NOVA IMS Information Management School

MAA

Mestrado em Métodos Analíticos Avançados
Master Program in Advanced Analytics

Machine learning approaches for tomato crop yield prediction in precision agriculture

María Fernanda Restrepo Suescún

Internship report presented as partial requirement for obtaining the Master's degree in Advanced Analytics

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

MACHINE LEARNING APPROACHES FOR TOMATO CROP YIELD PREDICTION IN PRECISION AGRICULTURE

by

María Fernanda Restrepo Suescún

Internship report presented as partial requirement for obtaining the Master's degree in Advanced Analytics

Advisor: *Prof Doutor* Flávio Luis Portas Pinheiro

August 2021

DEDICATION

Dedicated to my family and my beloved partner for their unconditional support.

ACKNOWLEDGEMENTS

A special thanks to the company for letting me participate in such an interesting and innovative project. The experience was very enriching for me and contributed a lot to my growth as a Data Scientist.

I also want to express my gratitude to my advisor Flávio Pinheiro who allowed me to work on this project. Through his classes, he gave me the knowledge and valuable tools that I applied in this work, and that will be useful throughout my professional career.

ABSTRACT

The objective of this project was to apply ML techniques to predict processing tomato crop yield given information on soil properties, weather conditions, and applied fertilizers. Besides being robust enough for predicting tomato productivity, the model needed to be interpretable and transparent for the business. The models assessed were *Decision Trees Regression,* ensemble bagging models like *Random Forest Regression,* and boosting techniques like *Gradient Boosting Regression,* and *Support Vector Regression*. Overall, *Gradient Boosting* and *Support Vector* models presented the best performance. For improving the predictive power, we combined the predictions of our two best models into a stacked approach with a Ridge Regression as the final model. The generalization error of the final chosen model on new data was 9.02 ton/ha for the MAE metric, 9.5% for the MAPE, and 13.5 ton/ha for the RMSE. This means that our model can predict tomato crop yield with an approximate error of 9 ton/ha. Even though our final model was complex and not intrinsically interpretable, we were able to apply model-agnostic interpretation methods like the SHAP summary plot to better understand the feature importance and feature effects, and the *Accumulated Local Effects* (ALE) plot, to explain how features influence the outcome of the model on average. In general, the objectives of the project were accomplished and the company was satisfied with the result of the model and its interpretation.

KEYWORDS

Yield prediction; tomato; agriculture; machine learning; ensemble learning.

INDEX

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

AI	Artificial Intelligence
CRISP-DM	Cross Industry Standard Process for Data Mining
DT	Decision Trees
LAD	Least Absolute Deviation
LS	Least Squares
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
ML	Machine Learning
PA	Precision Agriculture
RBF	Radial Basis Function
RMSE	Root Mean Squared Error
RF	Random Forest
SL	Supervised Learning
SVM	Support Vector Machine
UL	Unsupervised Learning

1. INTRODUCTION

Artificial Intelligence (AI) is changing the way businesses face their day-to-day activities. Particularly, there are some economic sectors for which the adoption of AI is fundamental for their development. In that sense, the Agriculture sector is a perfect illustration of how AI can play a worldwide impact in improving the efficiency of its operation. In 2009, the Food and Agriculture Organization of the United Nations published a study suggesting that by 2050 global food production will have to rise by 70% to meet the projected demand (Food and Agriculture Organization of the United Nations, 2009). For this purpose, it is necessary to overcome the challenge of improving crop yield production and quality while reducing operational costs, in other words, producing more and better food with fewer resources. The latter, in fact, is the main objective of Precision Agriculture (PA).

Precision Agriculture is also known as Smart Farming, and as the name implies, it is a strategy that helps farmers make more intelligent decisions based on information and technology. Through the cycle of data collection and analysis, farmers can vary input use and cultivation methods – including the application of fertilizers, irrigation, soil management - to carefully tailoring soil and crop management to fit the diverse settings found in each field. In simple terms, Precision Agriculture aims to help the farmers in "doing the right thing, at the right time, in the right place, in the right way." Precision Agriculture, therefore, differs from conventional farming as it bases its decisions on information and not mainly on intuition, and it links this data to management actions that lead to a reduction in costs, optimization of yields and quality, and protection of the environment (Srinivasan, 2006).

Without the "information revolution," it is unlikely that Precision Agriculture would have developed to its current status. Precision Agriculture relies its data collection process in all kinds of devices such as sensors, cameras, robots, drones, and other artifacts present on the ground 24/7. Similarly, AI, predictive analytics, and other technologies are quickly becoming essential tools for guiding farmers' decision-making process (Forbes, 2021).

1.1. PROJECT DESCRIPTION

This project aims to apply Artificial Intelligence (AI) and Machine Learning (ML) methods to predict tomato crop yield given some attributes such as soil properties, weather conditions, and fertilizer application. Besides being robust enough for predicting tomato productivity, the objective is that the model can be interpretable and transparent for the business.

The project was developed as a Data Science internship at Y Group, a Portuguese company dedicated to the processing of tomato and present also in Spain and Chile. Due to its operations in both hemispheres, the company cultivates and process tomato twice a year which has led it to be one of the bigger producers of tomato concentrate worldwide. Among its main products are Ketchup, Tomato Pulp, Diced Tomato, and Tomato Paste. In the Portuguese market, Y Group is better known for its retail brands. The real name of Y Group was protected by request of the company.

This project is organized as follows. In section 2, we provide a theoretical framework for the work. In Section 3, we explain the methodological workflow. Section 4 presents the results, and finally, section 5 and section 6 summarize the conclusions and limitations found during this work.

2. THEORETICAL FRAMEWORK

In this section, we provide a brief overview of the technical and background knowledge needed to understand this project. The chapter divides into four parts. In subsection 2.1. we summarize the details and concepts underlying the processing tomato harvest cycle. In subsection 2.2. we introduce the agricultural background and concepts mentioned along the project. Subsection 2.3. summarizes the AI methodologies applied. Lastly, subsection 2.4. compiles relevant literature for the subject.

2.1. PROCESSING TOMATO

Processing tomato is a crop grown exclusively to be industrially processed and turned into paste, juice, sauces, or canned peeled tomatoes. The main requirements for being industrially accepted are a bright red color, uniformity, and be free of defects. Additionally, an important feature of processing tomato is the degrees Brix (ºBrix), a measure of the sugar content of an aqueous solution. One degree Brix represents one gram of sucrose in 100 grams of solution. Tomatoes for processing require a minimum ºBrix of 4.5 (Yara, 2021). A combination between color and ºBrix represents the quality of the tomato fruit. The major the ºBrix and the stronger the red color of the fruit, the higher the price. Green and rotten tomatoes are rejected by the factory.

2.1.1. GROWING CONDITIONS

The grown of processing tomato relies on many aspects like soil properties, weather conditions, and proper fertilization and irrigation. It can be produced on soils with a wide range of textures, but it grows optimally in sandy loam or loamy soils. The optimum pH range for production is 6.2 to 6.8. The crop adapts to a great variety of climates but does not tolerate temperatures below 12ºC. The optimum temperatures for growth are from 18ºC to 27ºC. The hot, humid weather coupled with frequent rainfall and mild winters favor the development of diseases (University of Georgia, 2017).

2.1.2. AGRICULTURAL CYCLE

The harvest cycle starts at the beginning of the year when the soil is recovered from previous campaigns and prepared for the next one. Some of the methods to protect the soil during cold seasons include the use of cover crops, a plant that is grown primarily for the benefit of the soil rather than for crop productivity. These cover crops protect the soil from water and wind erosion and contribute later as organic matter. Some examples of cover crops are wheat, oats, and rye. Land preparation should involve enough tillage and soil mobilization to provide the best soil structure for root growth and development of the plants. It could be needed to add some herbicides or phytosanitary products if weeds or diseases are detected. When the soil is prepared, the plantation process begins. In this case, tomato seeds are previously germinated in greenhouses and then transplanted into the field. During the time the tomato is in the soil, fertilization and irrigation are applied depending on the development of the crop and conditions of the soil. Further application of herbicides or pesticides can also be required along the cycle. Depending on the tomato variety, the crop can be on the field between 90 and 120 days. Since it can be difficult to judge externally the maturity of the tomato, farmers often take a representative sample of fruit and cut it open for internal examination. When the tomato has reached the mature stage, the harvesting process begins. In this case, the process is carried out with specialized equipment for tomato harvesting that extracts the tomato from the soil while one or two persons select the classify the tomato able to go to the factory (University of Georgia, 2017).

2.2. AGRICULTURAL BACKGROUND

As mentioned before, many factors influence the growing process of a tomato plant. These factors include soil properties, fertilization, and weather conditions. The objective of this subsection is to explain further these concepts that are going to be mentioned along the project.

2.2.1. SOIL PROPERTIES

Soil properties refers to physical, biological and chemical conditions that characterizes the soils. All soils contain mineral particles, organic matter, water and air. The combination of these determine the soil's properties –its texture, pH, electrical conductivity, organic matter and nutrients.

- **Soil Texture**

Texture refers to the proportion of sand, silt, and clay present in the soil that determines its coarseness or fineness. Texture influences the amount of water the soil can hold, the rate of water movement through the soil, and how workable and fertile it is. For example, sandy soils are light and have a coarse texture what makes them easier to work. They also tend to be well aerated but do not hold much water and are low in nutrients. By contrast, clay soils, in general, hold more water and are better at supplying nutrients. A soil containing equal percentages of sand, silt, and clay is classified as loam and is considered the ideal soil (Queensland Government, 2016). As mentioned before, the tomato grows optimally in sandy loam or loamy soils.

The United States Department of Agriculture (USDA) defined twelve main soil texture classifications: sand, loamy sand, sandy loam, loam, silty loam, silt, sandy clay loam, clay loam, silty clay loam, sandy clay, silty clay, and clay. Figure 1, show this classification and the ranges of percentages of sand, silt, and clay associated with each one.

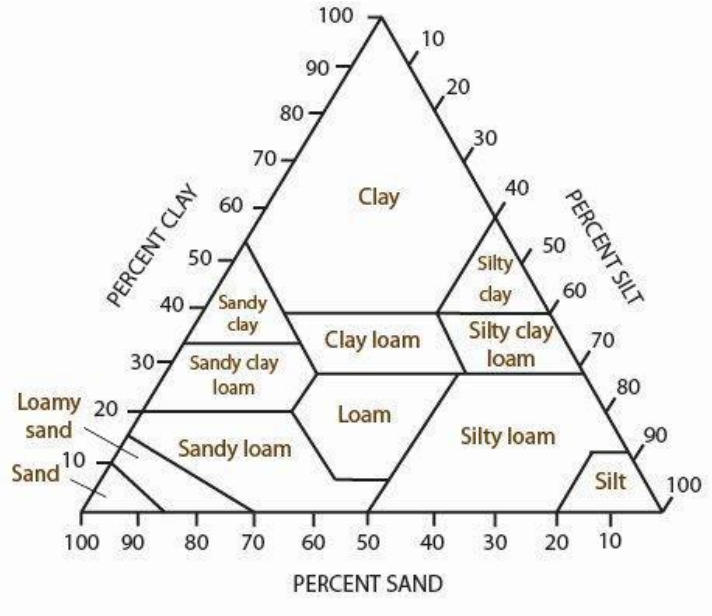

Figure 1. Texture classification according to USDA (Source: Queensland Government, 2016)

- **Soil pH**

pH is an indicator of the acidity, neutrality, or alkalinity present on a soil. Specifically, pH measures the hydrogen ion concentration. Low pH values imply high acidity and high hydrogen ion concentration. On the other hand, high pH values indicate alkalinity and low hydrogen ion concentration. Soils are classified according to their pH values as acidic for pH values less than 6.5 (strongly acidic for pH less than 5.5), neutral with pH values between 6.5 and 7.5, and alkaline for pH over 7.5.

