# Practical Data Science Coursera Specialization

by DeepLearning.AI

## Course #1 Analyze Datasets and Train ML Models using AutoML

Course Site

Made By: Matias Borghi

# Table of Contents

# Summary

In the first course of the Practical Data Science Specialization, you will learn foundational concepts for exploratory data analysis (EDA), automated machine learning (AutoML), and text classification algorithms. With Amazon SageMaker Clarify and Amazon SageMaker Data Wrangler, you will analyze a dataset for statistical bias, transform the dataset into machine-readable features, and select the most important features to train a multi-class text classifier. You will then perform automated machine learning (AutoML) to automatically train, tune, and deploy the best text-classification algorithm for the given dataset using Amazon SageMaker Autopilot. Next, you will work with Amazon SageMaker BlazingText, a highly optimized and scalable implementation of the popular FastText algorithm, to train a text classifier with very little code.

Practical data science is geared towards handling massive datasets that do not fit in your local hardware and could originate from multiple sources. One of the biggest benefits of developing and running data science projects in the cloud is the agility and elasticity that the cloud offers to scale up and out at a minimum cost.

The Practical Data Science Specialization helps you develop the practical skills to effectively deploy your data science projects and overcome challenges at each step of the ML workflow using Amazon SageMaker. This Specialization is designed for data-focused developers, scientists, and analysts familiar with the Python and SQL programming languages and want to learn how to build, train, and deploy scalable, end-to-end ML pipelines - both automated and human-in-the-loop - in the AWS cloud.

## Week 1: Explore the Use Case and Analyze the Dataset

**Ingest, explore, and visualize a product review data set for multi-class text classification.**

## Learning Objectives

- **Describe the discipline of practical data science in the cloud**
- **Define the task and use case: Multi-class classification for sentiment analysis of product reviews**
- **Ingest and explore data**
- **Analyze data using visualizations**

## Week 2: Data Bias and Feature Importance

**Determine the most important features in a data set and detect statistical biases.**

## Learning Objectives

- **Describe the concept of data bias and compare popular bias metrics**
- **Demonstrate how to detect data bias**
- **Understand feature importance**

## Week 3: Use Automated Machine Learning to train a Text Classifier

**Inspect and compare models generated with automated machine learning (AutoML).**

## Learning Objectives

- **Describe the concept of Automated Machine Learning (AutoML)**
- **Discuss how SageMaker Autopilot uniquely implements AutoML**
- **Demonstrate how to train a text classifier with AutoML**

## Week 4: Built-in algorithms

**Train a text classifier with BlazingText and deploy the classifier as a real-time inference endpoint to serve predictions.**

## Learning Objectives

- **Summarize why and when to choose built-in algorithms**
- **Describe use case and algorithms**
- **Understand the evolution of text analysis algorithms**
- **Discuss word2vec, FastText and BlazingText algorithms**
- **Transform raw review data into features to train a text classifier**
- **Apply the Amazon SageMaker built-in BlazingText algorithm to train a text classifier**
- **Deploy the text classifier and make predictions**

