INVESTIGACIÓN e INNOVACIÓN en INGENIERÍAS
ISSN2344-8652

Architecture for Assessing Agronomic Data within a Big Data Environment

Arquitectura para el análisis de datos agronómicos en un ambiente de Big Data

Luis Felipe Vargas Rojas **Víctor Andrés Bucheli Guerrero**

Universidad del Valle, Colombia

OPEN ACCESS

Recibido: 21/05/2020
Aceptado: 23/09/2020
Publicado: 19/11/2020

Correspondencia de autores:
felipe.vargas@correounivalle.edu.co

Abstract

Objective: This study proposes a big data architecture for crop yield prediction processes. **Methodology:** The research methodology used in this study was two-fold: crop yield prediction processes and evaluation of several related software architectures. In this regard, the implementation requirements for storage and processing systems were established. **Results:** The resulting architecture consists of a MongoDB collection data model, a Kafka queuing system, and a PySpark processing system. This architecture inherits the capacity for vertical and horizontal scaling from these technologies to serve heterogeneous data and domain-specific variables and facilitate interaction with different transformations and machine learning models. **Conclusions:** Big data technologies can be used to accurately model crop yield prediction processes. This scheme serves as a reference for conducting agronomic data analysis in a scalable and flexible big data environment.

Keywords: Data Analysis, Agronomic Data, Big Data, Apache Spark, MongoDB.

Resumen

Objetivo: Proponer una arquitectura de Big Data especializada en el proceso de predicción del rendimiento de un cultivo. Metodología: Se investigó tomando en cuenta dos enfoques, por un lado, el proceso de predicción de rendimientos de cultivos, por otro lado, se estudiaron arquitecturas de software relacionadas. A partir de las investigaciones se definieron los requerimientos para un sistema de almacenamiento y procesamiento en este ámbito. Resultados: la arquitectura incluyó (1) un modelo de datos en colecciones MongoDB; (2) un sistema de encolamiento Kafka; y (3) un sistema de procesamiento en PySpark. La arquitectura hereda de las tecnologías usadas la capacidad de escalamiento vertical y horizontal, de atender datos heterogéneos y variables de dominio específico, además de permitir la interacción con diferentes transformaciones y modelos de aprendizaje automático. Conclusión: Las tecnologías de Big Data pueden modelar el proceso de predicción de rendimientos de cultivo, este esquema sirve como referencia para llevar a cabo análisis de datos agronómicos sobre un ambiente de Big Data escalable y flexible.

Palabras clave: Análisis de datos, Datos Agronómicos, Big Data, Apache Spark, MongoDB.

Como citar (IEEE): L. Vargas Rojas., V. Bucheli Guerrero. " "Arquitectura para el análisis de datos agronómicos en un ambiente de Big Data". *Investigación e Innovación en Ingenierías*, vol. 8, n°2, 233-250, 2020. https://doi.org/10.17081/invinno.8.2.4211

Introduction

The Food and Agriculture Organization of the United States predicted in 2019 that agricultural production would need to rise by 70% to sustain a population that is expected to exceed 9 billion by 2050 [1]. This warning, coupled with climate change issues, compromises food security. More exhaustive data analysis systems must support agricultural processes to mitigate the impact caused. Crop yield prediction (CYP) is one issue in this field that is of interest. Crop yield is the quantity of crop harvested, expressed in kilograms per hectare. Additionally, CYP is affected by gene-environment interaction. CYP is difficult to model due to its complexity and nonlinear behavior. They also depend on many dimensions, many of which are often unknown, and on quantitative or qualitative types [2]. Some statistical and machine learning models, such as neural network models [2], decision trees [3], and linear regression [4], have been developed to approach this problem statistically as a regression problem.

The integration of experiments under different formats, as well as its capabilities for collecting more detailed information on crop growing seasons, have transformed this domain into an adequate space for big data applications [5]. For example, volume from sensor data; the variety generated by sensor data, spreadsheets, information systems, and traditional database systems; and velocity so that sensor networks can process timely data and prevent plague or losses due to drastic climate conditions.

This paper is organized as follows. The state-of-the-art section discusses related literature works, discerning between studies geared toward assessing prediction models and software development or architectural proposals. The methodology section defines the requirements, components, and technologies chosen for the architecture proposed. This architecture is detailed in the results section. Finally, the conclusion section presents further assessments and discussions.

State of the Art

This section discusses two forecasting system approaches: studies focusing on comparing forecasting errors and studies where software components or platforms are implemented.

Crop Yield Regression Models

These models have a significant statistical component, and datasets are often integrated into a CSV file that serves as the model's database in terms of predictive quality. Given the large number of variables that come into play in this prediction, it may be considered a multidimensional regression problem [4]. The corresponding function may be defined as the following:

$$F(G,E,M), \quad (1)$$

where G is the variable referring to the crop's genetic characteristics; E is the variable related to the environment in which planting occurs (climate, soil); M is the management practices established before and during crop growth. This function returns a continuous value known as crop yield or agricultural output.

Table 1. Learning Algorithms used for Crop Yield Forecasting . +

Model	Algorithm	Reference
Linear Regression	Multiple	[3] [6]
	Robust	[7]
Kernel	SVR	[3] [4]
Regression Trees	CART	[4]
	M5-Prime	[3]
Neural Networks	MLP	[7] [2] [8] [4] [9]
	RBF	[4]
Ensemble	RF	[10] [1]
	CIF	[10] [11]

Source: Prepared by the Authors

Table 1 summarizes the algorithms used for crop yield forecasting. Each algorithm is based on a representation model that directly affects the type of problem it can deal with and offers advantages and disadvantages. This table differentiates between statistical and machine learning models. Qualities such as the ability to support atypical data or ease of interpretation are some of the comparison criteria used for these studies. A detailed comparative table of the algorithms can be found in [6].

