**MODULE 1 SUMMARY**

Congratulations! You have completed this module. At this point in the course, you know:

* 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
  + Modelling - 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**

Which language should I learn?

Python, R, SQL

**Python**

Most widely used in DS

For:  
People already know to program

Learn how to program

80% professional use

AI, DS, MK, Web development, IOT, Rasp Pi

IBM, wiki, Google, CERN, Nasa, FB, Amazon

High level general purpose rpogramming language

Large standard library

Database, web scrapping, ML, analytics

Scientific computing libraries: Pandas, Numpy, Scipy, Matplotlib

AI: TensorFlow, PyTorch, Keras, Scikit-learn

Used for Natural Language Procesing (NLP) using the the Natural Language Tookit (NLTK)

Well documented diversity and inclusion efforts (pyladies)

Clear and readable syntax

Huge global community

A picture containing text, screenshot, font

Description automatically generated**R Language**

Open source – Python

Free software – R

Why R?  
Free software

Private use

Stastician, mathematician, data miners to develop software, graphing and data analysis

Minimal programming background

Learner with a data science career

Popular in academian

IBM, Google, Facebook, Microsoft

Largest repositort of statistical knowledfe

150000 publicly released packages to conduct complex explarotary data analysis

Intergates well with other computer languages like C++ , Java, C, .Net, Python

Common methematical operations like matrix multiplication give immediate results

Has stronger object oriented prgomramming languages

Global communites: useR!, SatRdays, R-Ladies

**SQL**

SQL=structured query language

Simple and powerful

Older than Python and R by about 20 years

Developed by IBM in 1964

Useful in handling structred data

The SQL language is subdivided into Clauses, Expressions, Predicates, Queries, and Statements.

Relational databases = Dataset and Excel Spreadsheet

Been developed also for NoSQL

Helps get job in data science and data engineering

Speed up workflow executions

Acts as an interpreter between you and database

Enables you to apply your SQL knowledge in other databased

MySQL. IBMDB2, MariaDB, SQLite

Non-procedural language

Limited to querying and managing data

Managing data in relational databases

Interpreter between you and the database

Can use SQL to other databases

**Other Languages for Data Science**

**Java**

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

Weka, Java-ML, Apache Mlib, Deeplearning4, Hadoop

**Scala**

General purpose programming language that provides support for functional programming

Extension yo Java, interoperable with Java as it also runs on JVM

“Scalable Language”  
Apache Spark-Shark, Mlib, GraphX, SparkStreaming

**C++**

Extension to C

Improved processing speed, system programming, broader control over appp

Develops programs that feed data to customer in real time

TensorFlow, MangoDB (NoSQL), Caffe

**JavaScript**

General purpose language that extended beyond the brower with Node.js

Not related to Java language

TensorFlow.js make deep learning possible in Node.js and in browser

**Julia**

Designed in MIT for numerical analysis and computayional science

Provices speedy development

Executed directly on the processor

Calls C, Go, Java, Matlab, R fortran, Python libraries

Parallelism

Young language

JuliaDB – owkring with persitent large dataset

**Module 2 Summary**

Congratulations! You have completed this module. At this point in the course, you know:

* 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, 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 2  
Libraries, APIs, Datasets and Models**

Libraries: collection of functions and methods that allow you to perform many actions without writing the code.

Python Libraires:  
- Scientific computing libraries

-Visualization libraries

-High level machine learning and deep learning

Scientific computing libraries contain built-in modules providing different functionalities, which you can use directly. They are also called frameworks. For example, Pandas offers data structures and tools for effective data cleaning, manipulation, and analysis.

data visualization methods to communicate with others and display meaningful results of an analysis. These libraries enable you to create graphs, charts, and maps.

Apache Spark is a general-purpose cluster-computing framework allowing you to process data using compute clusters.

Scala libraries:  
- Vegas – statistical data visualization  
-Big DL-Deep learning  
  
R libraries:  
-ggplot2 – data visual

**Application Programming Interface (APIs)**

API : An application programming interface (API) allows communication between two pieces of software.

REST APIs: Representational State Transfer APIs

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.

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 service performs the operation.

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.

In the API call, you will send a copy of the audio file to the API; this is called a post request. Then the API will send the text transcription of what the individual is saying. At the backend, the API is making a Get request.

**Data Sets – Powering Data Science**

A data set is a structured collection of data. Data embodies information represented as text, numbers, or media such as images, audio, or video files.

A tabular data set comprises a collection of rows containing columns that store the information. One popular tabular data format is "comma separated values," or CSV.

Hierarchical or network data structures are typically used to represent relationships between data. Hierarchical data is organized in a tree-like format, whereas network data is stored as a graph.

A data set might also include raw data files, such as images or audio. The Modified National Institute of Standards and Technology (MNIST) dataset is popular for data science. It contains images of handwritten digits and is commonly used to train image processing systems.

Open data distribution and use might be restricted, as defined by certain licensing terms. Without a license for open data distribution, many data sets were shared in the past under open-source software licenses. These licenses were not designed to cover specific considerations related to the distribution and use of data sets.

Two licenses were initially created for sharing data: CDLA-Sharing and CDLA-Permissive. The CDLA-Sharing license grants you permission to use and modify the data. The license stipulates that if you publish your modified version of the data, you must do so under the same license terms as the original data. The CDLA-Permissive license also grants you permission to use and modify the data. However, you are not required to share changes to the data.

Neither license imposes any restrictions on results you might derive by using the data, which is important in data science.

**Sharing Enterprise Data – Data Asset eXchange**

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 and MAX are available on the IBM Developer website.