# Week 1: Explore the Use Case and Analyze the Dataset

## Introduction to Practical Data Science

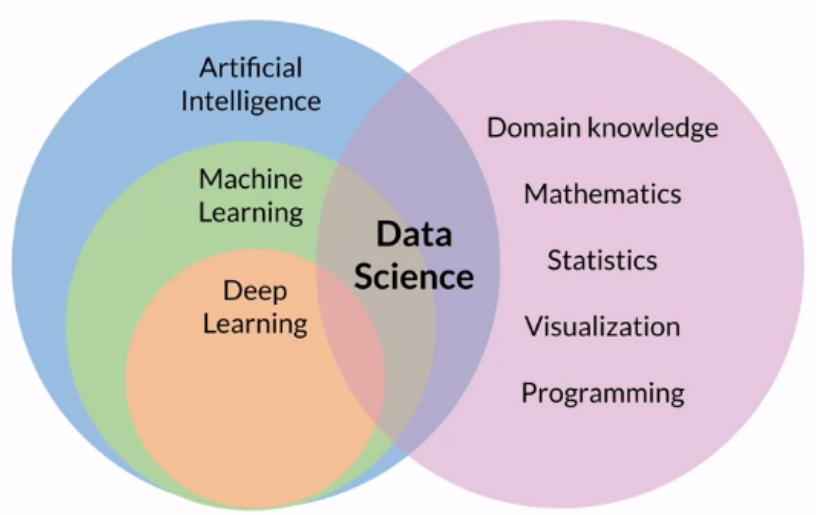
### Welcome

I will start this week with a brief introduction to the discipline of practical data science and discuss the benefits of performing data science in the Cloud. Practical data science helps you to improve your data science and machine learning skills, work with almost any amount of data and implement your use cases in the most efficient way for your use case. By moving your data science projects into the Cloud, you are no longer bound by resource limitations, such as your laptop, CPU, or memory. You can run data analysis on virtually any size of data. You can slice your data and run data transformations in parallel. You can switch from CPU to GPU if you want to speed up your model training and you can do much of this in just a few clicks. In the course of this week, I will give you a short overview of the data science and machine learning concepts, you will learn to apply and the toolkits you will use. I will then move on to define the task and the use case you will focus on. To give you our first hint, you will learn how to work with text data. Specifically, you will focus on a multi-class classification for sentiment analysis of product reviews. As always, data science starts with data. I will introduce you this week to the dataset you will work with to implement this text classification task. You will learn how easy it is to ingest the data into a central repository and explore the data using various tools from our practical data science and machine learning toolbox. You will learn how to analyze the data further using interactive queries and learn how to visualize the results. The analysis will reveal important information for future tasks in the model development process, as you will see.

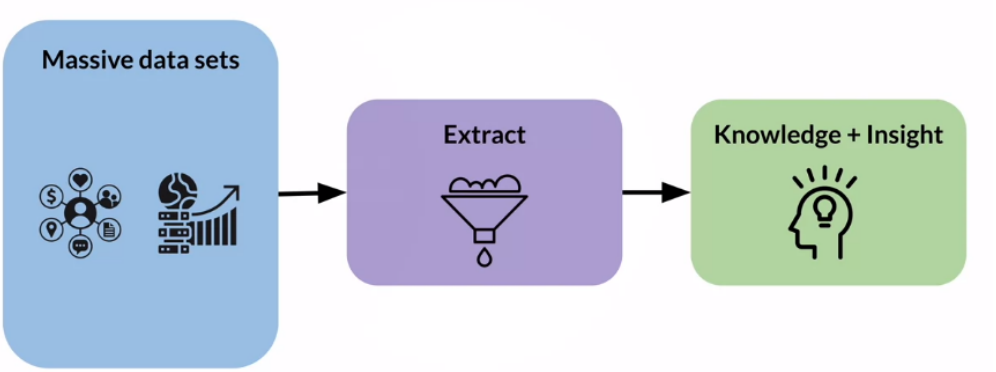
### Practical Data Science

I want to briefly define what's behind practical data science in the Cloud and introduce you to the data science and machine learning concepts which you will learn to apply and the toolkits you will learn to use.

Let me start by comparing the terms artificial intelligence, machine learning, deep learning, and data science.



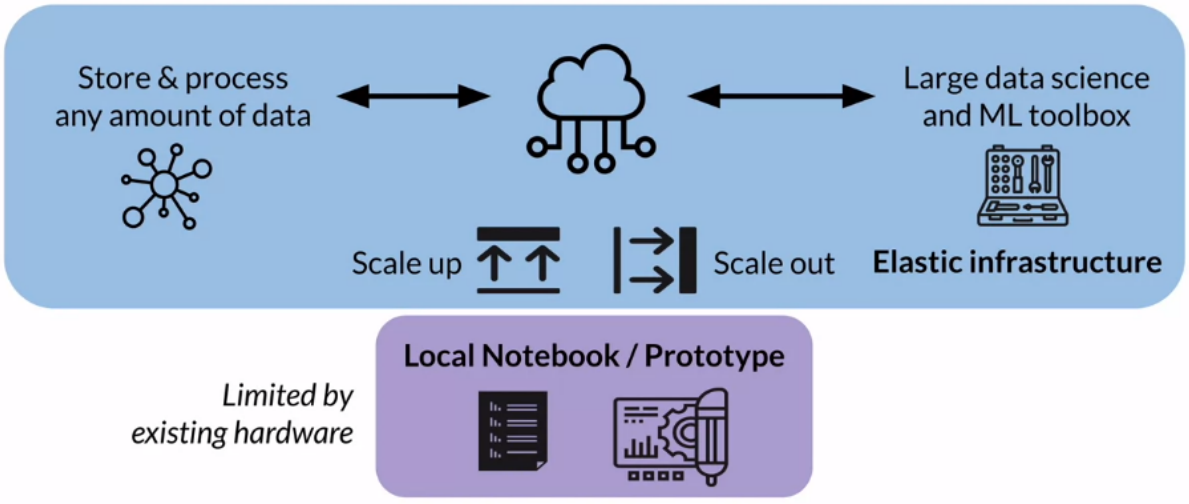

**Artificial intelligence** or AI is generally described as a technique that lets machines mimic human behavior. **Machine learning** or ML is a subset of AI that uses statistical methods and algorithms that are able to learn from data without being explicitly programmed. Then finally, **deep learning** is yet another subset of machine learning that uses artificial neural networks to learn from data. If you're new at data science, you see that this discipline touches all fields. **Data science** truly is an interdisciplinary field that combines business and domain knowledge with mathematics, statistics, data visualization, and programming skills.