Linear Regression

Multiple linear regression (MLR) has been widely used for CYP, even though it is not a true machine learning algorithm [3]. It is a statistical technique applied to predict a dependent variable Yi, using a set of independent explanatory variables Xij. In [7], MLR is defined as the following:

$$Y_i = \sum_{j=1}^{K} B_{i,j} + \varepsilon_i \qquad (2)$$

where k is the number of explanatory variables, Bj is the regression coefficient j, and Xij is the value j from observation i.

In [3], this model was applied to a dataset of different crops, with the largest set containing 2617 samples. The average yield obtained using this algorithm was the root mean square error (RMSE) of 5.41% and root relative squared error (RRSE) of 86.18%. They also used WEKA as an analysis tool, and no information on training time was found.

[8] used another technique known as robust linear regression. This technique is an MLR variation that exploits information as much as possible without removing outliers. When datasets have high leverage (how far is a given variable from its mean) and outliers are identified, this method is appropriate. This technique is important when the assumptions of the least squares regression are violated. The R2 metric for this model was 0.65, which explains more than 60% of the variability. This analysis used 254 records; no information was given on the tools or training times.

Neural Networks

This is the commonly preferred method for CYP, especially for multilayer perceptron networks with backpropagation. Unlike linear regression, this method can generate nonlinear mappings between inputs and outputs. Cross-validation has been used to address overtraining issues, which are associated with

achieving perfect performance using the training set but demonstrating low performance when using test data [9].

Mean squared error (MSE) was used to measure the network performance in [10]. They compared different network topologies, changing the number of neurons per layer from 2 to 32. Note that there is a lack of clear evidence to validate whether larger networks obtain better predictions. They used 5241 records, and no information was given on the assessment tools used or training times.

Regression Trees

Decision-tree learning is an inductive learning paradigm in which a model is built from data or observations according to some criteria. The model learns a rule of thumb from the observed instances [4].

MLR generates global models, and only one formula fits the feature space. However, regression trees use a different approach, recursively partitioning the feature space until reaching regions small enough to be represented by a simple model [11]. The first tree node (the root node) has no incoming edges, while the other nodes exhibit exactly one incoming edge [3]. A node with outgoing edges is called a test node, while those without outgoing edges are called a leaf node. Each node splits the dataset into two or more subspaces, applying impurity measures (standard deviation or the Gini index) as a division criterion. The leaf node assigns a numeric value to the last instance partition. However, the test node selection is based on the answer to which attribute is examined at the top of the tree. To answer this question, each attribute is assessed to determine its data parsing capabilities [4]. The process of predicting a new value is based on navigating this tree from the root node to a leaf node. The algorithms used differ in three aspects: the measure of impurities used in continuous attributes, the reduction rule, and the mechanism used to determine the value of the leaf node. For example, M5 uses standard deviation reduction as an impurity criterion, while CART uses variance.

A maximum of 2617 samples were evaluated over several test sets in [3], the M5-Prime variant of this method. On average, the following results were obtained: 5.14% RMSE and 79.46% RRSE. The analysis tool used was WEKA. No information was provided on training times.

In [4], the Matlab implementation of the classregtree was used. The data came from several training sets, each representing a specific-planting area, with the largest containing 5241 records. The best-case scenario yielded a medium absolute error (MAE) of 0.4 and an RMSE of 0.9 from this training set. No information was provided on training times.

Support Vector Regression (SVR)

A support vector machine is a supervised learning algorithm discovered by [12]. For the regression task, there is a variant known as SVR [13]. Given a training set, the goal of SVR is to approximate a linear function. This function seeks to minimize empirical risk, defined as the following:

$$R_{emp} = \frac{1}{2}\sum_{i=1}^{N} L_{\varepsilon}(y - f(x)) \qquad (3)$$

where $L_{\varepsilon}(y - f(x)) = \max((|\xi| - \varepsilon), 0)$. $|\xi|$ is the slack variable, introduced primarily to deal with otherwise impossible optimization problems. When this variable is used, errors are ignored as long as they are as small as a properly selected error. This function is known as the ε-insensitive loss function.

[3] used SVR to obtain RMSE and RRSE averages of 5.02% and 81.97%, respectively. No information was provided on training times.

In [4], SVR showed proper performance with an MAE and RMSE of 0.32 and 0.47, respectively.

Random Forest

A random forest (RF) [14] is an ensemble learning method for classification or regression. RF attaches an ensemble of decision tree models to a dataset. The value predicted for a regression problem is based on the calculation of the mean of the results generated by each tree. RF can also offer a ranking of the relative importance of each predictor variable. The importance of the variables is based on the out-of-bag (OOB) regression error prediction.

In [1], RF was used to measure sugarcane yields, specifically the R package called randomForest. They used 500 trees derived from 500 bootstrapped data. Additionally, separation points were calculated from a random subset of all available predictor variables. Almost 30% data were OOB and were not used to construct the decision tree. They calculated the prediction error using MSE. Their model explained about 70% of the variability exhibited by the data. However, their study did not provide an exact value for the quantity of data; instead, it says that the model was built using data from 1992 to 2013.