Soil pH influences nutrient absorption and plant growth. Most mineral nutrients are readily available to plants when soil pH is near neutral. In the case of the tomato crop, the optimum pH range for production is 6.2 to 6.8, as mentioned above. Figure 2 shows the range of pH found in soils.

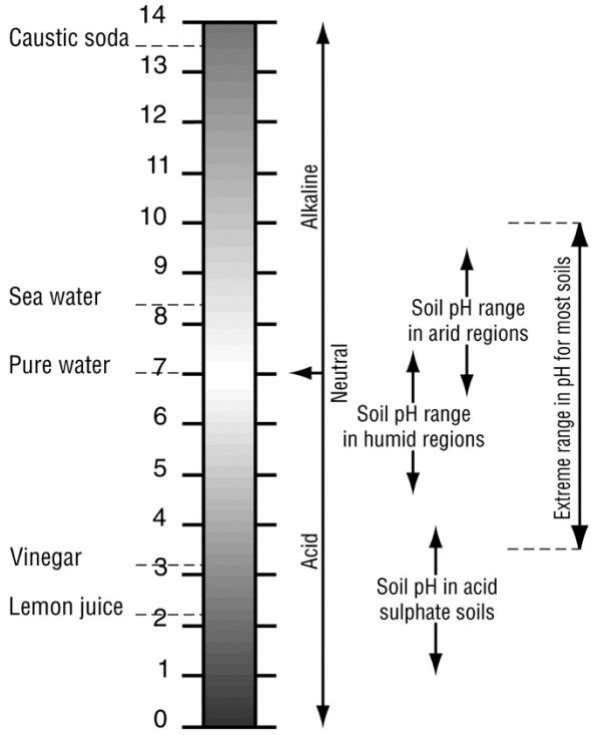

Figure 2. Range of pH found in soils (Source: Queensland Government, 2016)

- **Soil electrical conductivity**

Soil electrical conductivity (EC) is a measure of the ability of the soil to conduct electrical currents. Its unit of measurement can be either milliSiemens per meter (mS/m) or deciSiemens per meter (dS/m). EC is also used to measure the salinity of the soil, and it's an important indicator of soil health. The excess of salt may cause a variety of problems for plants. For instance, poor soil structure, plant nutrient availability, and toxicity for the plants. The tomato crop is moderately tolerant to salt (United States Department of Agriculture, s.f.).

- **Soil organic matter**

Soil organic matter is the organic component of the soil that consists of plant or animal tissue in various stages of decomposition. Organic matter is composed of three main components: plant residues and living organisms, active soil or decomposing organic matter, and stable organic matter or hummus. There are several benefits of organic matter. The first two components contribute to soil fertility as

the decomposition process results in the release of nutrients like nitrogen, phosphorus, or potassium. Organic matter also retains moisture, increases water infiltration into the soil, and contributes to nutrient exchange. Most productive soils have between 3 to 6% of organic matter (United States Department of Agriculture, s.f.).

- **Soil nutrients**

Soil supply of nutrients is key to the health of plants. Nutrients are classified into macro and micronutrients depending on the required quantities. The macronutrients are elements that plants require in relatively large amounts. These are nitrogen (N), phosphorus (P), potassium (K), sulfur (S), calcium (Ca), and magnesium (Mg). The micronutrients, on the other hand, are those that plants require in smaller amounts. These are copper (Cu), zinc (Zn), iron (Fe), manganese (Mn), boron (B), and molybdenum (Mo). The amount of macro and micronutrients available in soils depends on texture, pH, organisms living in the soil, organic matter, and the amount of water. (Queensland Government, 2016).

Table 1 shows the function performed by each of the macro and micronutrients in the plants.

Nutrient	Functions
Nitrogen (N)	Essential component of chlorophyll and protein. Key in growth.
Phosphorus (P)	Important for cellular division and root development.
Potassium (K)	Activator of enzymes present in photosynthesis. Supports synthesis of proteins. Help the plant to resist diseases and cold conditions.
Calcium (Ca)	Component of cell walls. Needed for normal mitosis. Contribute in membrane stability and preservation of chromosome structure.
Sulfur (S)	Helps maintenance of protein structure. Metabolism of vitamins.
Magnesium (Mg)	Component of chlorophyll molecule and essential for photosynthesis.
Iron (Fe)	Supports synthesis and preservation of chlorophyll.
Manganese (Mn)	Necessary in the plant's respiratory process.
Boron (B)	Support activities of some enzymes. Enables carbohydrate transport.
Zinc (Zn)	Supports use of phosphorus and nitrogen. Key in protein synthesis.
Copper (Cu)	Helps the creation of Vitamin A in plants.
Molybdenum (Mo)	Related with nitrogen utilization and fixation.

Table 1. Functions of macro and micronutriens in plants (Source: Food and Agriculture Organization of the United Nations, 1984)

2.2.2. SOIL FERTILIZATION

Plants, as all living beings, need food for their growth and development. However, unlike animals that need food in organic form, plants can make their own food directly from inorganic materials. They form organic tissues by taking water and mineral nutrients from the soil, carbon dioxide from the air and energy from the sun. This process of making their own food is called photosynthesis.

The essential elements for plants are Carbon (C), Hydrogen (H) and Oxygen (O), which are extracted exclusively from air and water. Additionally, the micro and macronutrients mentioned above, are essential for the plants and can be absorbed from the soil or through application of fertilizers. A

productive soil should contain all the essential plant nutrients in sufficient quantity and in balanced proportions to allow plants to grow and develop to their full potential. However, from the macronutrients there are three elements that plants need in larger quantities and are prioritized in the fertilization plans: Nitrogen, Phosphorus and Potassium. These nutrients are applied to the soil as fertilizers, depending on the soil tests results and if required by the plant during the campaign. These products are commonly known as NPK fertilizers, because of the chemical symbols of the macronutrients that compose them (Food and Agriculture Organization of the United Nations, 1984).

2.2.3. WEATHER CONDITIONS

Weather conditions have a significant impact on the tomato crop cycle. As these are factors that cannot be controlled by the farmers, it is interesting to understand their effect on the tomato crop yield to anticipate scenarios. Following, we briefly explain the climate parameters that are mentioned throughout the project.

Precipitation: refers to all aqueous particles in a liquid or solid state that originate in the atmosphere and fall to the earth's surface by the action of gravity. The usual unit of measurement of precipitation is millimeters (mm) of liquid water depth collected at a given point over a specified period.

Solar radiation: is the amount of electromagnetic radiation released by the sun and received at the earth's surface per unit area. It is measured in watts per square meter (W/m2).

Wind speed: refers to the ratio of the distance covered by moving air to the time taken to cover it. This air movement is caused by fluctuations in temperature. Wins speed is measured in meters per second (m/s) or kilometers per hour (Km/h).

Leaf wetness duration: refers to the time of continual contact of the leaves with liquid moisture, on both their top and bottom sides. This measure is key for fungus and disease control. The unit of measurement is usually minutes.

Temperature: is the quantity measured by a thermometer and that expresses hotness or coldness. Temperature can be measured in the Celsius scale (ºC), Fahrenheit scale (ºF), or Kelvin scale (K).

Relative humidity: refers to the quantity of actual water vapor that is in the air in comparison with the maximum possible. The relative humidity is generally reported as a percentage.

Dew point: is the temperature to which air needs to lose heat to become condensed with water vapor. As it is a temperature, the dew point can be measured in ºC, ºF, or K.

Evapotranspiration: is the amount of water transferred from the earth to the atmosphere by evaporation from the soil, and by transpiration from the leaves of the plants growing on it. Evapotranspiration is measured in mm.

(American Metereological Society, 2020) (Encyclopedia Britannica, 2021)

2.3. ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

In this subsection, we will briefly introduce the methods and techniques applied in this project, specifically, a short description of Machine Learning concepts and techniques.

2.3.1. OVERVIEW OF MACHINE LEARNING

Machine Learning can be defined as the process of solving a practical problem by gathering a dataset, and then algorithmically building a statistical model based on that dataset (Burkov, 2019).

Machine Learning models divide into two groups which aim at solving different problems, these groups are the models based on Supervised Learning (SL) and those that are based on Unsupervised Learning (UL). Supervised Learning models use labeled datasets to train algorithms to learn a specific task. Such a task could be either to classify data in categories (Classification) or predict outcomes accurately (Regression). The training datasets include inputs and correct outputs allowing the algorithm to learn over time. On the other hand, the principal objective of Unsupervised Learning is to discover hidden patterns or clustered unlabeled datasets through algorithms. Unsupervised Learning models are used for three main tasks – clustering, association, and dimensionality reduction (IBM Cloud Education, 2020). For this project, the focus will be on the Supervised Learning techniques for Regression.

Two main goals are key on Supervised Learning: model selection and model assessment. In model selection, we estimate the performance of different models to choose the best one, and in model assessment, having chosen a final model, we estimate its prediction or generalization error on new data. Generalization of a model is key on Supervised Learning and is related to the ability of a model to make accurate predictions on unseen data. The general approach for both model selection and model assessment tasks is to randomly divide the dataset into three parts: a training set, a validation set, and a test set. The training set is used to fit the models, the validation set is used to estimate prediction error for model selection, and the test set is used for assessment of the generalization error of the final chosen model (Hastie, Tibshirani, & Friedman, 2017). Figure 3 shows a typical split configuration where 50% of the dataset is used for training and 25% each for validation and testing.

Figure 3. Typical train - validation - test split

However, in some situations, the amount of data for training and testing will be limited, and to build proficient models it is necessary to use as much of the available data as possible for training. This is the case of this project where we don't have enough data to split it in a typical train - validation - test split. One approach for cope with this situation, it to use k-folds cross-validation where the dataset is split into k equally sized subsets or folds. One of the k-folds will act as the test set and the remaining folds will be used for training the model. This process is repeated for all k possible choices for the test set and the performance score from the k runs is then averaged (IBM Cloud Education, 2020). Figure 4 illustrates the k-fold cross-validation process for a case of five folds.

