**Libraries for Data Science**

Libraries are a collection of functions and methods that allow you to perform many actions without writing the code. We will focus on the following 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 Now, 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. It provides tools to work with different types of data. The primary instrument of Pandas is a two-dimensional table consisting of columns and rows, called a Data Frame. Pandas can also provide easy indexing so you can work with your data. NumPy libraries are based on arrays and matrices, allowing you to apply mathematical functions to the arrays. Pandas is built on top of NumPy. You use data visualization methods to communicate with others and display meaningful results of an analysis. These libraries enable you to create graphs, charts, and maps. The Matplotlib package is the most well-known library for data visualization. They are popular for making graphs and plots, and the graphs are easily customizable. Another high-level visualization library is Seaborn. It is based on matplotlib. This library generates heat maps, time series, and violin plots. Now, for machine learning, the Scikit-learn library contains tools for statistical modeling, including regression, classification, clustering, and so on. It is built on NumPy, SciPy, and matplotlib. It is simple to get started. In this high-level approach, you define the model and specify the parameter types you want to use. For building deep learning models, Keras allows you to build the standard deep learning model. Like Scikit, the high-level interface allows you to build models in a quick, simple manner. It can function using Graphics processing units (GPU) but in many cases, a lower-level environment is necessary for deep learning. TensorFlow is a low-level framework used in the large-scale production of deep learning models. It's designed for production and deployment but can be unwieldy for experimentation. Pytorch is used for experimentation, making it simple for researchers to test ideas. Apache Spark is a general-purpose cluster-computing framework allowing you to process data using compute clusters. The data is processed in parallel in more than one computer simultaneously. The Spark library has similar functionality to the following: Pandas, Numpy, and Scikit-learn. Apache Spark data processing jobs can be in: Python R Scala, and SQL There are many Scala libraries. Scala is predominately used in data engineering and data science. Let’s discuss some libraries that are complementary to Spark. Vegas is a Scala Library for statistical data visualizations. With Vegas, you can work with data files as well as Spark Data Frames. For deep learning, you can use big DL. R has built-in functionality for machine learning and data visualization, but there are also complementary libraries. ggplot2 is a popular library for data visualization in R. You can also use libraries that allow you to interface with Keras and TensorFlow. And R was a de-facto standard for open-source data science, but now Python will supersede it. In this video, you learned that: Libraries usually contain built-in modules providing different functionalities that can be used directly. 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. TensorFlow is a low-level framework used in large-scale production of deep learning models. And, Apache Spark is a general-purpose cluster-computing framework allowing you to process data using compute clusters.

**Application Program Interfaces (API)**

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. To further understand how an API works in a library, consider an example of the Pandas library. Pandas is a set of software components where not all components are written in Python. In your program, there is some data and a set of software components. You can use the Pandas API to process the data by communicating with the other software components. The software component at the back end can be the same, but there can be an API for different languages. Consider TensorFlow at the backend, written in C++ that can use APIs for other languages, such as: Python JavaScript C++ Java, and Go And thus, the API is just the interface. Other volunteer-developed APIs for TensorFlow are Julia Matlab R Scala And many more. So, REST APIs are another popular type of API. 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. So, let’s look at some common terms used with regards to API. 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. 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 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. Now, another example of a Rest API is Watson Text to Speech API. This API converts speech to text. 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. And finally, let’s look at another example, the Watson language-translator API. You send the text you would like to translate into Watson language-translator API. The API will translate the text and send the translation back to you. In this case, the API translates English to Spanish.

In this video, you learned an application programming interface (API) allows communication between two pieces of software, An API is the part of the library you see while the library contains all the components of the program. And REST APIs allow you to communicate through the internet and take advantage of resources like storage, data, artificially intelligent algorithms, and much more.