Architecture and Systems

This type of publication focuses on modeling and data structure. Some studies have described how wireless sensor networks and remote sensing are used. Additionally, the proposed architectures usually rely on relational database systems.

In [15], the authors focus on the deployment of a sensor network architecture for a monitoring system used in a crop field. An affordable and reliable real-time agronomic data collection system with a simple infrastructure was developed for these purposes. Their study discussed the hardware requirements and network configuration required for data transmission, as well as the minimum battery consumption required to guarantee long-term autonomy. Their experiment was conducted at an organic horticulture company in Murcia, Spain. The variables collected were soil temperature, moisture content, conductivity, and salinity, while the relative humidity and air temperature were measured. Finally, a well was created at a close distance from the field to measure the electrical conductivity of the water and its temperature. The generated datasets were stored in a MySQL database. Additionally, a monitoring application was created consisting of the following: a GUI that shows the nodes on a map and the data collected by each sensor; a data collector module with a dedicated connection that sends an event updating the data in a serial port; a database with detailed node information, their possible aggregations, the sensors integrated into each node, historical records, sensor types, and alarm history records (for example, battery failure). Network and energy consumption details are provided in the results. In their study, the highest peaks were generated during data acquisition, but they were usually short. Before, all measurements were performed manually, where each month, a person with portable devices would average agronomic variables at a low frequency. Here, descriptive event information is usually lost. After system implementation, crop status was monitored

in real-time, which allowed the researchers to verify whether optimal agronomic conditions for plant growth were being met and to take timely actions against adverse situations.

In [16], the creation of a data warehouse was proposed. This data warehouse considers the specific characteristics of agricultural data, its heterogeneity, and different levels of detail. The proposed design was based on the star schema of dimensions and facts proposed by Kimball. The data sources were endogenous from internal company operations and exogenous from government data and weather stations. A total of 29 datasets, each with 18 tables, averaging 1.4 GB in size, were used. The datasets were collected from the European Union countries and correspond to hundreds of thousands of field trials. The design includes facts regarding performance, treatments, operations, trade, and associations to multiple dimensions, such as soil, weather stations, and crops. Multiple facts can also share common dimensions that relate them to one another. The study excluded results or implementations.

In [17], a CYP framework was proposed, defining a step-by-step prediction. In step 1, a crop was selected; in step 2, independent variables were selected; in step 3, dependent variables were selected; in step 4, the dataset was selected; in step 5, the output variables were preprocessed to classify the performance as high, regular, and low. This study summarizes remote sensing data into indices such as NDVI, VCI, and TCI and combines them with climate and soil data. The algorithms used were OBIA for remote sensing, J48, and decision trees to apply mining approaches to agricultural conditions.

The variables collected were temperature, solar radiation, humidity, rainfall, soil type, seed variety, type of fertilizer, quantity, weed management, pest management, length of the land, planting pattern, and space between plants. The study does not discuss the results based on the model execution.

In [18], the study proposed a system for measuring the impact of intensive agriculture on the environment. For these purposes, the authors assessed geospatial data using big data analytics. They conducted this study in Catalonia, Spain, as this region exhibits high contamination rates due to excessive cattle farming. They also described the AgriBigCAT online platform, which uses information from multiple sources, and combines geospatial analysis with web technologies. Here, Apache Hive was used for data storage, and ArcGIS + Hadoop was used for data analysis. Finally, the view was developed on the platform implemented. The results from the application and assessments were studied from the maps. Their interactivity consists of a user selecting a region, an animal, and a contaminant. This demonstrates that higher concentrations of nitrogenous pollutants may be found in regions with higher cattle farming rates and a deficit of farming land.

In [19], the authors used AgDataBox-API to address data organization and management issues. This HTTP-accessible API seeks to centralize data and minimize the complexities arising from managing different formats, from georeferenced data to remote image sensors. They also discussed a modular architecture that facilitates the integration of different external applications. This was conducted using free access technologies and Postgres as a database.

Methodology

Requirements

After assessing the existing CYP literature containing environmental and phenotypic data and reviewing the different processes, datasets, and architectures proposed in these studies, the challenges derived from the construction of a storage system become evident. For example, some variables were difficult to categorize;

different predictors were observed in each experiment; no open-source big data implementations were observed; a clear reference framework was not used for macroprocesses. Additionally, experiments were isolated, and the data and results were not easily replicable.

For a data warehouse, the following must be considered [16]:

- The data warehouse must be flexible enough to store heterogeneous data.
- It must support connections to different data sources.
- It must support horizontal and vertical scalability.
- It must support interfaces with data processing tools.

Big data is defined by the five "V" [20]: velocity, volume, variety, value, and veracity. A very active community has recently been working on storage system proposals. This domain focuses on the advances that these systems showed in the Internet of Things, where proposals for decoupled systems accessed by processing systems to run prediction models or even generate dashboards, views, and descriptive analyses may be observed [21].

A processing system must consider the following:

- The processing system must be able to run regression machine learning models.
- It must support the creation of data preprocessing and harmonizing pipelines.
- It must allow users to interact with different scenarios to understand and explore the data.
- It must guarantee adequate scalability for processing and modeling tasks.

Table 2. Types of Data involved in Crop Growing Seasons.