Now, what's practical data science? **Practical data science** helps you to improve your data science and machine learning skills, work with almost any amount of data and implement their use cases in the most efficient way. It's different from working on a local development environment, such as your laptop, with small curated datasets. Practical data science is geared towards handling massive datasets that could originate from social media channels, mobile and web applications, public or company internal data sources, and much more, depending on the use case you're working on. This data is often messy, potentially error written, or even poorly documented. Practical data science tackles these issues by providing tools to analyze and clean

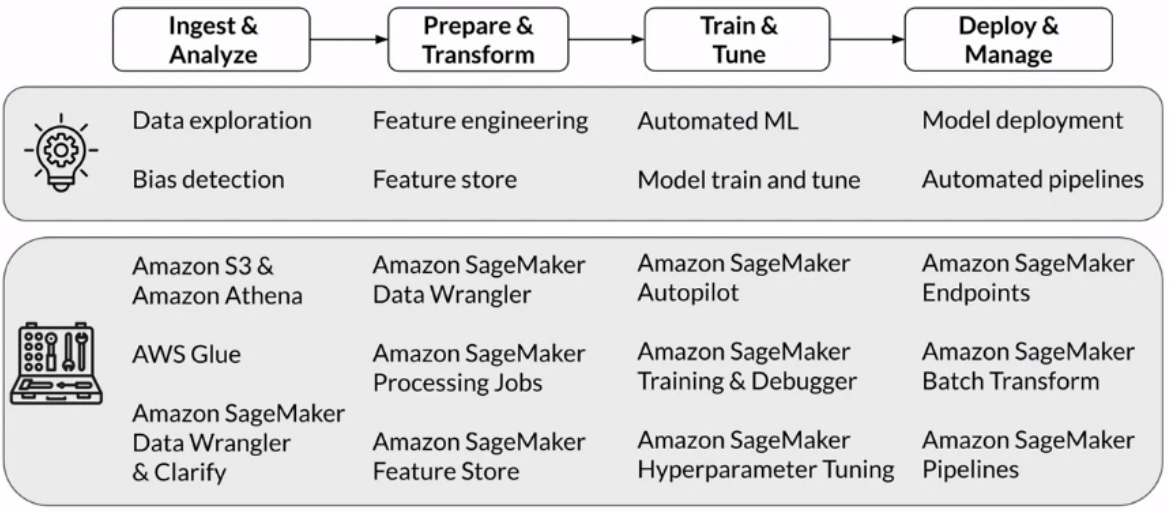
the data and to extract relevant features. This process leads to the ultimate goal of data science, which is knowledge distillation and gaining insight from those large datasets.

What's different about **practical data science in the Cloud**? If you develop data science projects in a local notebook or IDE environment for example, hosted on your laptop or company owned server pool you are limited by the existing hardware resources. For example, you have to carefully watch how much data you process and potentially lodge into memory. How much CPU processing power you have available to train and tune your model. If you need more, you need to buy additional computer resources. This process doesn't allow you to move and develop quickly. One of the biggest benefits of developing and running data science projects in the Cloud is the agility and elasticity that the Cloud offers. Maybe your model training takes too long because it consumes all of the CPU resources of the compute instance you have chosen. You can switch to using a compute instance that has more CPU resources, or even switch up to a GPU based compute instance. This is referred to as scaling up or instead of training your model on a single CPU instance, say you want to perform distributed model training in parallel across various compute instances. This is referred to as *scaling out*. Both scenarios can be accomplished in the Cloud within a few seconds. When the model training is completed, the instances are terminated as well. This means you only pay for what you actually use. This elastic infrastructure allows you to store and process almost any amount of data because the infrastructure scales too much the required resources. You can also innovate faster because you can try new datasets, new models, new code, or new machine learning libraries quickly and without any upfront investments. The Cloud also comes with a large data science and machine learning toolbox, you can choose from to perform your tasks as fast and efficiently as possible.