**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. A CSV file is a delimited text file where each line represents a row, and a comma separates data values. For example, imagine a dataset of observations from a weather station. Each row represents an observation at a given time, while each column contains information about that observation, such as the temperature, humidity, and other weather conditions. 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. For example, the connections between people on a social networking website are often represented 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. Traditionally, most data sets were private because they contained proprietary or confidential information such as customer data, pricing data, or other commercially sensitive information. These datasets are typically not shared publicly. Over time, many public and private entities such as scientific institutions, governments, organizations, and even companies have started making data sets available to the public as “open data,” providing free information. For example, the United Nations and federal and municipal governments worldwide have published many datasets on their websites, covering the economy, society, healthcare, transportation, the environment, and much more. Access to these and other open datasets enables data scientists, researchers, analysts, and others to uncover previously unknown and potentially valuable insights. They are used to create new applications for commercial purposes and the public good. They are also used to carry out further research. Open data has played a significant role in the growth of data science, machine learning, and artificial intelligence. It has allowed practitioners to hone their skills in various data sets. There are many open data sources on the internet. You can find a comprehensive list of available data portals worldwide on the Open Knowledge Foundation’s datacatalogs.org website. The United Nations, the European Union, and many other governmental and intergovernmental organizations maintain data repositories providing access to a wide range of information. On Kaggle, a popular data science online community, you can find (and contribute) data sets that might be of general interest. Google provides a search engine that might help you find data sets that could be of value to you. 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. To address the issue, the Linux Foundation created the Community Data License Agreement, or CDLA. 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. Note that neither license imposes any restrictions on results you might derive by using the data, which is important in data science. Let’s say, for example, that you are building a model that performs a prediction. If you are training the model using CDLA-licensed data sets, you are under no obligation to share the model or to share it under a specific license if you choose to share it. In this video, you’ve learned Open data is fundamental to Data Science. Community Data License Agreement makes it easier to share open data, and Open datasets might not meet enterprise requirements, due to the impact they might have on the business.

**Machine Learning Models – Learning from models to make predictions**

Now 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. The most commonly used type of machine learning is Supervised Learning. In 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. This type of learning 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. And so, you can train an algorithm to identify unsolicited emails. In 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. And the third type of learning, Reinforcement Learning, is loosely based on the way human beings and other organisms learn. So, think about a mouse in a maze. If the mouse gets to the end of the maze, it gets a piece of cheese. This is the “reward” for completing a task. The mouse learns through trial and error how to get through the maze to get as much cheese as it can. In a similar way, a reinforcement learning model learns the best set of actions to take, given its current environment, to get the most rewards over time. This type of learning has recently been very successful in beating the best human players in games such as Go, chess and popular strategy video games. 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. Now you can build a custom Deep Learning model from scratch or use pre-trained models from public model repositories. 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. You can download pre-trained state-of-the-art models from repositories that are commonly referred to as model zoos. Popular model zoos include those provided by TensorFlow, PyTorch, Keras, and ONNX. Models are also published by academic and commercial research groups. Let’s briefly outline the high-level tasks involved in building a model using an example. Assume you want to enable an application to identify objects in images by training a deep learning model. 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. 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. 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. Once training has commenced, you analyze the training results and repeat the process until the trained model performance meets your requirements. When the trained model performs as desired, you deploy it to make it available to your applications.

In this video you learned that: Machine learning (ML) uses algorithms – also known as “models” ‒ to identify patterns in the data. The process by which the model learns data patterns is called “model training”. Types of ML are Supervised, Unsupervised, and Reinforcement. Supervised learning comprises two types of models, regression and classification. And 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**