Category	Frequency	Type	Description
Observations	Hour/day/minutes	Numeric	Sensor data and weather stations
Objects	Hour/day	Binary	Remote sensor imaging
Context	Full season	Texts, categorical	Climate type, city, defined area, farm location, soil type, altitude
Handling	Full season/ month/day	Texts, categorical	Fertilizers, distances between crops, irrigation frequency

Table 2 semantically presents the datasets that affect crop growing seasons, highlighting the frequency, dataset types, and the entity they represent. Observational data feature a higher level of detail, while descriptive data exhibit a slower generation rate and represent a single record for the entire growing season.

Table 3. Crop Test Base Entity.

	Location	Handling	Crop	Planting Date	Harvest Date
ID	Where?	How?	What?	When?	When?
	How much yield?				

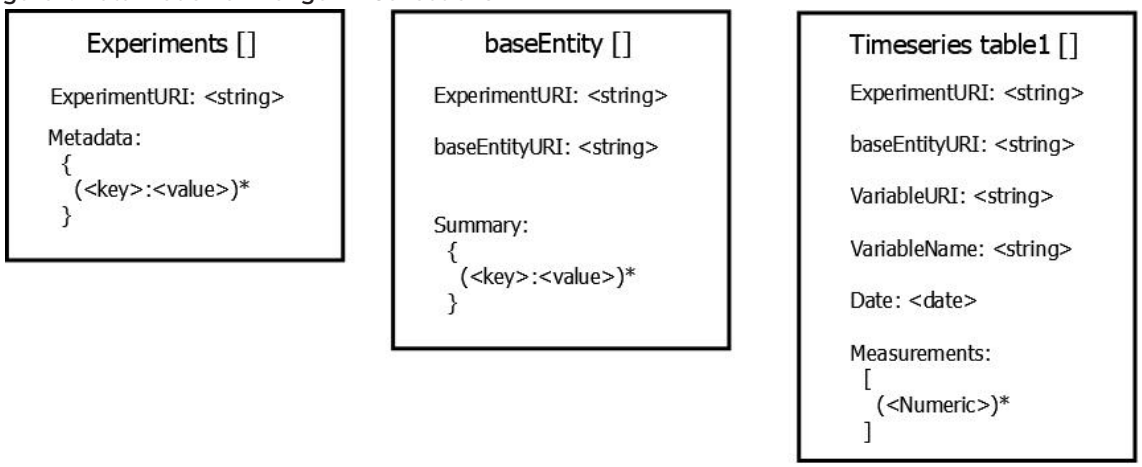
Source: Prepared by the Authors

All these data were derived from a base entity (see Table 3), where the base entity describes the event. A crop event is understood as everything that occurs between planting and harvesting and may affect the yield rate obtained [2].

Data Model

The data model considers the information in Table 2 and is based on the terms used in the Agricultural Model Intercomparison and Improvement Project (AgMIP) agriculture experiment standards [22].

Figure 1. Data Model for MongoDB Collections



Source: Prepared by the Authors

Figure 1 shows the data model for entities with low, medium, and high frequencies. Here, the frequency of experiments is regarded as low because the maximum number of experiments in a semester does not exceed 10. An experiment has metadata associated with the context in which it is evaluated, the companies involved, the objectives, and start and end dates, among others [23]. The next level, which is the base entity collection, can refer to a plant or planted area. This second level of identification groups the data. These data will become the records used for training the models in terms of learning models. This collection can be considered an agricultural event, as presented in Table 3. The third level refers to time-series variables. These variables are measured daily or sometimes after hours. For this study, they were measured daily. Defining a table for each time series was also recommended. For example, one for weather, image analysis data, and handling events. The scheme's flexibility lies in the ability to aggregate heterogeneous objects, while three collections are proposed as the foundation for harmonizing data.

Microservices and ETL

Within the context of this architecture, extraction, transformation, and load (ETL) are used to establish a connection with an external entity through sockets, HTTP, or a given protocol. ETLs are independent of the programming language used and use their database, which is why they are considered microservices. ETLs receive datasets from external entities; they organize these datasets and apply the necessary formats to finally deposit them in the queuing system.

The queuing system uses the publish-subscribe model. Every ETL microservice connected must have a client known as "a producer" to connect to the queuing system and can only perform the "publish" call.

The architecture suggests the microservices model because the developer can first build the process in any programming language. Then, the process is stored in a decoupled and independent system, which facilitates debugging and maintenance [24]. Additionally, different development teams can work on different ETLs.

Queuing System

A queuing system that can mediate data transfers was selected to prevent data losses and manage the frequency of inserts within the storage system. The complexity of the queuing system can be as simple as a memory buffer or as complex as databases or file systems used for temporary storage. This system protects the storage system as it is the only authorized means of publishing data to the store. Several controllers have been created in the main programming languages for client implementations (producers or consumers). Hence, the ETL microservice only needs to incorporate the said controller to send data, thus making the system interoperable and accessible for communication.

Cluster Storage

The warehouse system is a component that receives all data ready for integration. Its complexity depends on its implementation. This study proposed using a NoSQL database without previous schema requirements to store the different datasets for this item. This microservice creates a consumer client for the queuing system, which is the only component authorized to extract data from the queue.

MongoDB Database

MongoDB is better aligned to the agricultural context than other big data technologies. Table 4 presents the corresponding comparison criteria [25] [26].

Table 4. Criteria for comparing MongoDB and Other Big Data Technologies.