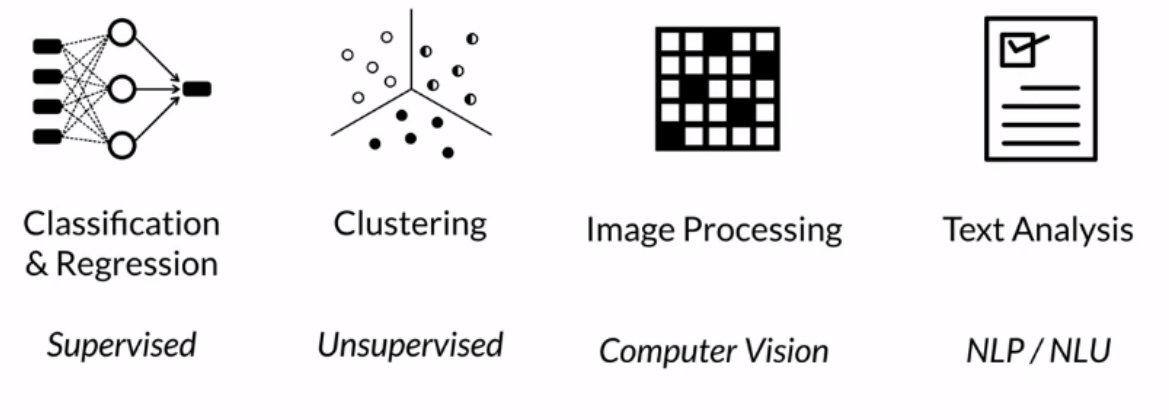



Let's have a look at a typical **machine learning workflow** and of course, every workflow starts with data. In the *ingest and analyze step*, you explore the data and analyze the data for potential statistical bias. Looking at the toolbox for this step, you will use Amazon Simple Storage Service or Amazon S3 and Amazon Athena to ingest, store and query your data. With AWS Glue, you will catalog the data in its schema. For statistical bias detection in data, you will learn how to work with Amazon SageMaker Data Wrangler and Amazon SageMaker Clarify, and don't worry right now, the tools will be explained in much more detail as you move throughout this course and the following courses of this specialization, as you learn how to apply each concept to implement the text classification use case. In the following weeks, you will step into the model development stage, where you'll start with *preparing and transforming* the data for model training. You will learn how to perform feature engineering and learn about the concept and

benefits of a feature story. Again, you will work with a powerful set of tools that are part of the Amazon SageMaker service. In the *model training and tuning stage*, you will learn how to use automated machine learning, cheek rate of first baseline, and a set of best model candidates for the use case. You will also perform custom model training and tuning in a later week. Again, you will work with additional tools part of the Amazon SageMaker service. In the *model deployment and management stage*, you will learn to discuss different model deployment options and strategies and how to orchestrate the model development as an automated pipeline. Similarly, your toolkit will be based on Amazon SageMaker.

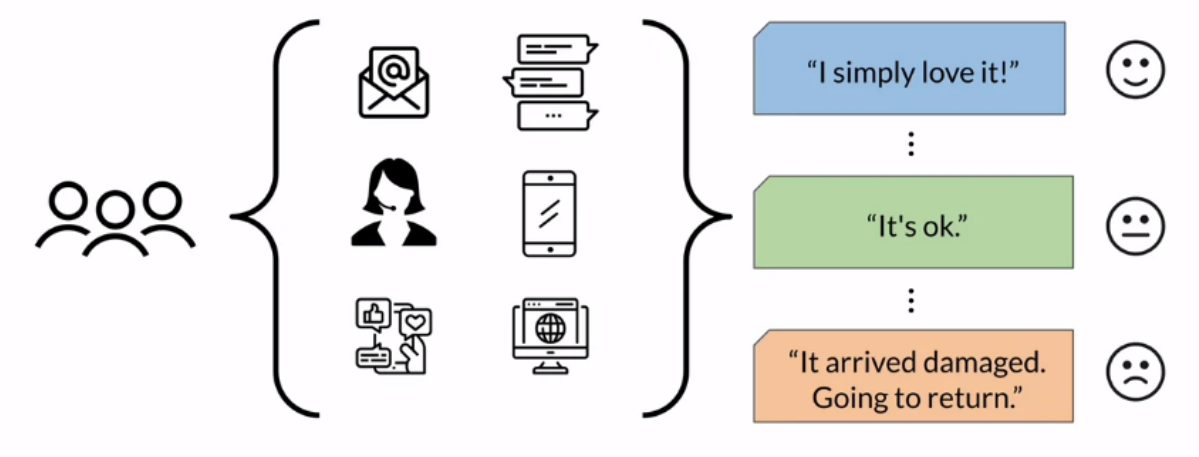



## Use case and data set




Many use cases can be classified into one of the major machine learning tasks and learning paradigms. Popular machine learning tasks are *classification and regression problems*, which are examples of **supervised learning**. In supervised learning, you learn by example by providing the algorithm with labeled data. With classification, the goal is to assign the input sample to a defined class. For example, is this email I received spam or not spam? In contrast, regression applies statistical methods to predict a continuous value, such as a house price given a set of related and non related input variables. Another popular task is *clustering* and clustering is an example of **unsupervised learning**. Here the data is not labelled, hence doesn't provide any examples. The clustering algorithm tries to find patterns in the data and starts grouping the data points into

distinct clusters. A clustering use case could be identifying different customer segments for marketing purposes. If you look more at the fields of AI applications, *image processing* is a major task as part of the broader scientific field of **computer vision**. You might need to classify images into pictures of dogs and cats, identify segments in the image to help your car's driver assistance systems to distinguish between speed signs and trees or to detect brand labels in an image. Following computer vision, *text analysis* has regained popularity and research and the industry in recent years. The field of **Natural Language Processing** or NLP or Natural Language Understanding NLU have been studied since the 1950s, but thanks to advances in deep learning and neural network architectures. You can see more advanced NLP tasks such as neural machine translations, sentiment analysis, question answering and others being implemented. And you will learn much more about the field of NLP and text analysis because this is the task you will focus on in this course.




