Course duration

- 5 days

Course Benefits

Learn about the world of data engineering and how to build and maintain the data infrastructure that holds your enterprise's advanced analytics capacities together.

Course Outline

1. Introduction – The Big Data Landscape and Key Components
    1. The Big Data EcoSystem at CVS
    2. YARN, Spark, Spark Streaming, Kafka
    3. Containers: Docker/Kubernetes
    4. Monitoring and Logging: Prometheus
2. Data Engineering Defined
    1. Data is King
    2. Translating Data into Business Insights
    3. What is Data Engineering
    4. The Data-Related Roles
    5. The Data Science Skill Sets
    6. The Data Engineer Role
    7. An Example of a Data Product
    8. Data Schema for Data Exchange Interoperability
    9. The Data Exchange Interoperability Options
    10. Big Data and NoSQL
    11. Data Physics
    12. The Traditional Client – Server Processing Pattern
    13. Data Locality (Distributed Computing Economics)
    14. The CAP Theorem
    15. Mechanisms to Guarantee a Single CAP Property
    16. The CAP Triangle
    17. Eventual Consistency
3. Data Processing Phases
    1. Typical Data Processing Pipeline
    2. Data Discovery Phase
    3. Data Harvesting Phase
    4. Data Priming Phase
    5. Data Logistics and Data Governance
    6. Exploratory Data Analysis

7. Model Planning Phase
8. Model Building Phase
9. Communicating the Results
10. Production Roll-out

4. Core Data Engineering Tasks
    1. Data acquisition in Python
    2. Database and Web interfaces
    3. Ensuring data quality
    4. Repairing and normalizing data
    5. Descriptive statistics computing features in Python
    6. Processing data at scale
5. Functional Programming Primer
    1. What is Functional Programming
    2. Benefits of Functional Programming
    3. Functions as Data
    4. Using Map Function
    5. Using Filter Function
    6. Lambda expressions
    7. List.sort() Using Lambda Expression
    8. Difference Between Simple Loops and map/filter Type Functions
    9. Additional Functions
    10. Summary
6. Introduction to PySpark
    1. What is Apache Spark
    2. Spark use cases
    3. Architectural overview
7. PySpark Shell
    1. What is the PySpark Shell
    2. Starting and using the shell
    3. Spark context
    4. PySpark Shell vs Spark Shell
8. Resilient Distributed Dataset
    1. What are Resilient Distributed Dataset (RDD)
    2. Creating RDDs
    3. Transformations and operations
9. Parallel Processing
    1. Spark cluster
    2. Data partitioning
    3. Applications, jobs and tasks
10. Shared Variables
    1. What are shared variables
    2. Broadcast variables
    3. Accumulators
11. Spark SQL
    1. What is Spark SQL
    2. Uniform data
    3. Hive

    4. SQL Context object
12. The Spark Machine Learning Library
    1. What is MLlib?
    2. Supported Languages
    3. MLlib Packages
    4. Dense and Sparse Vectors
    5. Labeled Point
    6. Python Example of Using the LabeledPoint Class
    7. LIBSVM format
    8. An Example of a LIBSVM File
    9. Loading LIBSVM Files
    10. Local Matrices
    11. Example of Creating Matrices in MLlib
    12. Distributed Matrices
    13. Example of Using a Distributed Matrix
    14. Classification and Regression Algorithm
    15. Clustering
    16. Summary
13. Streaming – Kafka and Spark
    1. Installing Apache Kafka
    2. Configuration Files
    3. Starting Kafka
    4. Using Kafka Command Line Client Tools
    5. Setting up a Multi-Broker Cluster
    6. Using Multi-Broker Cluster
    7. Kafka Connect
    8. Kafka Connect ? Configuration Files
    9. Using Kafka Connect to Import/Export Data
    10. Building Data Pipelines
    11. Considerations When Building Data Pipelines
    12. Timeliness
    13. Reliability
    14. High and Varying Throughput
    15. High and Varying Throughput (Contd.)
    16. Data Formats
    17. Data Formats (Contd.)
    18. Transformations
    19. Transformations (Contd.)
    20. Security
    21. Failure Handling
    22. Coupling and Agility
    23. Ad-hoc Pipelines
    24. Loss of Metadata
    25. Extreme Processing
    26. Kafka Connect Versus Producer and Consumer
    27. Kafka Connect Versus Producer and Consumer (Contd.)
    28. Spark Streaming Features

29. How It Works
30. Basic Data Stream Sources
31. Advanced Data Stream Sources
32. The DStream Object

14. Infrastructure Optimization
    1. Monitoring Distributed Systems: Retrieving performance statistics from cluster members, aggregating output, consolidating application logs.
    2. Operations Strategy: What approaches can be used to find errors and bugs in distributed applications, and devise solutions for them?
    3. Case Study/Demonstration
    4. Lab: Explore log aggregation in Splunk
15. Making Big Data Secure
    1. What is required to secure Big Data infrastructure?
    2. How can centralized security management software, such as Kerberos and LDAP, be configured as part of a broader security architecture?
    3. What special considerations are there for applications and users who need to access protected resources?
    4. How are permissions and roles managed so that Big Data processing resources, such as Spark applications running on top of YARN or Kubernetes, are able to access data stored within HDFS or in an object storage like Amazon S3?
    5. Lab: Configuring Secure Access to Big Data Resources
16. How does DevOps work in a data context?
    1. Infrastructure: Version Control (git, GitHub), Automation (Jenkins), Processing (Spark, Hadoop, YARN), Data Management (Kafka),
    2. Process Differences: DataOps is more than DevOps and data
    3. Lifecycle and Differences
    4. Incorporating Complex Data Infrastructure into Continuous Integration/Deployment
    5. Standardization of runtime environment using containers
    6. Accounting for Infrastructure Differences within IaC configuration
    7. Incorporating orchestration to handle supporting component deployment and management
    8. Statistical Process Control (SPC) to ensure pipeline and model repeatability
    9. Case Study/Demonstration
17. Tooling
    1. GitHub: Source Forge
    2. Docker and Jenkins: Continuous Integration
    3. Spinnaker: Continuous Deployment
    4. Lab: Continuous Integration of a Kafka Based Application Using Jenkins

Class Materials

Each student will receive a comprehensive set of materials, including course notes and all the class examples.

Class Prerequisites

Experience in the following *is required* for this R Programming class:

- Some understanding of data science.