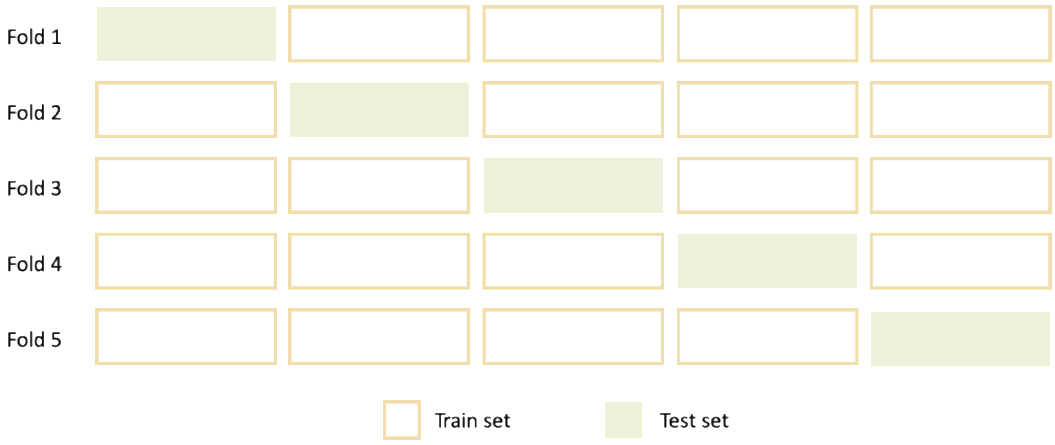

Figure 4. K-folds cross-validation process

One of the most important concepts in machine learning is the bias-variance trade-off. The bias of a model refers to the error introduced by the inability of the algorithm to capture the true relationship of the data. A model with no bias will never make any errors in prediction while one with high bias will fail to accurately predict the target variable. On the other hand, the variance of a model represents its sensibility to changes in the training set. A model with low variance has few fluctuations when built with new data while a model with high variance is very sensitive to the training data. The bias-variance trade-off represents how it is easy to decrease bias at the expense of increasing variance, and vice versa. In machine learning, the ideal algorithm has low bias by accurately modeling the true relationship, and has low variability by producing consistent predictions across different datasets. Figure 5, illustrates three situations in the bias-variance trade-off. The leftmost scenario represents a very simple model that does a bad job classifying the target variable and presents high bias. This situation where the model is unable to capture the relationship between the input and output variable accurately is called underfitting. In opposition, the rightmost scenario shows a complicated model that passes through all the data points. This model is memorizing the noise instead of capturing the approximate relationship and will probably fail to generalize on new data. This situation in which the model fits exactly against its training data is called overfitting (Fischetti, 2018).

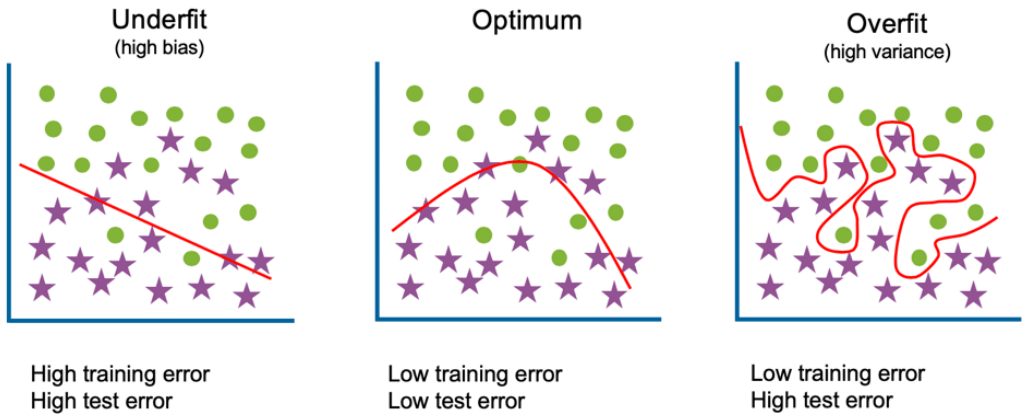

Figure 5. Scenario in the bias-variance trade-off (Source: IBM Cloud Education, 2020)

In contrast, the middle graph in Figure 5, represents an optimum situation and the goal when fitting a model: the “sweet spot” between underfitting and overfitting. Figure 6, illustrates better this ideal point in this bias-variance trade-off. As a model becomes more complicated, its bias or error in the training set continues to decrease. As the fitting process continues to iterate and the model begins to fit the data better and better the error in the validation set decreases as well. At some point, the model becomes very flexible and begins to memorize the noise in the training set or overfit, while the error in the validation set goes up. The sweet spot is the point when this validation error is minimized (Fischetti, 2018).

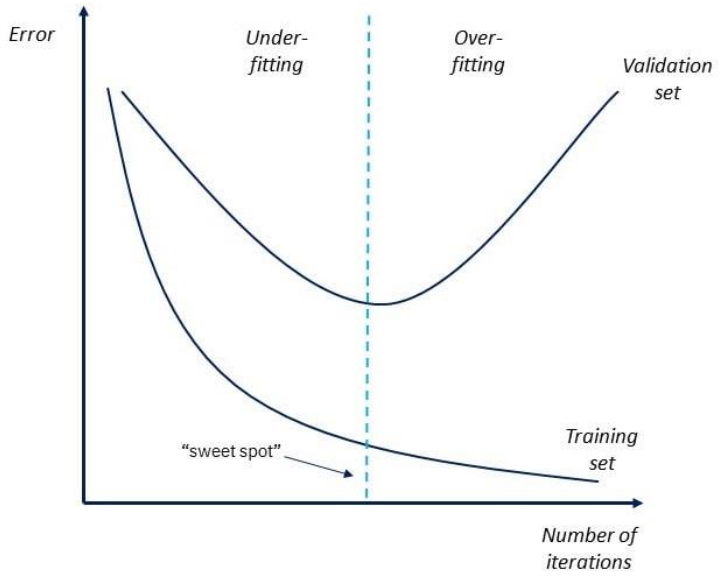

Figure 6. "Sweet point" between underfitting and overfitting (Source: IBM Cloud Education, 2020)

Following, we briefly described some popular ML models that will be used later in this project.

2.3.2. DECISION TREES

Decision trees (DT) are one of the most easily interpretable and popular models used for classification and regression purposes. This approach is very useful in situations where decisions should be transparent and easily understood and explained. This situation is common in the agricultural context. A decision tree looks like an upside-down tree with the trunk on top and the leaves on the bottom. The very top of the tree is called root node, the next ones are internal nodes or branches, and the last ones are called leaf nodes. A decision tree can be defined as a collection of decision nodes, connected by branches, extending downward from the root node until ending in leaf nodes. Starting at the root node, a series of splitting rules are applied to the attributes, with each possible outcome resulting in a branch. Each branch then leads either to another decision node or to a terminal leaf node. Each split point on the decision tree is chosen carefully to result in the most informative split (Larose & Larose, 2014) (Fischetti, 2018). Figure 7 illustrates the structure of simple decision tree.

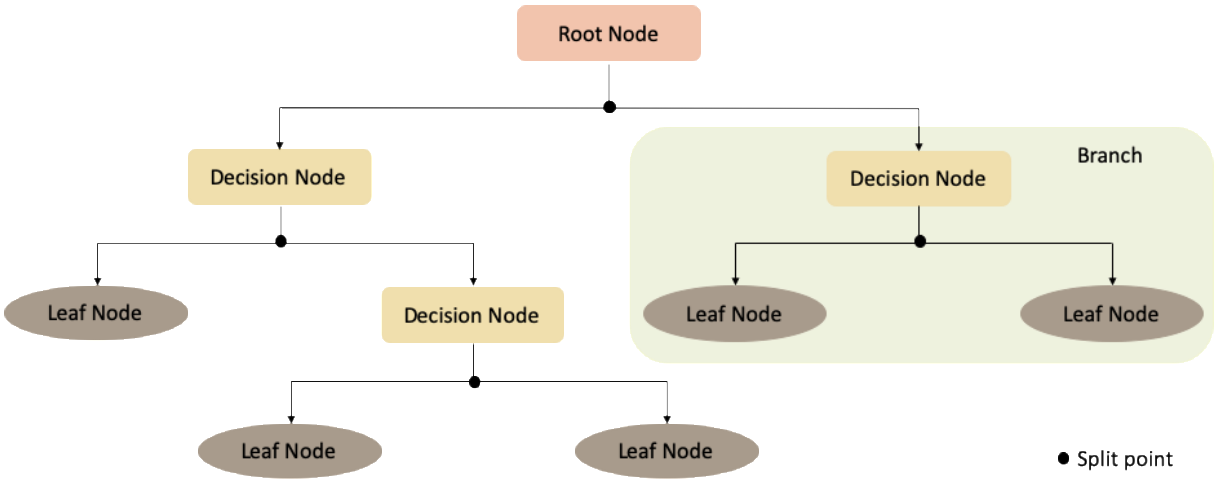

Figure 7. Structure of a Decision Tree

The recursive splitting process of building a tree can be summarized in the following steps:

1. Select a variable and split point that results in the best outcome. The first iteration will be for choosing the variable on the root node. For the case of classification, the idea is to choose the variable that more contributes to the decrease in impurity on the subsequent leaves. This is measured through Gini impurity and entropy reduction or information gain. For the case of regression, a reduction in the sum of squared residuals is measured.
2. For each of the resulting branches, check to see if some stopping criteria are met. For example, if a node pure is reached, there is no reason to continue splitting to reduce impurity and we will have arrived at a leaf node. Other stopping criteria that can be defined by us, is either a certain depth that the tree can't exceed or a minimum number of observations required by a leaf node.
3. Repeat the steps below on the branches that do not meet the stopping criteria.

One critical disadvantage of decision trees is that if there are not stopping criteria, as mentioned in step 2, the model will be prone to build complicated and large trees with leave nodes containing only one observation. In other words, the decision trees are prone to overfit the training data and not to generalize well on unseen data. These big and complex decision trees are models with low bias and high variance. A smaller tree with fewer splits will probably have lower variance and will generalize better at the cost of a little bias. One approach to reach such a simpler yet powerful tree is through a technique called pruning where complex branches that contribute little, are cut down and removed (Fischetti, 2018).

In summary, decision trees are simple and useful for interpretation. However, due to their high variance condition, their predictive performance is not competitive with other unsupervised learning techniques.

One of the approaches for dealing with the poor performance of decision trees is ensemble learning. That is a technique based on the idea of the "wisdom of crowds" which suggests that the decision-making of a larger group of people is usually better than the one of a single expert. This way, ensemble

models aggregate a group of single models to achieve a better final prediction. These single models, also known as weak learners, work together to form a strong learner and reduce bias or variance. Bagging and Boosting, are two main types of ensemble methods. The former is commonly used for variance reduction, while the latter is typically used for decreasing bias.

In bagging (acronym of bootstrap aggregation), a random sample of data in a training set is selected with replacement. That process is known as bootstrapping. After several bootstrapped datasets are generated, weak models are trained in parallel, and their outcomes are combined to reduce variance. For regression, we fit independent regression trees for the bootstrapped datasets and then average the results. For classification, we select the majority vote of the group of trees as the predicted class. One extension of the bagging method is the random forest algorithm, which uses bagging and feature randomness to create an uncorrelated forest of decision trees. We will talk about it in more detail later.