More specifically, you will perform multi-class classification for sentiment analysis of product reviews. Assume you work at an e-commerce company selling many different products online. Your customers are leaving product feedback across all the online channels. Whether it is through sending email, writing chat FAQ messages on your website, maybe calling into your support center or posting messages on your company's mobile app, popular social networks or partner websites. And as a business, you want to be able to capture this customer feedback as quickly as possible to spot any change in market trains or customer behavior and then be alerted about potential product issues. Your task is to build an NLP model that will take those product reviews as input. You will then use the model to classify the sentiment of the reviews into the three classes of positive, neutral and negative. For example, a review such as "I simply Love It", should be classified into the positive class.

**Multi-class classification** is a supervised learning task hence you need to provide your tax classifier model with examples how to correctly learn to classify the products and the product reviews into the right sentiment classes. A great resource to explore

| Input feature for model training | Label for model training |
|---|---|
| Review Text | Sentiment |
| I simply love it! | 1 (positive) |
| It's ok. | 0 (neutral) |
| It arrived damaged, going to return | -1 (negative) |

product reviews are e-commerce sites. You can use the review text as the input feature for the model training and the sentiment as a label for model training. The sentiment class is usually expressed as an integer value for model training such as 1 for positive sentiment, 0 for neutral sentiment and -1 for negative sentiment.

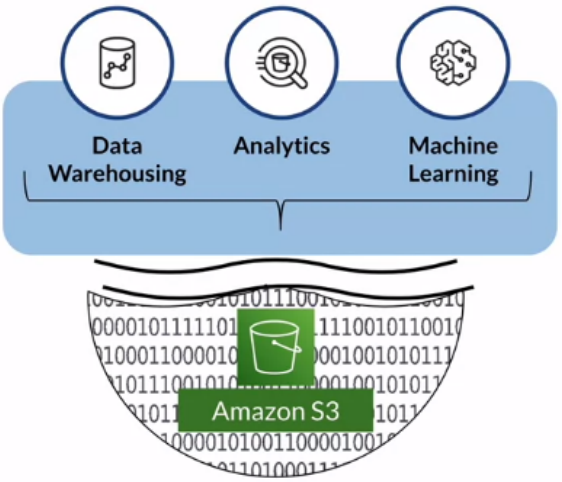
# Working with Data

## Data ingestion and exploration

One of the largest advantages of performing data science in the cloud is that you can store and process virtually any amount of data. The infrastructure scales elastically with the size of your data. Just think of the product reviews use case you will be working on. Imagine your e-commerce company is collecting all the customer feedback across all online channels. You need to capture sudden customer feedback streaming from social media channels, feedback captured and transcribed through supports under calls, incoming emails, mobile apps, and website data, and much more. To do that, you need a flexible and elastic repository that can start not only the different file formats, such as dealing with structured data, CSV files, as well as unstructured data such as support center call audio files. But it also needs to elastically scale the storage capacity as new data arrives.


- Centralized and secure repository
- Store, discover and share data at any scale
  - structured relational data
  - semi-structured data
  - unstructured data
  - streaming data
- Governance

Cloud based **data lakes** address this problem. Think of a data lake as this centralized and secure repository that can store, discover, and share virtually any amount and any type of data. You can ingest data in its raw format without any prior data transformation. Whether it's structured relational data in the form of CSV or TSV files, semi-structured data such as JSON or XML files, or unstructured data such as images, audio, and video files. You can also ingest streaming data, such as an application delivering a continuous feed of log files or feeds from social media channels into your data lake. A data lake needs to be governed. With new data arriving at any point in time you need to implement ways to discover and catalog the new data. You also need to secure and control access to the data to comply with the political data security, privacy, and governance regulations. With this governance in place, you can now give data signs and machine learning teams access to large and diverse datasets.




- Amazon Simple Storage Service (Amazon S3)
- Object storage
- Durable, available, exabyte scale
- Secure, compliant, auditable

Data lakes are often built on top of objects storage such as **Amazon S3**. You're probably familiar with file and block storage. File storage stores and manages data as individual files organized in hierarchical file folder structures. In contrast, block storage stores and manages data as individual chunks called the blocks. Each block receives a unique identifier, but no additional metadata is stored with that block. With object storage, data is stored and managed as objects, which consists of the data itself, any relevant metadata, such as when the object was last modified and a unique identifier. Object storage is particularly helpful for storing and retrieving growing amounts of data of any type hence it's the perfect foundation for data lakes. Amazon S3 gives you access to durable and high available object storage in the cloud. You can ingest virtually anything from just appear dataset files to exabytes of data. AWS also provides additional tools and services to assist you in building a secure, compliant, and auditable data lake on top of S3. With a data lake in place, you can now use this centralized data repository to enable data warehousing analytics and also machine learning.