The Model Asset eXchange, or “MAX”, on the IBM Developer platform, is a free open source resource for deep learning models. The tasks needed to train a model from scratch require a large amount of data, labor, time, and resources. Because of this, time to value can be quite long. To reduce time to value, consider taking advantage of pre-trained models for certain types of problems. These pre-trained models can be ready to use right away, or they might take less time to train. Models are created by running data through a Model using compute resources and domain expertise. After research, evaluation, test, train and validate steps are complete, you will have a validated model. The Model Asset eXchange is a free open source repository for ready-to-use and customizable deep learning microservices. These microservices are configured to use pre-trained or custom-trainable deep learning models to solve common business problems. These models have been fully tested, and can be quickly deployed in local and cloud environments. All models in MAX are available under permissive open source licenses, making it easier to use them for personal and commercial purposes, which reduces the risk of legal liabilities. On MAX, you can find models for a variety of domains, including: Object detection, Image, audio, video, and text classification, Named entity recognition, Image to text translation, Human pose detection, and more. Let’s look at the components of a typical model-serving microservice. Each microservice includes a pre-trained deep learning model, code that pre-processes the input before it is analyzed by the model, code that post-processes the model output, and a standardized public API that makes the services functionality available to applications. Model-serving microservices are created by running inputs through a validated model and then applying the output to a rest API. After implement, package, document, and test steps are complete, you will have a model-serving microservice that can then be sent to a Local machine, or a Private, Hybrid, or Public cloud. MAX model-serving microservices are built and distributed as open source Docker images. Docker is a container platform that makes it easy to build and deploy applications. The Docker image source is published on GitHub and can be downloaded and customized for use in personal and commercial environments. Use the Kubernetes open source system to automate the deployment, scaling, and management of these Docker images. Red Hat OpenShift is a popular enterprise-grade Kubernetes platform. It is available on IBM Cloud, Google Cloud Platform, Amazon Web Services, and Microsoft Azure. Let’s explore some machine learning models. Go to ml-exchange.org. Here you can view and use multiple predefined models. We'll explore the predefined object detector model. This model will recognize objects in an image because it consists of: a deep convolutional net base model for image feature extraction, and added convolutional layers specialized in object detection. On the MAX object detector page, select CodePen. CodePen is an online tool used by developers to edit front-end languages like HTML, JavaScript, and CSS. You will be redirected to the CodePen page, where you can select MAX Tensorflow.js model. This model is trained to identify objects in an image and assigns each pixel of the image to a particular object. Here you can upload different images of a person, dog, cat, truck, or car. The model was previously trained on labeled images, so now it can recognize images even when they are not labeled. Select an image to see what happens when the model invokes the prediction endpoint. Click on Extract prediction. This invokes the prediction endpoint, and the image is uploaded. The prebuilt TFJS model prepares the input image for pre-processing. The deep learning model algorithm identifies the different objects in the image. It generates its response using the prediction results and returns the result to the application. You will see the existing image separated into two different images: the background image and the image of the dog. The model test is complete. You have confirmed that this model is able to identify items within an image without using predefined labels. In this video, you learned: 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. Red 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.

**Sharing Enterprise Data – Data Asset eXchange**

There are many open data sets available to the public, but it can be difficult to find data sets that are both high quality and have clearly defined license and usage terms. To help solve this challenge, IBM created the Data Asset eXchange, or "DAX”. DAX provides a curated collection of open data sets, both from IBM Research and trusted third-party sources. These data sets are ready for use in enterprise applications, with a wide variety of application types, including images, video, text, and audio. DAX aims to foster data sharing and collaboration by keeping data sets available under a Community Data License Agreement (or CDLA). DAX makes it easier for developers to get started with data sets because it provides a single place to access unique, high-quality data sets from trusted sources like IBM Research. It also provides tutorial notebooks that walk through the basics of data cleaning, pre-processing, and exploratory analysis. Certain data sets include advanced notebooks that explain how to perform more complex tasks, like creating charts, training machine-learning models, integrating deep learning via the Model Asset eXchange, and running statistical analysis and time-series analysis. The Data Asset eXchange and the Model Asset eXchange are both available on the IBM Developer website. With these resources, developers can create end-to-end analytic and machine learning workflows and to consume open data and models with confidence under clearly defined license terms. Now, let’s explore the Data Asset eXchange. Open https://developer.ibm.com/ in your web browser. Then select “Open Source at IBM” From the drop-down, select “Data Asset eXchange”. In the Data Asset eXchange, multiple open data sets are available for you to explore. Let’s say you’ve found a data set that might be very interesting to you: the “NOAA Weather Data - JFK Airport” data set, which contains data from a weather station at the John F. Kennedy Airport in New York. On this data set page, you can click Get this data set to download the NOAA data set from the cloud storage. Run data set notebooks to access the notebooks associated with the data set in Watson. and Preview the data and Notebooks to explore DAX metadata, glossary and the notebook.

Most data sets on DAX are complemented by one or more Notebooks. Click assets to view all the Jupyter Notebooks and data available. You can then click the source code to view all the notebooks associated with your NOAA project. You can execute all the notebooks in Watson studio to perform data cleaning, pre-processing, and exploratory analysis. If you are already familiar with opening the notebooks in Watson studio, you can log into your IBM Cloud account, create a project, and load all the notebooks into the project. Data sets on DAX also consist of one or more data files. Click the Data option to view the data files, available in your project. In this video, you learned that 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.