Boosting, on the other hand, combines a group of weak models into a stronger one to minimize training errors or reduce bias. Unlike bagging where the weak learners are trained in parallel, in boosting they learn sequentially, this means that each model tries to reduce the errors of its predecessor. Gradient boosting, an example of boosting algorithms is explained later (IBM Cloud Education, 2020).

2.3.3. RANDOM FOREST

As mentioned above, Random Forest (RF) model belongs to the ensemble methods that use the bagging technique. The process for building a RF is the following (Hastie, Tibshirani, & Friedman, 2017):

1. Create a bootstrapped dataset, that is, to randomly select samples from the original dataset with the possibility of picking the same sample more than once.
2. Grow a decision tree using the bootstrapped data set, but instead of using all the features, select a random subset of variables.
3. Repeat steps 1 and 2 until the number of desired individual decision trees is reached.
4. Aggregate the results of the ensemble of trees. For regression average the result of each one, and for classification calculate the majority vote of the group.

Random forest has played a relevant role in ML lately due to key advantages like reducing the risk of overfitting, and its convenience at evaluating variable importance or contribution to the model. However, the algorithm also presents some challenges like high time consumption when training and the resource requirements. Additionally, the interpretability of a random forest output is more challenging than the one of a single decision tree (IBM Cloud Education, 2020).

2.3.4. GRADIENT BOOSTING

As for bagging models, boosting has the objective of combining many weak learners to produce a more powerful model. However, unlike bagging methods like random forest, which rely on a simple average of the individual learners, the idea of boosting is to sequentially improve the ensemble. Gradient boosting belongs to these types of models, and its name is because it combines the gradient descent algorithm and the boosting method (IBM Cloud Education, 2020).

Gradient boosting models starts with a weak model that can be a single leaf or a decision tree with only a few splits. The algorithm sequentially boots the performance by fitting new trees to the residuals

of the previous ones. The new tree on the sequence will focus on the observations where the previous tree had the largest prediction error trying to fix them up. Gradient boosting is recognized as a gradient descent algorithm. This algorithm tries to adjust parameters iteratively to minimize a cost function. An important parameter in gradient descent is the size of the step to find the minimum which is controlled by a *learning rate,* and that also deals with overfitting.

2.3.5. SUPPORT VECTOR MACHINES

Support Vector Machines (SVM), is a technique used for both classifications and regression problems, developed in the '90s. This approach is considered one of the best "out of the box" models for showing good performance in many scenarios, including the scenario of small datasets as in our case.

The SVM is a generalization of a simple and intuitive classifier called the *maximal margin classifier*, which is the optimal hyperplane selected in the case where two classes are linearly separable. In a *p*-dimensional space, a hyperplane is a flat subspace with dimension *p-1*. For example, in two dimensions a hyperplane is a straight line, and in three dimensions, a plane. If the data can be perfectly separated using a hyperplane, then there will be an infinite number of them and it will be necessary to choose one. A reasonable way of choosing this hyperplane is to calculate the distance from each training observation to a given separating hyperplane and compute the *margin*, that is the minimal distance from the observations to the hyperplane. The selected classifier will be based on the *maximal margin hyperplane*, the separating hyperplane for which the margin is largest. Figure 8 shows in solid line the maximal margin hyperplane that separates two classes represented by the blue and purple points. The margin, represented by the arrows, is the distance from the solid line to the dashes lines. The points that lie on the dashed lines are called *support vectors* which are data points that are closer to the hyperplane and support it in the sense of influencing its position and orientation.

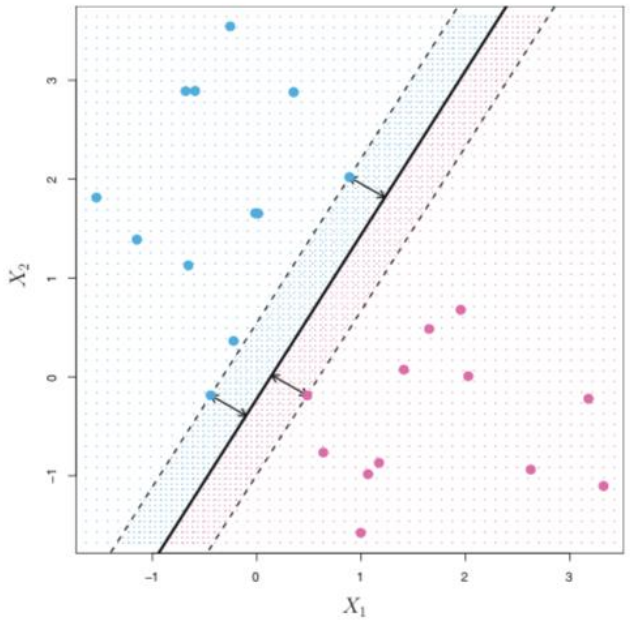

Figure 8. Maximal margin hyperplane (Source: James et al., 2013)

Although the maximal margin classifier is simple and sometimes successful, it is not realistic and can't be applied to most datasets as it requires the classes to be separable by a linear boundary. Besides,

this approach can sometimes lead to overfitting since the maximal hyperplane is extremely sensitive to a change in a single observation. A more robust alternative to the maximal margin classifier is the *support vector classifier,* also known as *soft margin classifier* since the margin is soft and allows some misclassification in the interest of greater robustness to individual observations and better classification of most of the training observations.

The support vector classifier is a powerful approach for non-separable cases. However, it has the shortcoming of relying exclusively on linear decision boundaries. A further extension of the support vector classifier that deals with nonlinear relationships is the *support vector machine (SVM).*

SVM works by enlarging the feature space using quadratic, cubic, and even higher-order polynomial functions of the predictors called kernels. In other words, SVM uses a kernel to implicitly transform the original space into a higher dimensional space during the training process. The word "implicitly" is key because the task of the kernel is to calculate the high-dimensional relationships without doing this transformation explicitly. This property of the kernels known as the *kernel trick* reduces the amount of computation required.

Specifically, a kernel is a function that quantifies the similarity of two observations. For instance, a *linear kernel* uses Pearson correlation to quantify such similarity. There are also *polynomial kernels* that lead to a more flexible decision boundary by fitting a support vector classifier in a high dimensional space involving polynomials. Another example of a nonlinear kernel and a popular choice is the *radial kernel.* Figure 9 shows two different kernels applied to nonlinearly separable data; in the left, a polynomial of degree 3, and in the right a radial kernel. Both kernels can capture the decision boundary (James, Witten, Tibshirani, & Hastie, 2013) (Burkov, 2019).

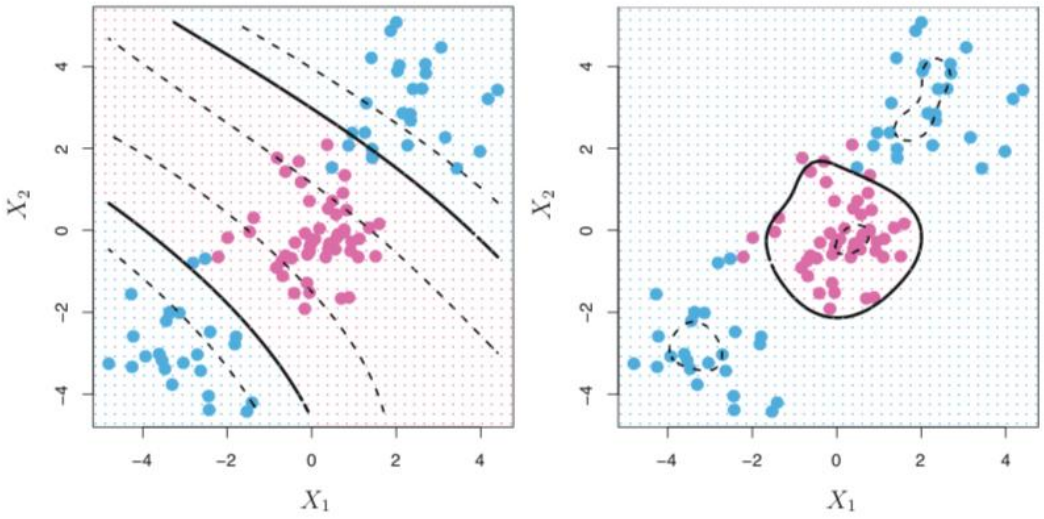

Figure 9. SVM with a polynomial and a radial kernel (Source: James et al., 2013)

The SVM concepts can be generalized to regression problems with the *support vector regression* (SVR). This model find the optimal regression hyperplane so that most training samples lie within a certain margin around it, and uses a symmetrical loss function, which equally penalizes high and low misestimates. As in classification, SVR uses kernels for the non-linear approach and is not computationally expensive. SVR has been proven to be an effective tool for estimation with prediction accuracy and generalization capabilities (Awad & Khanna, 2015).

2.3.6. STACKED MODELS

Stacking is a technique used for both classifications and regression problems that combines the prediction of several base learners. Unlike ensemble models that aggregate a group of weak learners to achieve a better final prediction, stacking is designed to ensemble a group of strong learners. First, the base learners are trained individually, then a final model called the *super learner* computes the final prediction based on the predictions of the base learners. Such staked models tend to perform as well or better than any of the individual learners in the staking. A common approach for training a staking model is as follows:

1. Select and train a set of individual base learners, and find the optimal hyperparameters that provide the best predictions. Collect the predictions from each base learner which will be the input values for the stacking model. For these models to be part of the stacking they all must be trained on the same dataset and with the same distribution of cross-validation folds.
2. Specify and train the final model that will learn from all the individual predictions and generate predictions on new data.

Stacking presents better results when the individual base learners have different predictions. If the individual models generate similar predictions, the benefit of combining them will be lower (Boehmke & Greenwell, 2020).

2.3.7. INTERPRETABILITY IN MACHINE LEARNING

Machine learning has been gaining popularity in business due to its predictive power in certain tasks. However, there are situations in which besides the predictions themselves, the reasons behind such predictions are also important for the business. Interpretability can be defined as the degree to which a human can understand the cause of a decision. Some models are intrinsically interpretable and transparent, for instance, linear regression. However, these interpretable models tend to be simple and relatively inflexible. Typically, as the complexity of a model increases, interpretability is lost.

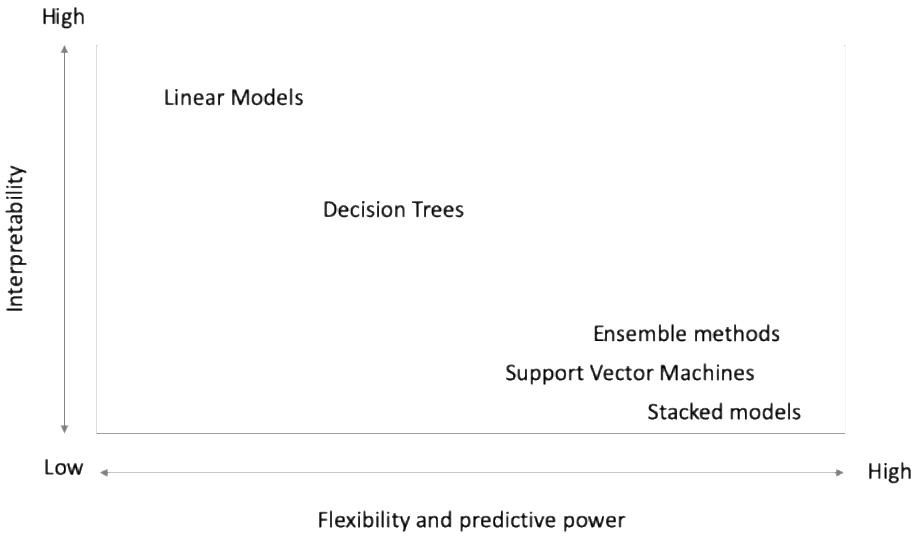

Figure 10. Trade-off between flexibility and interpretability

Figure 10 shows this trade-off between flexibility and interpretability of the models that we have previously described. As mentioned above, decision trees are simple and useful for interpretation but their predictive power is not competitive with other unsupervised learning techniques. On the other hand, approaches like the ensembles and SVM are powerful but not interpretable. Fortunately, there are model-agnostic interpretation methods that can be applied to any model and can especially help to explain these kinds of "black-box" approaches.