## AWS Data Wrangler

- Open source Python library
- Connects pandas DataFrames and AWS data services
- Load/unload data from
  - data lakes
  - data warehouses
  - databases

```
!pip install awswrangler

import awswrangler as wr
import pandas as pd

# Retrieving the data directly from Amazon S3
df = wr.s3.read_csv(
     path='s3://bucket/prefix/')
```

Now, let me introduce some additional toolbox items you will be working on this week. One of them is **AWS Data Wrangler**. AWS Data Wrangler is an open source Python library developed by members of the AWS professional services team. The library connects Pandas DataFrame with AWS data-related Services. Pandas is a very popular Python data analysis and manipulation tool. AWS Data Wrangler offers abstracted functions to load or unload data from

data lakes, data warehouses or databases on AWS. You can install the library through the "*pip install awswrangler"* command. Here's a sample code snippet that shows how you can work with AWS Data Wrangler. First, you import the library together with Pandas. If you want to read, for example, CSV data from your S3 data lake into a Pandas DataFrame, you can run this command shown here using the S3 read CSV function, providing the S3 path to your data.

## AWS Glue Data Catalog

Another tool you will be using this week is the **AWS Glue Data Catalog**. This data catalog service is used to register or catalog the data stored in S3. Similar to taking inventory in a shop, you need to know what data is stored in your S3 data lake or bucket as an individual container for objects is called. Using the Data Catalog Service, you create a reference to the data basically S3 to table mapping. The AWS Glue table which is created inside an AWS Glue database only contains the metadata information such as the data schema. *It's important to note that no data is moved. All the data remains in your S3 location*. You catalog where to find the data and which schema should be used to query the data. Instead of manually registering the data, you can also use **AWS Glue Crawler**. A Crawler can be used and set up to run on a schedule or to automatically find new data which includes inferring the data schema and also to update the data catalog.

AWS Glue
Data Catalog

| | |
|---|---|
| **Name** | reviews |
| **Database** | dsoaws_deep_learning |
| **Classification** | csv |
| **Location** | s3://<bucket>/<prefix> |

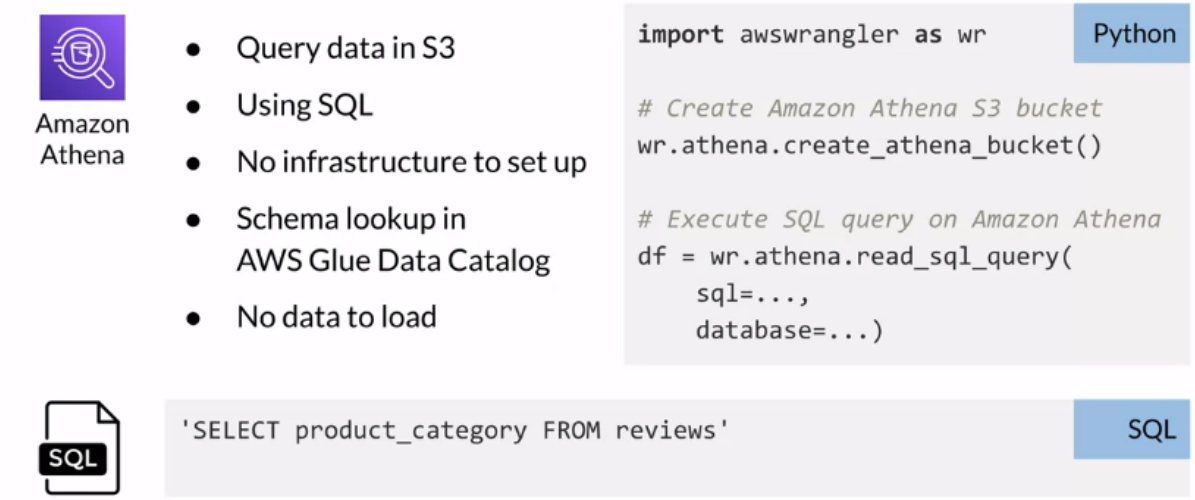
- Creates reference to data ("S3-to-table" mapping)
- Just metadata / schema stored in tables
- No data is moved
- *AWS Glue Crawlers* can be set up to automatically
  - infer data schema
  - update data catalog

Now how can you register the data? You can use the AWS Data Wrangler tool just as I introduced. The first step is to create an AWS Glue Data Catalog database. To do that, import the AWS Wrangler Python library as shown here, and then call the *catalog.create_database* function, providing a name for the database to create. AWS Data Wrangler also offers a convenience function called *catalog.create_CSV_table* that you can use to register the CSV data with the AWS Glue Data Catalog. The function will only store the schema and the metadata in the AWS Glue Data Catalog table that you specify. The actual data again remains in your S3 bucket.