You can get, run, and preview data sets and notebooks on DAX, and DAX notebooks are opened in Watson Studio.

**Machine Learning Models – Learning From Models to Make Predictions**

Data contains a wealth of information that can be used to solve certain types of problems. Traditional data analysis approaches can be a person manually inspecting the data or a specialized computer program that automates the human analysis. These approaches reach their limits due to the amount of data to be analyzed or the complexity of the problem.

Machine learning (ML) uses algorithms – also known as “models” - to identify patterns in the data. The process by which the model learns these patterns from data is called “model training”.

Once a model is trained, it can then be used to make predictions. When the model is presented with new data, it tries to make predictions or decisions based on the patterns it has learned from past data. Machine Learning models can be divided into three basic classes: Supervised Learning, Unsupervised Learning, and Reinforcement Learning.

Supervised Learning:  
A human provides input data and correct outputs. The model tries to identify relationships and dependencies between the input data and the correct output.

Comprises two types of models, regression and classification. Regression models are used to predict a numeric (or “real”) value. For example, if information is given about past home sales, such as geographic location, size, number of bedrooms, and sales price, you can train a model to predict the estimated sales price for other homes with similar characteristics. Classification models are used to predict whether some information or data belongs to a category (or “class”). For example, for a set of emails along with a designation you can classify whether they are to be considered as spam or not.

Unsupervised Learning:  
The data is not labeled by a human. The models must analyze the data and try to identify patterns and structure within the data based on its characteristics.

Clustering is an example of this learning style. Clustering models are used to divide each record of a dataset into one of a similar group. An example of a clustering model could be providing purchase recommendations for an e-commerce store, based on past shopping behavior and the content of a shopping basket. Another example is anomaly detection that identifies outliers in a dataset, such as fraudulent credit card transactions or suspicious online log-in attempts.

Reinforcement Learning:  
Based on the way human beings and other organisms learn.

A reinforcement learning model learns the best set of actions to take, given its current environment, to get the most rewards over time.

Deep learning is a specialized type of machine learning. It refers to a general set of models and techniques that loosely emulate the way the human brain solves a wide range of problems. It is commonly used to analyze natural language (both spoken and text), images, audio, video, to forecast time series data and much more.

Deep learning has recently been very successful in these and other areas and hence is becoming an increasingly popular and important tool for data science. It requires large datasets of labeled data to train a model, is compute intensive, and usually requires special purpose hardware to achieve acceptable training times.

Deep Learning models are implemented using popular frameworks such as TensorFlow, PyTorch and Keras. The learning frameworks provide a Python API and many support other programming languages, such as C++ and JavaScript.

**The Model Asset eXchange**

**Module 3 Summary**

Congratulations! You have completed this module. At this point in the course, you know:

* 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.

**Module 4 Summary**

Congratulations! You have completed this module. At this point in the course, you know:

* 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**

**Introduction to R and Rstudio**

R is a statistical programming language.

It is a powerful tool for data processing and manipulation, statistical inference, data analysis, and machine learning algorithm.

Based on 2017 analysis, it was found that R is used most by academics, healthcare, and the government.

R supports importing of data from different sources like flat files, databases, web, and statistical software such as SPSS and STATA.

R capabilities:  
R is a preferred language for some data scientists because R functions are easy to use.

It is also known for producing great visualizations and contains packages to handle data analysis without the need to install additional libraries.

Rstudio:  
A popular integrated development environment for developing and running the R language source code and programs is RStudio. It improves and increases productivity with the R language. R studio includes: a syntax-highlighting editor that supports direct code execution and a place where you can keep a record of your work, a Console for typing R commands, a workspace and History tab that shows the list of R objects you created during your R session and the history of all previous commands, and finally, Files, Plots, Packages, and Help tabs.

Popular R Libraries:  
dplyr – manipulating data

Stringr – string manipulation

Ggplot – data visual

Caret – machine learning

**Plotting in R Studio**

Use the install.packages and the package name command.

Ggplot – Histograms, bar charts, scatterplots

Plotly – Webbased data visualization, saves as html

Lattice – complex, multivariable data sets

Leaflet – interactive plot

**Overview of Git/Github**

Git – Free and open source software

Distributed version control system

Accessible anywehre in the world

Github – web hosting of Git

The SSH protocol is a method for secure remote login from one computer to another.

A repository contains your project folders that are set up for version control.

A fork is a copy of a repository.

A pull request is the way you request that someone reviews and approves your changes before they become final.

A working directory contains the files and subdirectories on your computer that are associated with a Git repository.

Basic Git Commands:  
When starting out with a new repository, you only need create it once: either locally, and then push to GitHub, or by cloning an existing repository by using the command "git init".

"git add" moves changes from the working directory to the staging area.

"git status" allows you to see the state of your working directory and the staged snapshot of your changes.

"git commit" takes your staged snapshot of changes and commits them to the project.

"git reset" undoes changes that you’ve made to the files in your working directory.

"git log" enables you to browse previous changes to a project.

"git branch" lets you create an isolated environment within your repository to make changes.

"git checkout" lets you see and change existing branches.

"git merge" lets you put everything back together again.

Go to try.github.io to download the cheat sheets and run through the tutorials.

**Introduction to GitHub**

GitHub is the online hosting service for Git repositories.

Repositories store documents including application source code and enable contributors to track and maintain version-control.

What is special about the Git Repository model? Git is designed as a distributed version-control system. Primarily focused on tracking source code during development. Contains elements to coordinate among programmers, track changes, and support non-linear workflows.

**Module 5 Summary**

Congratulations! You have completed this module. At this point in the course, you know:

* 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.