One popular method for interpretation of ML models is the *Partial Dependence Plot (PDP)* which indicates the marginal effect that the variables have on the predictions of a model and the kind of relationship between the outcome and the variables. This method is intuitive and easy to interpret but has a big disadvantage of assuming that the variables are not correlated, which is non-realistic in some scenarios. If the features of a model are correlated, then the partial dependence plot can't be trusted. Luckily there is an unbiased alternative to PDP, the *Accumulated Local Effects* (ALE) plot, which describes how variables influence the outcome of a model on average (Molnar, 2021).

To manage correlated features, the *Accumulated Local Effects* plot makes use of the conditional distribution to separate the effect of the feature of interest. To do so, it creates augmented data based on the conditional distribution (Oracle, 2021). To estimate the local effects, the variable is separated into several intervals, and then differences in the predictions over the augmented data are calculated, lastly, those differences are averaged. In an ALE plot, each line is interpreted as the change in the model prediction when the selected feature has a given value compared to the average prediction (Molnar, 2021).

Alternatively, there is another interesting model-agnostic method for interpretability called *Shapley Values.* This method is based on cooperative game theory and tries to assign payouts to players depending on their contributions to the total gain. When applied to the ML scenario, the players are the feature values that collaborated to predict a certain value. The Shapley value is the average of the marginal contributions of a variable value among all the possible combinations of features (Molnar, 2021) (Oracle, 2021).

For a better understanding of the concept of Shapley value in the context of Machine Learning, let's give an example. Let's imagine we have trained a model to predict car prices. For a certain car, the model predicts a price of 30,000 euros and we want to explain this prediction. The car is five-year-old, is electric, and its color is black. If the average price prediction for all cars is 25,000 euros it would be of interest to know the amount contributed by each variable to a certain prediction compared to the average prediction. In our example, the gain would be the difference between the prediction for that car and the average prediction of all cars, in this case, 5,000 euros. The players would be the variables that cooperated to obtain the gain, in this case, the fact that the car is five-year-old, is electric, and its color is black. If we were to calculate the Shapley value for the feature *car_color = black,* we would need to calculate all possible coalitions or combinations among the rest of the variables. The coalitions would be the following four: *1. No feature values, 2. car_age = 5 years, 3. car_motor = electric, 4. car_age = 5 years + car_motor = electric.* For each combination, we would need to calculate the predicted price with and without the feature *car_color = black* and compute the difference to get the marginal contribution. When averaging those differences, we will have calculated the Shapley value.

2.4. LITERATURE REVIEW

In this subsection, we will give an overview of relevant scientific literature on the use of Artificial Intelligence and Machine Learning in the context of yield prediction in agriculture. Since still, rare studies involve this kind of methodologies, we will mention a few that are most related to our project. No studies of tomato crop yield prediction were found.

In their study, (Shahhosseini, Hu, & Archontoulis, 2020) aimed to use Machine Learning models to forecast corn yield in the states of Illinois, Indiana, and Iowa of the United States. For that purpose, they collected data from 2000 to 2018 on soil and weather conditions as well as management variables. After some phases of feature selection, they decreased the number of variables from 597 to 72. Several Machine Learning models were trained: from linear and Lasso regression to ensembles like random forest and Extreme Gradient Boosting (XGBoost). Additionally, they proposed and tested several customized stacked and ensemble models. For the training and validation process, the authors opted for a sequential k-folds cross-validation approach since they were in a time series scenario. The results showed that a Stacked Lasso Regression made the least biased prediction. For interpretability of the models, they opted for Partial Dependence Plots and Feature importance.

Likewise, (Kumar Srivastava, et al., 2021), compared the performance of several Machine Learning methods for predicting winter wheat yield in 271 counties across Germany for 41 years (1979-2019). The information they counted on consisted of weather, soil, and crop phenology data (sowing, flowering, and harvest dates) weekly aggregated. The model tested included simple methods like K-nearest neighbors, Lasso and Ridge Regression and Regression Trees, and more complex ones like Random Forest, Support Vector Regression, Extreme Gradient Boosting, and Deep Neural Networks. After performing feature selection and hyperparameter tuning using grid search, they proceed with the training phase with a k-folds cross-validation strategy. The obtained results showed that the Deep Neural Networks model presented the performance, followed by the Extreme Gradient Boosting. By contrast, linear models achieved the worst performance as they didn't capture the nonlinear effects of weather and soil conditions. For interpretability, the authors calculated Shapley values of features.

On the other hand, (Meroni, Waldner, Seguinia, Kerdiles, & Rembold, 2021) assessed the performance of Machine Learning models for forecasting crop yield when having small data. Specifically, they took the case study of Argelia to predict yield for its main cereal crops: barley, soft wheat, and durum wheat, in 20 provinces. To do so, they used monthly data of 17 years including a set of satellite-derived vegetation and weather variables. Among the tested models, there were Lasso Regression, Random Forest, Support Vector Regression, Gradient Boosting Regression, and Neural Networks. Hyperparameters of each model were optimized using an exhaustive grid search. Given the small data condition, they used a leave-one-out cross-validation strategy, where they left out one year at each iteration. The authors tested the Machine Learning models against naïve benchmark methods like averages and very simple linear regression for predicting crop yield. Surprisingly, the simple benchmark models outperformed 60% of the Machine Learning models, and the difference in accuracy between the best models and the naïve ones was not always significant.

3. METHODOLOGY

The selected methodology for developing this project was the Cross Industry Standard Process for Data Mining (CRISP-DM). This is a process model with six phases that provides a framework for designing, creating, testing and deploying machine learning solutions (Stirrup & Oliva Ramos, 2017). Figure 11 illustrates the six phases of CRISP-DM methodology.

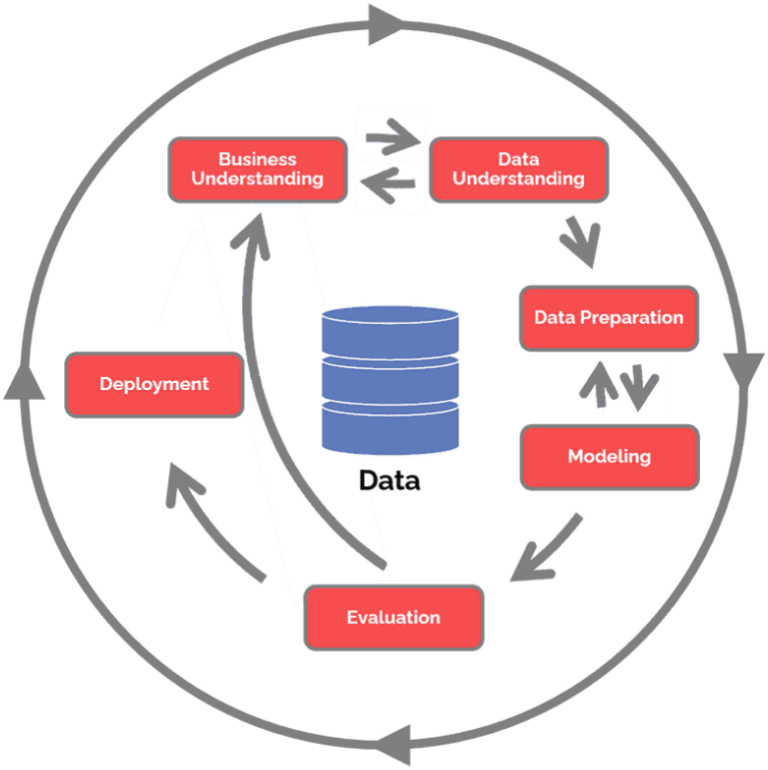

Figure 11. CRISP-DM methodology (Source: Data Science Project Management)

Following, we will briefly explain the six stages of the CRISP-DM methodology. However, for this document, the focus will be mainly on the Modeling stage, in which we will dive into more details.

I. Business Understanding

This stage focuses on meticulously understanding the objectives and requirements of the project from the business perspective. It requires defining a scope of what is wanted to be accomplished, determine resources availability, define the business objectives and produce the project plan. Additionally, in this stage, the business needs are looked at from a data science technical perspective, selecting the technologies and tools that are going to be used to approach the problem.

II. Data Understanding

The objective of this phase is to identify, collect and analyze the data sets that can contribute to the accomplishment of the project goals. This stage requires describing and exploring the data sets, identifying relationships among the variables, and verifying the data quality. This process must be constantly followed by the business experts, since their feedback is key for the understanding of the business problem and the data itself.

III. Data Preparation

The goal of this phase is to clean, transform and prepare the data for modeling. This stage requires selecting the final data sets that will be used, clean the data to correct or remove erroneous values, derive new attributes, integrate the data sets, and re-format as necessary. This phase of the process is probably the one that takes the most time, usually 80 % of the project duration, and can make a difference at the modeling stage.

IV. Modeling

In this phase, the objective is to apply and assess various modeling techniques to the data. Here the models are iteratively improving and refining. If necessary, it may involve going back to the data preparation to correct any unexpected issue. This phase requires selecting the modeling techniques, generating the test design, build and assess the models. Generally, multiple models are evaluated at the same time and compete against each other.

V. Evaluation

In this stage, the models are tested and verified to evaluate if they meet the business objectives and requirements defined in the first step. This phase requires reviewing that the process has been properly executed, summarizing findings, and correcting anything if needed. Based on this evaluation, it is decided whether to proceed to deployment or continue iterating the process.

VI. Deployment

In this final phase, the selected model is put into production to be used by the users. The complexity level of the deployment process depends on the business requirements. It is necessary to develop an exhaustive monitoring and maintenance plan to avoid issues during the production phase. It is also important to conduct a detailed review of the whole process and notice issues that could be improved in the future (Data Science Project Management).

Following, we will describe the application of each step of the CRISP-DM methodology in this project. Some steps will be explained more deeply than others, depending on their relevance for this document.

3.1. BUSINESS UNDERSTANDING

In this project, the principal objective of the company was to be able to predict the processing tomato productivity given some factors like the soil properties, the weather conditions, and the fertilization applied. Additionally, it was important for the company to understand the variables that most influenced the tomato crop yield.