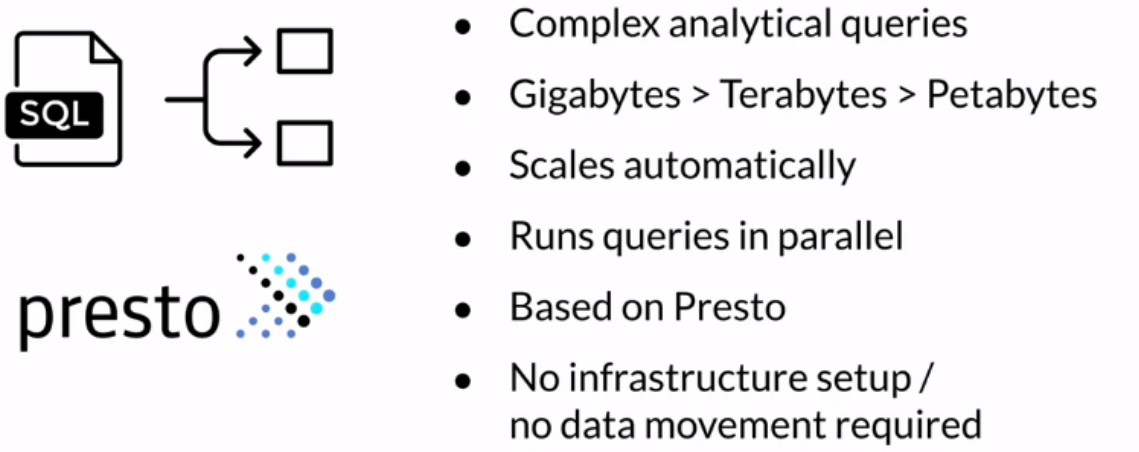
```
import awswrangler as wr

# Create a database in the
# AWS Glue Data Catalog
wr.catalog.create_database(
     name=...)

# Create CSV table (metadata only) in the
# AWS Glue Data Catalog
wr.catalog.create_csv_table(
     table=...,
     column_types=...,
     ...)
```




Now you can query the data stored in S3 using a tool called **Amazon Athena**. Athena is an interactive queries service that lets you run standard SQL queries to explore your data. Athena is serverless which means you don't need to set up any infrastructure to run those queries, and no matter how large the data is that you want to query, you can simply type your SQL query referencing the dataset schema you provided in the AWS Glue Data Catalog. No data is loaded or moved, and here is a sample SQL query. Let's list all product categories from the AWS Glue table called reviews, and imagine this table points to the dataset stored in S3. To run this query, you can use the previously introduced AWS Data Wrangler tool again. From the Python environment you're working in, just use the Data Wrangler, *athena.read_SQL_query* function. Pass in the SQL statement you have written and point to the AWS Glue database which contains the table you'll reference here in the SQL query. Again, this database and table only contains the metadata of your data. The data still resides in S3, and when you run this Python command, AWS Data Wrangler will send this SQL query to Amazon Athena. Athena then runs the query on the specified dataset and stores the results in S3, and it also returns the results in a Pandas DataFrame as specified in the command shown here.



Given this simplicity, you might wonder what is so special about this, and I have to admit the SQL query I've shown here was fairly simple. But just imagine building highly complex analytical queries to run against not just gigabytes, but potentially terabytes or petabytes of data. Using Athena, you don't have to worry about any compute and memory resources to support this query, because Athena will automatically scale out and split the query into simpler queries to run in parallel against your data. Athena is based on Presto, an open source distributed SQL engine developed for this exact use case, running interactive queries against data sources of all sizes. Remember, no installation or infrastructure setup is needed, and no data movement is required. Just register your data with AWS Glue and use Amazon Athena to explore your datasets from the comfort of your Python environment.

## Data visualization

Visualizing your data is one of the most effective ways of exploring your data across multiple dimensions at once. Depending on what kind of data you are exploring and what kind of relationships in the data you're looking for. The type of visualizations you use might be different. Let's take a look at a few of those visualizations you will use this week, and the tools you will work with.

**Pandas**, as introduced earlier, is used for data analysis and data manipulation. **NumPy is** used to perform scientific computing in python. Similarly, matplotlib and Seaborn, are popular python libraries for creating visualizations. **Matplotlib** helps to create static animated and interactive visualizations. **Seaborn** is based on matplotlib, and adds statistical data visualizations.


You will use these tools to visualize the product reviews.

The purpose of exploring and visualizing the data set is to better understand data set characteristics, from simple things such as understanding the number of samples you have available for model training to answering more complex business questions. Let me start with the first one. To get an understanding of how many samples the data set contains, let's run this SQL query using amazon Athena, which will return the number of reviews in each sentiment class. The query result is best suited to be visualized in the bar chart.

