- Plotting with ggplot.
- Adding titles and changing the axis names using the ggtitle and lab’s function.
- A Distributed Version Control System (DVCS) keeps track of changes to code, regardless of where it is stored.

- Version control allows multiple users to work on the same codebase or repository, mirroring the codebase on their own computers if needed, while the distributed version control software helps manage synchronization amongst the various codebase mirrors.
- Repositories are storage structures that:
    - Store the code
    - Track issues and changes
    - Enable you to collaborate with others
- Git is one of the most popular distributed version control systems.
- GitHub, GitLab and Bitbucket are examples of hosted version control systems.
- Branches are used to isolate changes to code. When the changes are complete, they can be merged back into the main branch.
- Repositories can be cloned to make it possible to work locally, then sync changes back to the original.
- Organization is a collection of user accounts that owns repositories