From a Data Science perspective, the goal was to create an ML model for predicting productivity. Besides being robust enough for predicting tomato crop yield, this model had to be highly interpretable. The “black-box” types of models are not a good idea in industries like agriculture, where people tend to be a little skeptical about these kinds of approaches.

For this purpose, the principal resource the company had, was historical data of productivity of annual campaigns, soil analysis from laboratories, and past weather conditions. This information, though, was

not organized and not structured. At first, we had to do the arduous work of data collection and data architecture. Although it was not the objective of this project, it is worth mentioning.

3.2. DATA UNDERSTANDING

As mentioned before, this stage was very complex since we had to collect and structure all the data sources. Such sources came from several suppliers like internal people, laboratories, and farm management software. Additionally, the information was in different formats like Excel files, Web Services, Relational Database Views, and even PDF files.

For the process of integration, we designed and built a relational database architecture with Microsoft Azure services. We created an Azure SQL server and database, and the information was integrated through the Data Factory Azure service. We end up with a normalized database that allowed us to access and use the data efficiently.

Following, we explain deeper each data source and its structure in the relational database system.

3.2.1. LAND PARCEL INFORMATION

The administrative division of the lands owned by the company consists of two basic structures: zones and parcels. The parcel is the minimum unit of land and zones are a group of neighboring parcels. The company owned 51 parcels distributed in 5 zones. The *Parcels* table in the relational database system consisted of the list of parcels and the zone to which each one belonged and a code associated with both parcel and zone. Table 2 shows the distribution of the parcels in the different zones. Because of confidentiality policies, we didn't show the actual names but instead used a codification.

Zone	Number of Parcels
A	6
B	5
C	16
D	10
E	14
Total	51

Table 2. Distribution of zones and parcels

3.2.2. SOIL PROPERTIES INFORMATION

Annually, three to five months before the beginning of the plantation campaign, soil samples are taken on each one of the parcels and these are sent to laboratories to analyze the soil properties. This information is crucial to understand the conditions of the soils and evaluate their nutrients needs. Before this project, this information was used exclusively for operational purposes, but the effects of the soil's properties on tomato productivity had not been studied previously. As mentioned above, this historical information had to be collected from several sources, formatted and integrated into our relational database system into a table called *Soil_Properties* at Parcel level. Table 3 shows the soil properties information that was gathered, its units of measurement, and the data type of the variable.

Parameter	Unit of measurement	Data type
Sand	%	Numeric
Clay	%	Numeric
Silt	%	Numeric
pH	-	Numeric
Electrical conductivity (EC)	mS/m	Numeric
Nitrogen (N)	mg/Kg	Numeric
Phosphorus pentoxide (P_2O_5)	mg/Kg	Numeric
Potassium oxide (K_2O)	meq/100g	Numeric
Organic Matter (OM)	%	Numeric
Calcium (Ca)	meq/100g	Numeric
Magnesium (Mg)	meq/100g	Numeric
Iron (Fe)	mg/Kg	Numeric
Manganese (Mn)	mg/Kg	Numeric
Copper (Cu)	mg/Kg	Numeric
Zinc (Zn)	mg/Kg	Numeric
Boron (B)	mg/Kg	Numeric
Sodium (Na)	meq/100g	Numeric
Calcium carbonate ($CaCO_3$)	mg/Kg	Numeric
Calcium - Magnesium ratio (Ca:Mg)	-	Numeric
Potassium - Magnesium ratio (K:Mg)	-	Numeric
Carbon - Nitrogen ratio (C:N)	-	Numeric
Soil Texture (USDA rules)	-	Categorical

Table 3. Soil properties parameters

As mentioned in the Theoretical Framework section, one of the units of measurement for electrical conductivity is milliSiemens per meter (mS/m). Regarding the units of measurement for the macro and micronutrients, *mg/Kg* refers to the concentration in milligrams of the nutrients per one kilogram of soil, and *meq/100g* stands for milliequivalents per 100 grams of soil. This last unit is used to report the cation exchange capacity (CEC) of the soil which is an indication of the amount of negative charges present on it.

Following we show some data exploration we performed on soil properties parameters.

- **Texture**

Figure 12, created with the *Plotly* Python package, maps all parcels in the soil texture classification defined by the United States Department of Agriculture (USDA). Each of the points represents a sample taken from a parcel and the form of the point indicates the zone to which the parcel belongs. In some parcels, there were taken more than one sample. From the figure, it can be highlighted that most of the parcels belong to the Clay, Silty clay, and Silty clay loam classification. All the parcels from zones A and C belong to these kinds of soils. On the other hand, all parcels from zone B have sandy soil. Finally, soils from zones D and E are diverse in texture.

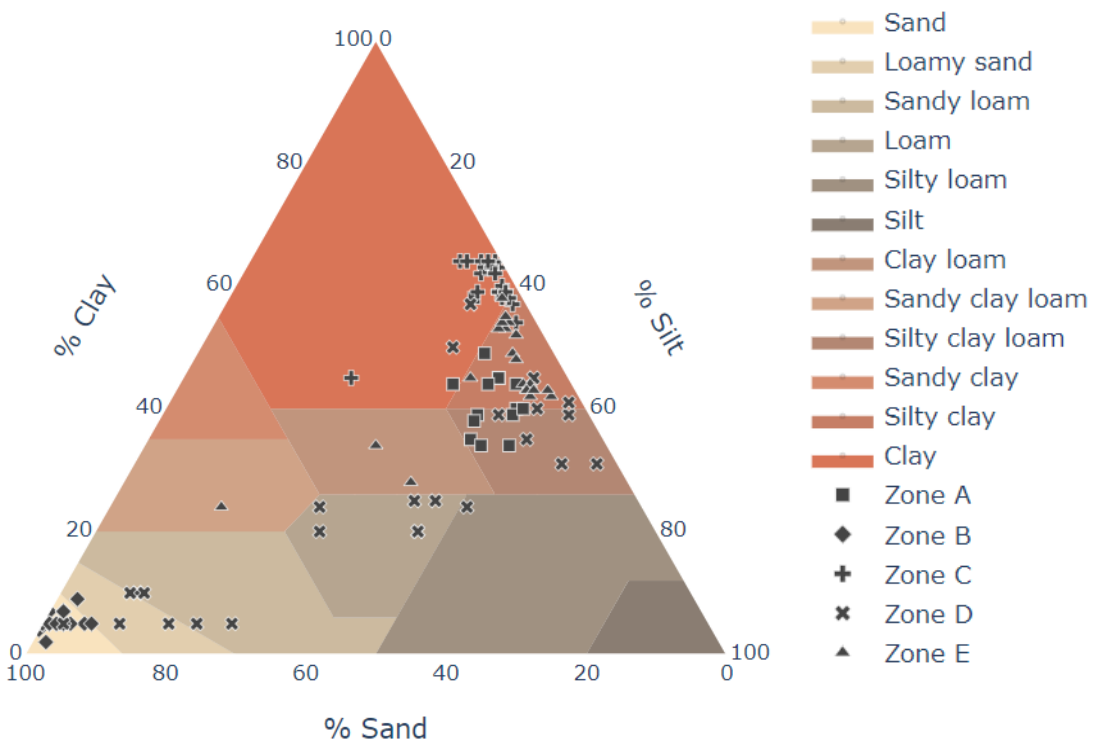

Figure 12. Soil texture classification of parcels

- **pH**

As described in the Theoretical Framework, soils can be classified according to their pH values as acidic for pH values less than 6.5 (strongly acidic for pH less than 5.5), neutral with pH values between 6.5 and 7.5, and alkaline for pH over 7.5. Table 4, indicates the distribution of parcels in these three classifications desegregated by zone.

Zone	Acidic	Neutral	Alkaline
A	0%	5%	95%
B	22%	69%	9%
C	0%	3%	97%
D	0%	22%	78%
E	4%	46%	50%
Total	4%	26%	70%

Table 4. Distribution of pH classification per zone

From Table 4 it can be noticed that most parcels have alkaline soil. Only zone B has most of its parcels in the neutral classification and is the zone with more acid parcels. This is related to the fact that soils in zone B are sandy, and sandy soils tend to be more acidic. On the other hand, the alkalinity might also be caused by the nature of the soils themselves as clay soils are prone to be on the alkaline side of the scale. Soils can be alkaline also because of receiving water that contains highly alkaline substances like calcium or magnesium carbonate.

Acidic and alkaline soils need to be treated before the plantation campaign at the land preparation stage. Acid soils are generally neutralized by adding lime, which naturally contains calcium and

magnesium carbonate that work increasing the soil's pH. On the other hand, alkalinity is usually treated by incorporating organic matter into the soil, given the acidic reaction produced by decomposition that helps reduce pH levels.

- **Correlation between soil properties**

Figure 13 presents a heat map of the linear correlation between the parameters measured on the soil. The stronger the red color of the square, the higher the positive correlation between both parameters, and the stronger the blue color, the higher the negative correlation.