SQL Query

```
SELECT sentiment, COUNT(*) AS count_sentiment
FROM dsoaws_deep_learning.reviews
GROUP BY sentiment
ORDER BY sentiment DESC, count_sentiment
```

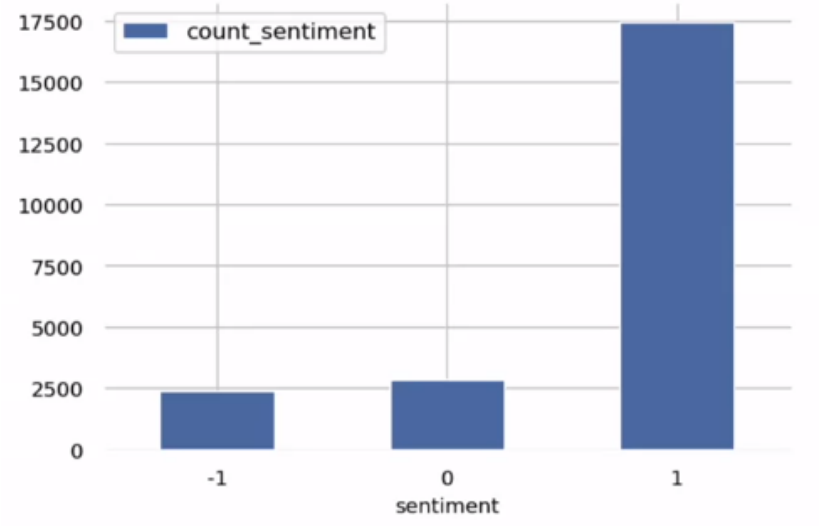
Here is the visualization code. I import the matplotlib library, for a simple bar chart, you can use the pandas data frame that holds the query results, and then call the plot bar function directly. You define the data frame columns which are used as the X and Y axis data, and at any additional data such as title to the plot. You can also use matplotlib to enrich the plot, let's say with labels for the X and Y axis. And when you're done, you can call the show function, and here can see the result in a sample bar chart.

Python visualization code

```
import matplotlib.pyplot as plt
chart = df.plot.bar(
     x="sentiment",
     y="count_sentiment")

plt.xlabel("sentiment")
plt.show(chart)
```

The bar chart also makes it easy to see that the positive sentiment class here, numbered with one, contains many more samples than any other of the classes. You basically have an unbalanced distribution of samples across the sentiment classes. You will learn how to address this in later week.




Here is another interesting visualization, this time I want to show you how to calculate percentiles and visualize data distributions. For this purpose, I will calculate the distribution of the review length or the number of words per review. The SQL query here is pretty straightforward. I split the review text in the column review body by a space character which gives me a list of the individual words, and then I calculate the cardinality which gives me the number of words.

SQL Query

```
SELECT CARDINALITY(SPLIT(review_body, ' ')) as num_words
FROM dsoaws_deep_learning.reviews
```

Before I plot the distribution, I want to calculate the percentiles. You can use the described function on the panda's data frame that contains the review length, and you specify the percentiles to calculate. To visualize the distribution of review length, I choose a histogram. Histograms represent frequency distributions. They show how often each different value occurs.

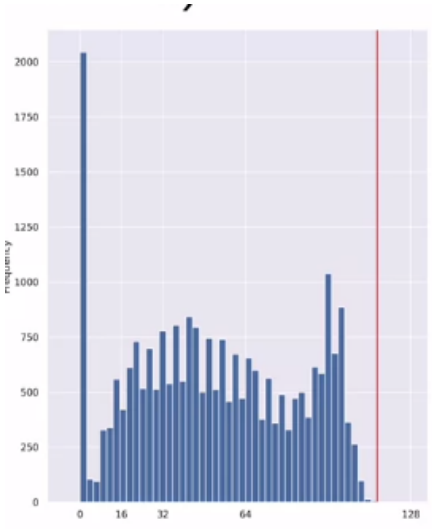
In this case, I want to group the review length in 100 bins, and I add a red marker to highlight the 100th percentile.

Python visualization code

```
summary = df["num_words"].describe(
    percentiles=[0.10, 0.20, 0.30, 0.40, 0.50, 0.60, 0.70, 0.80, 0.90, 1.00])

df["num_words"].plot.hist(
    xticks=[0, 16, 32, 64, 128, 256], bins=100,
    range=[0, 256]).axvline(x=summary["100%"], c="red")
```

Let's have a look at the result, and here we go. On the right, you can see sample results of the percentile calculations. You can see that the shortest review had only one word, and if you remember I grouped all review length into the 100 bins, which are represented here by the blue bars in the histogram. The fewer bins you specify, the fewer bars show up here. The X axis shows the length and the Y axis represents their frequency. And if you look at the 100 percentile highlighted and right here, you can see that all of the reviews have 115 words fewer in this sample. And this information will be very helpful later in the course when you start building the text classifier model.




| | |
|---|---|
| mean | 52.51 |
| std | 31.38 |
| min | 1.00 |
| 10% | 10.00 |
| 20% | 22.00 |
| 30% | 32.00 |
| 40% | 41.00 |
| 50% | 51.00 |
| 60% | 61.00 |
| 70% | 73.00 |
| 80% | 88.00 |
| 90% | 97.00 |
| **100%** | **115.00** |

## Additional reading material

If you wish to dive more deeply into the topics covered this week, feel free to check out these optional references. (You won't have to read these to complete this week's practice quizzes.)

- AWS Data Wrangler
- AWS Glue
- Amazon Athena
- Matplotlib
- Seaborn
- Pandas
- Numpy

# Week 2: Data Bias and Feature Importance

## Statistical bias and feature importance