Criterion	MongoDB	Cassandra	HDFS
CAP theorem	CP	AP	AP
Model	Master	Peer-to-peer	Master
	slave		slave
Aggregation framework	(+)	(-)	(-)
Map-reduce functions	(+)	(-)	(+)
Specific data queries	(+) Supports two indexing levels	(+-) Only row indexes	(-) Data reading required
Query language	(-) Complex language	(-) Similar to SQL but without JOIN	(-) File searches
Flexibility and complex schemes	(+) JSON	(-)	(-)
IoT support	(+)	(+)	(-)
Scaling	(+)	(+)	(++) up to petabytes

Source: Prepared by the Authors

(+) Complies, (–) Does not comply, (+–) Partially Complies

Given that this study focuses on agricultural data, some criteria from Table 4 become quite relevant: schema flexibility and complex data, heterogeneous data, and the schema used in AgMIP. The MongoDB collection design can handle these complexities well by responding to the R1 requirement. However, [22] is JSON-based. Specific data queries: complete data readings are not always necessary because agricultural data must be read interactively and using certain units of time. Instead, an analysis focused on a given moment or dimension can complement processing requirement R7. Aggregation frameworks: since agricultural data come in very different formats, a database-preprocessing framework facilitates the steps leading to data analysis. However, real-time and availability are not priority characteristics for agricultural assessments. Taking a few minutes to respond does not affect decision-making much unless it is a very detailed analysis.

Apache Spark

Apache Spark [27] is an open-source framework for large-scale distributed processing. It offers high-level application programming interfaces (APIs) for structured query language (SQL), machine learning, streaming, and graph processing. Additionally, it is available in different languages, such as Python, Java, R, and Scala. Furthermore, its storage system is agnostic and provides connectors and access methods for different data sources. It can also combine these data sources. Spark applications have been used in various domains, from finance to scientific data processing, in which top-level libraries are typically combined. Since its launch in 2010, Spark has become the most active open-source project for big data processing, and it unifies a set of high-level libraries. This feature distinguishes it from special-purpose cluster processing systems such as Storm, GraphLab, and Impala. Still, Spark competes with them on performance. For example, the largest use case published includes a cluster of 8000 nodes from the China Tencent social network that processes 1PB of data per day [28].

Spark is considered an evolved version of the Hadoop processing system because it exceeds its performance (10x–100x). Spark also extends its map-reduce features and adds new functionalities. Moreover, unlike map-reduce, which always accesses the disk and consequently significantly affects performance, Spark caches RDDs in pipelines prevent multiple disk reads before executing iterative functions that retrieve the same records. Spark retains the same expressiveness, scalability, and fault tolerance of a map-reduce implementation but with a framework capable of serving a broader range of applications [27].

Spark programs are written in terms of two types of operations: transformations and actions. These programs are executed by distributed datasets optimized for in-memory processing. Spark's root structure is based on the architecture of resilient distributed datasets (RDD). This structure is an abstraction used to distribute large datasets between cluster nodes and perform operations on them, such as filter, map, reduce, and groupBy [27].

The Apache Spark implementation was conducted in Scala, a statically typed high-level programming language for the Java VM. Spark exposes a functional programming interface similar to DryadLINQ. RDDs are Scala objects, immutable, ephemeral (by default), and read-only [27]. RDDs support fault tolerance during lineage, and if an RDD partition is lost, the RDD has enough information about how it was derived from other RDDs to specifically rebuild that partition [28].

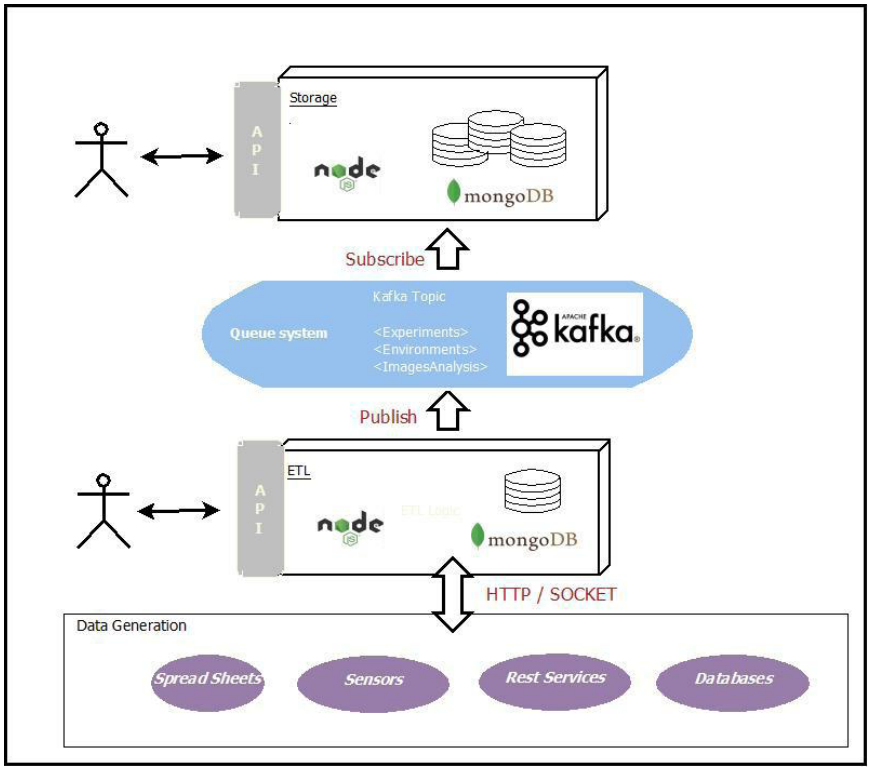
Results

Storage Architecture

This architecture addresses R1–R4 requirements. Figure 2 illustrates the architecture of the storage system. In a bottom-up view, the following components can be observed:

- External data sources are consumed or accessed from the ETL.
- An ETL is implemented under the microservices concept. Since it is decoupled, it can only issue messages to the queuing system using the published method. The architecture proposes using this ETL as a manual development component to connect to different data sources, thus fulfilling the R2 requirement.
- The queuing system is a mediator that prevents the storage cluster from receiving many connections. This system protects the data warehouse and proposes a clear and simple message exchange model for connecting new ETLs. The Kafka topic label is also observed, and each topic is a bridge that defines to which collection the datasets belong.
- These datasets are stored in the storage cluster. This is a MongoDB configuration that will depend on the resources available. MongoDB guaranties the scalability requested by the R3 requirement.
- Finally, some access APIs are defined at the ETL level to define which datasets are transferred to the warehouse. Another API is defined at the warehouse level to view the data stored there.

Figure 2. Storage System: (1) Data Generation conducted by External Agents; (2) Data Collection and Validation; (3) Queuing System; (4) Scalable Cluster for Storing Large Volumes of Heterogeneous Data; (5) access point via web (APIs)



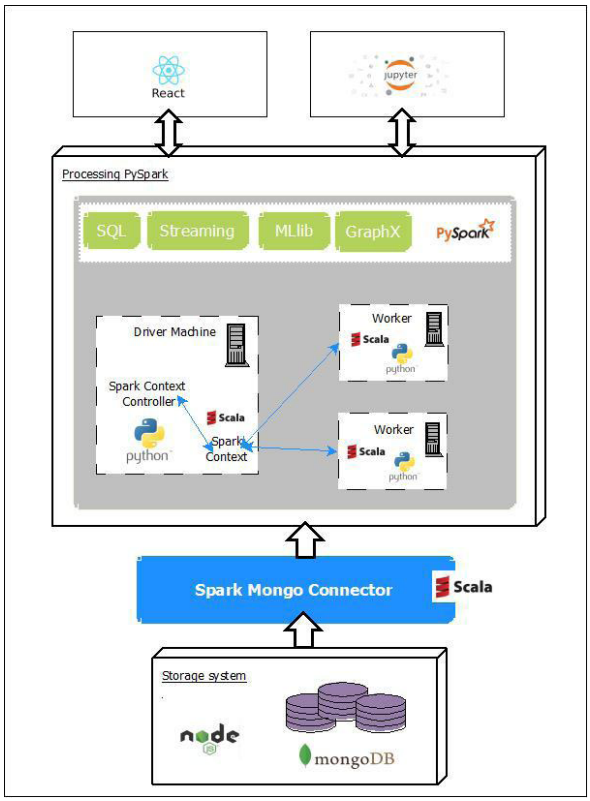
Source: Prepared by the Authors

Processing Architecture

This architecture supplements the requirements mentioned above, R5–R8. Figure 3 shows an overview of the architecture, which is grounded in PySpark and MongoDB technologies.

- The storage system contains the foundation data connected to the processing system.
- This connection is achieved through an open-source driver that establishes a bridge between MongoDB and Spark. This is completely aligned with requirement R4. This connector converts a MongoDB collection into a Spark dataframe. An aggregation pipeline that runs natively in MongoDB can be defined among the data extraction configurations.
- PySpark comprises a driver node and several workers. In the driver, a flow is defined to represent the application. Additionally, functions are called in parallel for on-demand execution by the workers. Spark's functions will be executed in the Java VM, so they require higher performance rates. This guaranties the processing scalability mentioned in requirement R8.
- The architecture includes the highest-level Spark libraries, such as SQL, Streaming, MLib, and graphX. In a Spark session, calls from these libraries can be combined to create interactive applications. This corresponds to requirements R5–R6.
- Finally, the access layer implies that notebooks can be created or accessed via a web application and invoked interactively.

Figure 3. Processing System: (1) MongoDB Database; (2) Communications Connector; Spark Logic Components, Driver Node, and Workers; (4) Spark High-Level Libraries: Mlib for Machine Learning, SQL for Data Retrieval and Aggregation, Streaming for Stream Processing, and Graphx for Graph Analysis; (5) Interaction with Users through a Web Application and a Notebook



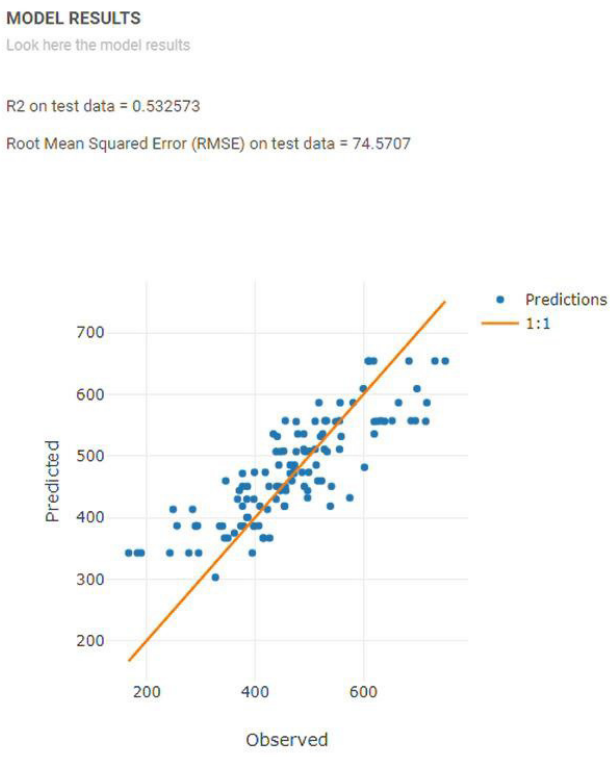
Source: Prepared by the Authors

These two schemes complement each other, thus covering the needs expressed in requirements R1–R8. This architecture has the potential to create a robust system and serve as an alternative to assessing agronomic data and inheriting all benefits of big data technologies.