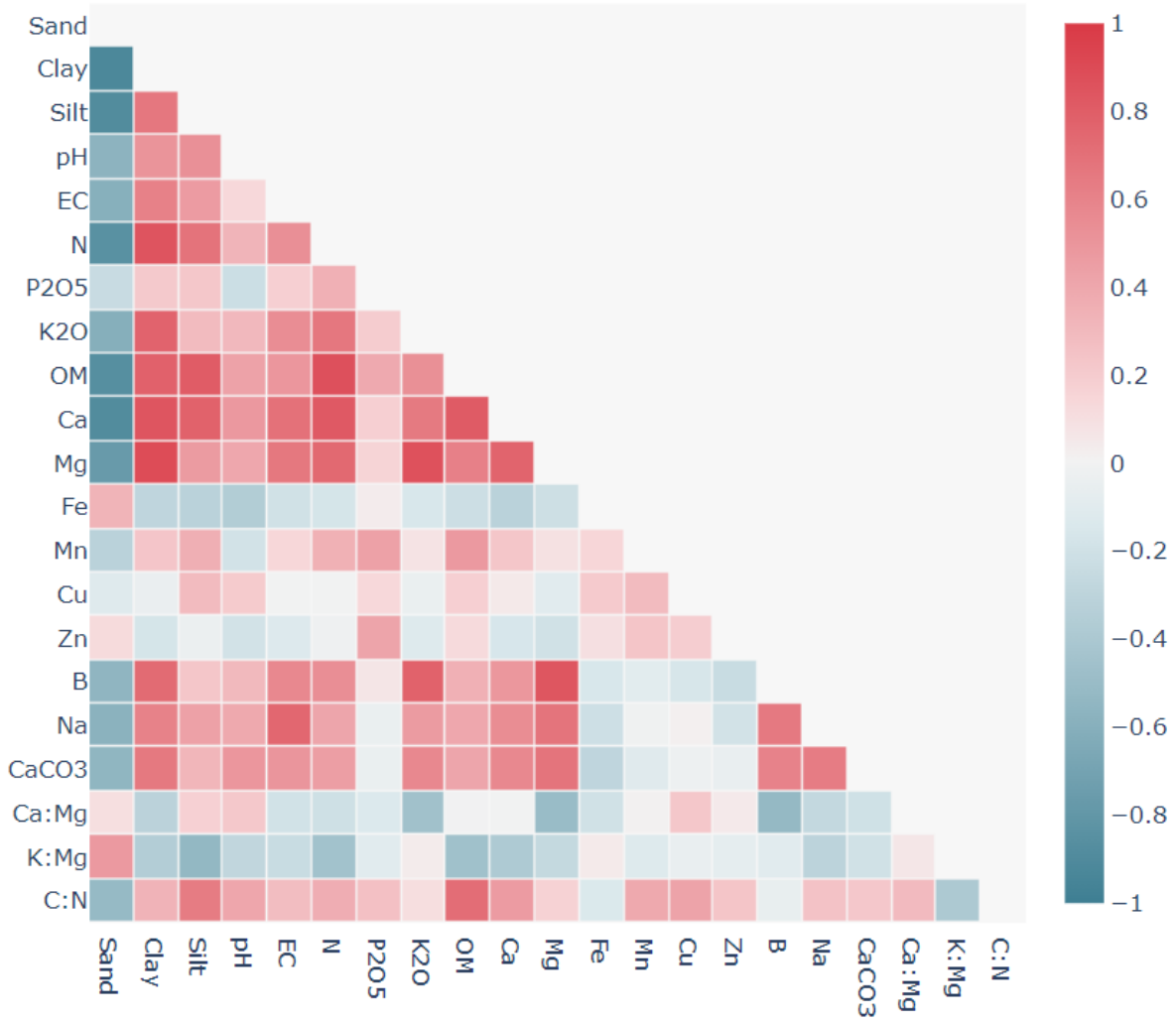

Figure 13. Correlation between soil parameters

From the figure are derived some interesting facts. For instance, the percentage of sand in the soil is negatively related to most of the nutrients. On the contrary, the percentage of clay is positively correlated with a higher presence of nutrients. This matches with the fact that sandy soils tend to be well aerated but do not hold much water and are low in nutrients. By contrast, clay soils, in general, hold more water and are better at supplying nutrients.

3.2.3. WHEATHER INFORMATION

For measuring weather information, the company owns meteorological stations at each one of its five zones. These stations capture climate parameters daily. Although the stations belong to the company, an external supplier operates and captures the information. To integrate the information into our database system, the supplier made available a Web Service with the historical and current data. Later we created a process through Azure Data Factory resource to integrate all the information in our database into a table called *Weather_Parameters* at zone level. Table 5 shows the parameters that were measured at the weather stations.

Parameter	Unit of Measure	Data type
Precipitation (total)	mm	Numeric
Solar radiation (average)	W/m^2	Numeric
Wind speed (average)	Km/h	Numeric
Wind speed (minimum)	Km/h	Numeric
Wind speed (maximum)	Km/h	Numeric
Leaf wetness	minutes	Numeric
Air temperature (average)	ºC	Numeric
Air temperature (minimum)	ºC	Numeric
Air temperature (maximum)	ºC	Numeric
Relative humidity (average)	%	Numeric
Relative humidity (minimum)	%	Numeric
Relative humidity (maximum)	%	Numeric
Dew point (average)	ºC	Numeric
Dew point (minimum)	ºC	Numeric
Dew point (maximum)	ºC	Numeric
Evapotranspiration	mm	Numeric

Table 5. Weather parameters measured at the stations

3.2.4. FERTILIZATION INFORMATION

This information refers to the total amount of fertilizer applied during the plantation campaign on each one of the parcels. As mentioned in the theoretical framework, there are three macronutrients of primary importance for the plants: Nitrogen, Phosphorus, and Potassium. These nutrients are applied to the soil as fertilizers, depending on the soil tests results and if are required by the plant during the campaign. These products are commonly known as NPK fertilizers, because of the chemical symbols of the macronutrients that compose them. The amounts of fertilizers applied are daily registered in a farm management software owned by the company that structured the data in a relational database system. The information was made available through views that later were easily integrated into our system in a table called *Fertilization*, which registered the total dose of fertilizers applied in Kg per one hectare on each parcel. Figure 14 shows the average amount of fertilizers applied by zone in kg/ha. From the chart, it can be inferred that, in general, Nitrogen (N) is the most applied macronutrient and Phosphorus (P) is the less applied. Sandy soils as the ones present in Zone B are more prone to

Potassium (K) deficiency, and it can be confirmed in the chart that this zone received the largest amount of Potassium. Zone C, got the largest rate of Nitrogen and the lowest amount of Phosphorus. In general, clay and loamy soils benefit the most from high Nitrogen supply.

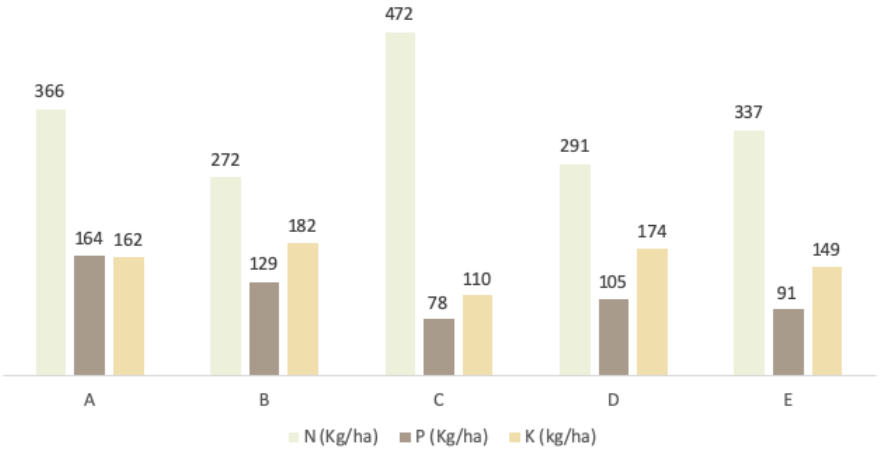

Figure 14. Average fertilizers applied by zone

3.2.5. PRODUCTIVITY INFORMATION

As previously described, when the tomato has reached its maturity stage, it is harvested and send to the factory to be processed. At the factory, only tomatoes that achieve quality standards are accepted. At each delivery, an estimated weight of tomato accepted is calculated. Once deliveries have finished, the productivity of each parcel is calculated as the total weight of accepted tomato in tons over the planted area in hectares (ton/ha). In some cases, more than one tomato variety is planted in the same parcel, and the productivity is calculated at the variety level, considering the weight of the accepted tomato and the planted area for each variety. At our relational database productivity information was integrated into a table called *Productivity* at parcel and tomato variety level. The productivity or yield of tomato crop is our target variable to predict at the modeling stage.

3.3. DATA PREPARATION

After having collected, integrated, and first explored the data sources, this stage had the objective of preparing the final dataset for modeling. For that purpose, each of the data sources was inspected in detail for evaluating its quality and identifying possible erroneous values. This process was under the supervision of business experts who gave feedback when atypical values were found. Fortunately, most of the data sources were in an optimal state. There were only found a couple of anomaly values in the Precipitation field of the *Weather_Parameters* table, due to errors in the data collection at the weather stations. As there were few values, they were manually replaced by searching open information of the nearest public weather stations.

As the *Weather_Parameters* table had daily values of several climate variables, it was necessary to transform and derive new features from the existing ones. It is important to notice that it was considered only the climate during the planting period. This planting period consists of 90 to 120 days

between the plantation and harvesting periods that can be between April and August. Each one of the parcels had different dates of plantation and harvesting so the influence of the climate factors was different. For each climate parameter, it was calculated the mean, standard deviation, minimum and maximum value during the plantation campaign at each parcel. For the Precipitation, it was also calculated the accumulated sum. Additionally, there were derived some features like the *longest dry spell*, the *longest_wet_spell,* and the number of days with temperatures above 27ºC.

After having verified the quality of the data, corrected erroneous values and derived new features, all the data sources were joined into a final dataset for modeling at parcel and tomato variety level. Figure 15 summarizes all data sources that were used for creating the final dataset. We ended up with 440 observations and 61 features from four agricultural campaigns (from 2017 to 2020).

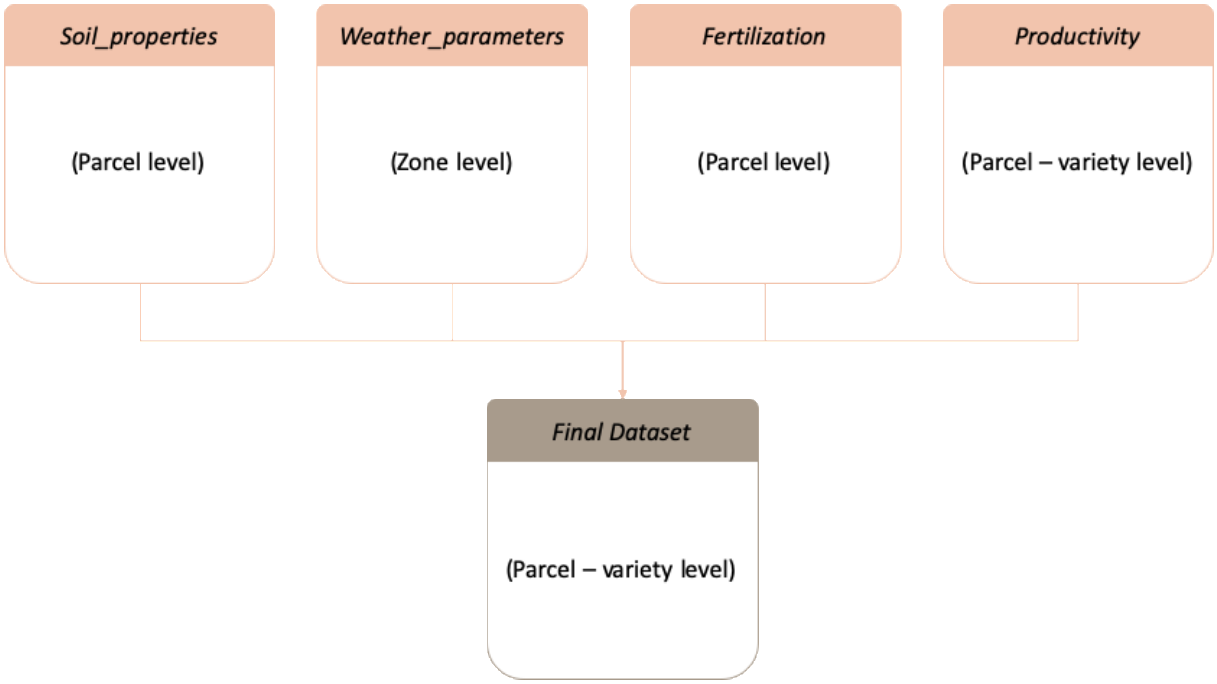

Figure 15. Summary of data sources and final dataset

3.4. MODELING

After having defined and obtained the final dataset, this stage had the objective of training and assessing different ML models in an iterative process. This phase required selecting the modeling techniques, generating the test design, building and evaluating the models. As mentioned before, our target variable was the *Productivity in (ton/ha)* for each pair of parcel and tomato variety. This variable was then set aside from the final data set and stored in a separated vector called *y,* which represented our output variable. The rest of the features were put into a matrix called *X,* serving as our input variables. The modelling process was performed with *Scikit-learn,* the free machine learning library for the *Python* programming language.