Proof of Concept with PHIS Data

A Proof of Concept (POC) was conducted to validate these components. In this POC, the architecture designed from software components became real. Testing data were obtained from the Phenoarch phenotyping platform [29]. This platform generates daily data on plant growth. The architecture was adapted to the processes of obtaining, storing, and harmonizing the data. The data that persist in the storage system can be queried interactively for the execution of processing pipelines and data analysis processes on Spark. The Spark RandomForest model was run in this test to predict plant biomass. This value can be considered a measure of plant-scale performance. Figure 4 shows a preview of the POC executed.

Figure 4. Preview of Proof of Concept Results



Source: Prepared by the Authors

Conclusions

The characterization of CYP experiments demonstrated that input data are not constant and contain different levels of detail to assess agricultural data and flexible and scalable storage systems. The proposed system can store these datasets but supports domain-specific and time-series variables. Due to its flexibility, the proposed system supports the centralization of heterogeneous data while imposing a data model that facilitates experiment integration.

Modern big data technology must be used, first and foremost, to handle data variety and combine experiments from many studies to acquire new insights while processing agricultural data. However, volume is not yet a major concern within this domain; as integration issues are overcome, these challenges will begin to emerge. Spark has proved to be a robust tool that unifies data analysis with preprocessing. Since it is agnostic, it is easy to adapt to a document base. With this combination of Spark–MongoDB–Kafka technologies, the requirements presented in R1–R8 are largely covered. The value added using these tools is the ability to store heterogeneous data and distribute the processing of large data volumes.

References

1. Y. Everingham, J. Sexton, D. Skocaj, and G. Inman-Bamber, "Accurate prediction of sugarcane yield using a random forest algorithm," Agronomy for Sustainable Development, vol. 36, no. 2, pp. 1–9, 2016. [Online]. DOI: https://doi.org/10.1007/s13593-016-0364-z
2. D. Jiménez, J. Cock, H. F. Satizábal, A. Pérez-Uribe, A. Jarvis, P. Van Damme et al., "Analysis of Andean Blackberry (rubus glaucus) production models obtained by means of artificial neural networks exploiting information collected by small-scale growers in Colombia and publicly available meteorological data," Computers and electronics in agriculture, vol. 69, no. 2, pp. 198–208, 2009. [Online]. DOI: https://doi.org/10.1016/j.compag.2009.08.008
3. A. Gonzalez-Sanchez, J. Frausto-Solis, and W. Ojeda-Bustamante, "Predictive ability of machine learning methods for massive crop yield prediction," Spanish Journal of Agricultural Research, vol. 12, no. 2, pp. 313–328, 2014. [Online]. DOI: http://hdl.handle.net/20.500.12013/1927
4. G. Ruß, "Data mining of agricultural yield data: A comparison of regression models," Industrial
5. Conference on Data Mining. Springer, 2009, pp. 24–37. [Online]. DOI: https://doi.org/10.1007/978-3-642-03067-3_3
6. R. Lokers, R. Knapen, S. Janssen, Y. van Randen, and J. Jansen, "Analysis of big data technologies for use in agro-environmental science," Environmental Modelling & Software, vol. 84, pp. 494–504, 2016. [Online]. DOI: https://doi.org/10.1016/j.envsoft.2016.07.017
7. S. Delerce, H. Dorado, A. Grillon, M. C. Rebolledo, S. D. Prager, V. H. Patiño, G. G. Varón, and D. Jiménez, "Assessing weather-yield relationships in rice at local scale using data mining approaches," PloS one, vol. 11, no. 8, p. e0161620, 2016. [Online]. DOI: https://dx.doi.org/10.1371%2Fjournal.pone.0161620
8. L.Wasserman, All of statistics: a concise course in statistical inference. Springer Science & Business Media, 2013.
9. D. Jiménez, J. Cock, A. Jarvis, J. Garcia, H. F. Satizábal, P. Van Damme, A. Pérez-Uribe, and M. A. Barreto-Sanz, "Interpretation of commercial production information: A case study of lull (solanum quitoense), an under-researched andean fruit," Agricultural Systems, vol. 104, no. 3, pp. 258–270, 2011. [Online]. DOI: https://doi.org/10.1016/j.agsy.2010.10.004

10. S. Haykin and N. Network, A comprehensive foundation, 2004, vol. 2, no. 2004.

11. G. Ruß, R. Kruse, M. Schneider, and P. Wagner, "Estimation of neural network parameters for wheat yield prediction," IFIP International Conference on Artificial Intelligence in Theory and Practice. Springer, 2008, pp. 109–118. [Online]. DOI: https://doi.org/10.1007/978-0-387-09695-7_11

12. J. R. Quinlan et al., "Learning with continuous classes," 5th Australian joint conference on artificial intelligence, vol. 92. Singapore, 1992, pp. 343–348. [Online]. DOI: https://doi.org/10.1142/9789814536271

13. B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in Proceedings of the fifth annual workshop on Computational learning theory. ACM, 1992, pp. 144–152. [Online]. DOI: https://doi.org/10.1145/130385.130401

14. S. R. Gunn et al., "Support vector machines for classification and regression," ISIS technical report, vol. 14, 1998.

15. L. Breiman, "Random forests," Machine learning, vol. 45, no. 1, pp. 5–32, 2001.

16. J. L. Riquelme, F. Soto, J. Suardáz, P. Sánchez, A. Iborra, and J. Vera, "Wireless sensor networks for precision horticulture in southern Spain," Computers and electronics in agriculture, vol. 68, no. 1, pp. 25–35, 2009. [Online]. DOI: https://doi.org/10.1016/j.compag.2009.04.006

17. V. M. Ngo, N.-A. Le-Khac, M. Kechadi et al., "An efficient data warehouse for crop yield prediction," arXiv preprint arXiv:1807.00035, 2018. [Online]. DOI: https://arxiv.org/abs/1807.00035

18. A. Manjula and G. Narsimha, "Xcypf: A flexible and extensible framework for agricultural crop yield prediction," in Intelligent Systems and Control (ISCO), 2015 IEEE 9th International Conference on. IEEE, 2015, pp. 1–5. [Online]. DOI: https://doi.org/10.1109/ISCO.2015.7282311

19. A. Kamilaris, A. Assumpcio, A. B. Blasi, M. Torrellas, and F. X. Prenafeta-Boldú, "Estimating the environmental impact of agriculture by means of geospatial and big data analysis: The case of Catalonia," From Science to Society. Springer, 2018, pp. 39–48. [Online]. DOI: https://doi.org/10.1007/978-3-319-65687-8_4

20. C. Bazzi, E. Jasse, E. Souza, P. S. Graziano MagalhÃ£es, G. Michelon, K. Schenatto, and A. Gavioli, "Agdatabox-API (Application Programming Interface) a paper from the proceedings of the 14th International Conference on Precision Agriculture," 07 2018.

21. M. Chi, A. Plaza, J. A. Benediktsson, Z. Sun, J. Shen, and Y. Zhu, "Big data for remote sensing Challenges

22. And opportunities," Proceedings of the IEEE, vol. 104, no. 11, pp. 2207–2219, 2016. [Online]. DOI: https://doi.org/10.1109/JPROC.2016.2598228

23. J. Mintert, D. Widmar, M. Langemeier, M. Boehlje, B. Erickson et al., "The challenges of precision agriculture: is Big Data the answer?" in the Southern Agricultural Economics Association Annual Meeting, San Antonio, Texas, no. 230057, 2016. [Online]. DOI: http://dx.doi.org/10.22004/ag.econ.230057

24. C. Rosenzweig, J. W. Jones, J. L. Hatfield, A. C. Ruane, K. J. Boote, P. Thorburn, J. M. Antle, G. C. Nelson, C. Porter, S. Janssen et al., "The agricultural model intercomparison and improvement project (agmip): protocols and pilot studies," Agricultural and Forest Meteorology, vol. 170, pp. 166–182, 2013. [Online]. DOI: https://doi.org/10.1016/j.agrformet.2012.09.011

25. J. W. White, L. Hunt, K. J. Boote, J. W. Jones, J. Koo, S. Kim, C. H. Porter, P. W. Wilkens, and G. Hoogenboom, "Integrated description of agricultural field experiments and production: The Icasa version 2.0 data standards," Computers and Electronics in Agriculture, vol. 96, pp. 1–12, 2013. [Online]. DOI: https://doi.org/10.1016/j.compag.2013.04.003

26. I. Nadareishvili, R. Mitra, M. McLarty, and M. Amundsen, Microservice architecture: aligning principles, practices, and culture. " O'Reilly Media, Inc.", 2016.

27. D. G. Chandra, “Base analysis of NoSQL Database,” Future Generation Computer Systems, vol. 52, pp. 13–21, 2015. [Online]. DOI: https://doi.org/10.1016/j.future.2015.05.003

28. N. Q. Mehmood, R. Culmone, and L. Mostarda, “Modeling temporal aspects of sensor data for MongoDB NoSQL database,” Journal of Big Data, vol. 4, no. 1, p. 8, 2017. [Online]. DOI: https://doi. org/10.1186/s40537-017-0068-5

29. M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, “Spark: Cluster computing with working sets.” HotCloud, vol. 10, no. 10-10, p. 95, 2010.

30. M. Zaharia, R. S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, M. J. Franklin et al., “Apache spark a unified engine for big data processing,” Communications of the ACM, vol. 59, no. 11, pp. 56–65, 2016. [Online]. DOI: https://doi.org/10.1145/2934664

31. P. Neveu, A. Tireau, N. Hilgert, V. Nègre, J. Mineau-Cesari, N. Brichet, R. Chapuis, I. Sanchez, C. Pommier, B. Charnomordic et al., “Dealing with multi-source and multi-scale information in plant phenolics: the ontology-driven phenotyping hybrid information system,” New Phytologist, vol. 221, no. 1, pp. 588–601, 2019. [Online]. DOI: https://doi.org/10.1111/nph.15385