3.4.1. MODELS

Considering that our task in this project was to predict a continuous outcome, the approach we should use was Supervised Learning techniques for Regression. Since we were looking for a highly interpretable model and we were short in data samples, our initial options were simple approaches like linear models and decision trees. However, we quickly discarded linear models as our variables were highly correlated and the assumption of no or little multicollinearity was violated. Additionally, the relationship between some features and the target variables was not expected to be linear. This way we opted for tree-based models which are more robust to correlated data and nonlinear relationships. Besides the simple *Decision Trees Regression,* we included into our options ensemble bagging models like *Random Forest Regression* and boosting techniques like *Gradient Boosting Regression.* Finally, we included a final approach that had shown good performance in scenarios of small datasets and capturing nonlinear relationships, the *Support Vector Regression*. These last three models are in general more robust and complex but also more difficult to interpret. Fortunately, there are some techniques for interpreting these "black-box" types of models.

3.4.2. TEST DESIGN

Since we were in a scenario of a small dataset and we needed to make the most of the data in the training stage, we used the k-folds cross-validation approach for the training and validation process. As shown in Figure 16, we split all the data into two groups, a training set (80% of the total data) in which we performed the k-folds cross-validation, and a test set (20% of the total data) that was held out for final evaluation. As explained earlier, in k-folds cross-validation the dataset is split into k equally sized subsets or folds. One of the k-folds will act as the validation set and the remaining folds will be used for training the model. The procedure is repeated for all k possible choices, and the performance score from the k runs is then averaged. Throughout the process of cross-validation, we fit the models and estimate the prediction error for model selection. Additionally, in this stage, we choose a set of optimal hyperparameters through the process of hyperparameter tuning. The test set was used for the assessment of the generalization error of the final chosen model.

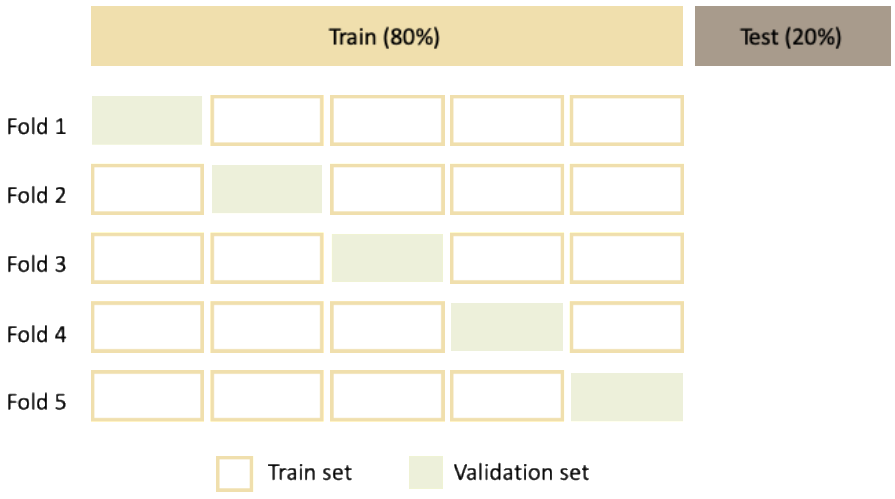

Figure 16. Cross-validation strategy used

3.4.3. HYPERPARAMETER TUNING AND FEATURE SELECTION

Hyperparameter tuning is an important task for the training process that deals with the selection of an optimal set of hyperparameters that optimize the outcome of a model. While the parameters of a model are derived via training, hyperparameters are the ones that define the model architecture and must be tuned before the training stage. In other words, hyperparameters are part of the input to the ML algorithm and parameters are part of the output of the fitting process. A good choice of hyperparameters can optimize the training process and make a model more accurate.

Instead of using methods like the grid or random search for performing the hyperparameter tuning, we opted for a Bayesian optimization approach which works faster and better than the previous ones. In Bayesian optimization, at every iteration, the algorithm detects which regions in the hyperparameter space are more interesting to explore and which are not. This approach typically requires fewer iterations to get to the optimal set of hyperparameter values. The tool that we used for this hyperparameter tuning was *Optuna*, an automatic hyperparameter optimization framework, particularly designed for machine learning and available as a Python library.

The process of feature selection is another important task before training to reduce the computational cost and improve the performance of the model. In our case, we had a considerable number of features and needed to select the best ones. As we didn't have a large volume of data and it was computational feasible, we decided to combine the process of feature selection with hyperparameter tuning. For that purpose, we followed the next methodology:

1. As most of the candidate models were tree-based (except for the SVR), we decided to run an initial random forest regression with the default hyperparameters and with all the features. We then capture the Gini Importance and created a set of features ranked according to their importance.
2. For each model, we performed hyperparameter optimization with Optuna. The sets of features calculated in the previous step were treated as an additional hyperparameter.

Besides the feature selection, we also treated the data scaling as a hyperparameter. The algorithm had to select between no transforming the data, or applying standardization (equation 1) or min-max normalization (equation 2):

$$X_{scaled} = \frac{x - \mu}{\sigma} \qquad (1)$$

$$X_{scaled} = \frac{x - \min(x)}{\max(x) - \min(x)} \qquad (2)$$

2.4.4. PERFORMANCE METRICS FOR MODEL EVALUATION

For evaluating the performance of the model, we selected metrics that were intuitive for interpretation and relatively easy to explain as our model needed to be very transparent for the business. For that purpose, we choose the *Mean Absolute Error – MAE* (equation 3) the *Mean Absolute Percentage Error – MAPE* (equation 4) and the *Root Mean Squared Error – RMSE* (equation 5).

$$MAE = \frac{1}{n}\sum_{i}|y_i - \hat{y}_i| \qquad (3)$$

$$MAPE = \frac{100}{n}\sum_{i}\left|\frac{y_i - \hat{y}_i}{yi}\right| \qquad (4)$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i}(y_i - \hat{y}_i)^2} \qquad (5)$$

The results of the hyperparameter tuning, the feature selection, the model selection, and the assessment and interpretation of the final model are described in the later section of Results.

3.5. EVALUATION

In this stage, our final model was tested and verified to evaluate if it met the business objectives and requirements defined in the first step. As mentioned earlier, the principal objective was to create a model to predict the processing tomato productivity given some factors like the soil properties, the weather conditions, and the fertilization applied. Besides being robust enough for predicting tomato crop yield, this model had to be highly interpretable as the "black-box" types of models were not a good idea in industries like agriculture, where people tend to be a little skeptical about these kinds of approaches. From this perspective, we could meet the requirement of creating a robust model for predicting productivity. Although our final model was not simple and not intrinsically interpretable, we were able to use some interpretability techniques that helped us to explain the prediction of the model to the business. To share the results of this project, we had meetings with people in different positions in the company from growers, to field managers. Even though some people were very skeptical about the benefits of using ML approaches, in general, most of them liked the work and got interested on continue iterating the process.

3.6. DEPLOYMENT

Since it was the first time that a project involving Machine Learning was carried out in the company, the objective was to have a first approach to this kind of technique rather than to have a model to put into production. Although the model had a good performance and its interpretation was well-received, it was still too early to base the business decision on it. The model established a good starting point for the implementation of ML approaches, but it is still necessary to capture more data and to refine and test the model further. On this first iteration, the deployment stage consisted more in sharing the results and familiarizing the people with this kind of approach. The idea is to continue feeding, improving, and assessing the model with the result of each campaign. Hopefully, over time, the model will help farmers make better decisions and optimize productivity.

4. RESULTS AND DISCUSSION

The first step in the training process was to tune the hyperparameters of all the models with Optuna. As mentioned earlier, this step also included the feature and scaling selection. For the feature selection, we trained an initial Random Forest Regressor with all the features and the default hyperparameters. Then we captured the Gini importance for each feature and rank all according to their importance. Figure 17 shows the Top 15 of these ranked features that represented 91% of all Gini Importance. The set of features ranked by Gini Importance were treated as an additional hyperparameter for tuning. The feature in the first place was the total amount of Potassium (K) in kg/ha applied as fertilizer to the soil. There were also five weather parameters and seven soil properties present in this set. The tomato varieties *T10* and *T13* are two of the most widely used varieties in the tomato culture due to their pest and disease resistance and their ability to adapt. The names of the varieties were changed by request of the company.

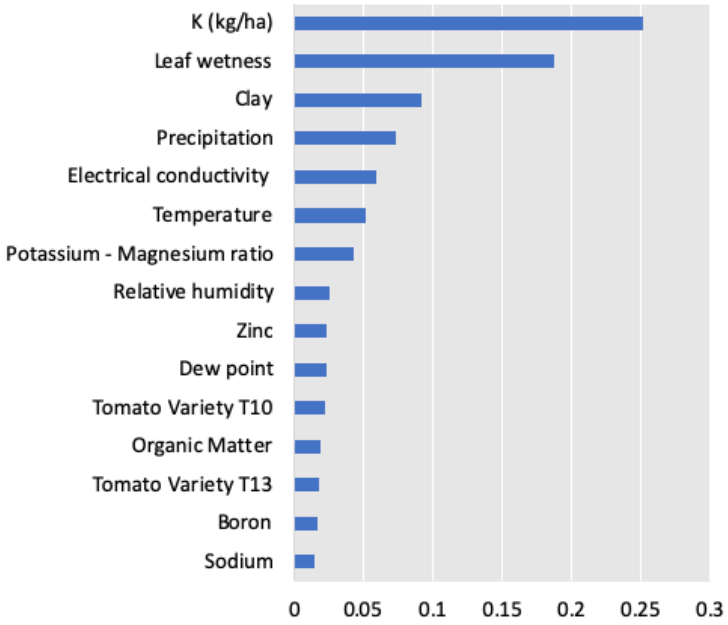

Figure 17. Top 15 features ranked by Gini Importance

Table 6 shows the results of the hyperparameter tuning process for the four models specifying the search space for each hyperparameter, the selected value, the time for optimization and the best score. For each of the models, we run 100 trials optimizing the MAE metric. SVR was the model that obtained the best MAE (11.06) in one of the trials. The second-best score was achieved by the *Gradient Boosting* model (11.86). *Random Forest* and *Gradient Boosting* presented the highest optimization times as expected since ensemble models typically present high time consumption. SVR required the shortest optimization time, although it could be because it was trained only with scaled data. Scaling data is a good practice before training Support Vector Machines otherwise, not scaling the data can make the model less accurate and produce high computational costs.

The Optimization History Plot of the hyperparameter tuning process for the four models, can be found in the Annexes.

Model	Hyperparameter	Search space	Selected value	Time to Optimize	Best MAE
Decision Tree	Criterion	MSE, Friedman MSE, MAE	MSE	4.0 seconds	12.43
	Maximum depth	[1, 20]	4		
	Minimum samples leaf	[2, 20]	4		
	Maximum lead nodes	[2, 20]	15		
	Prunning parameter (alpha)	[0, 0.1] (step=0.001)	0.024		
	Best features	[1, 15]	Top 9		
	Best scaling	No scaling, Std., MinMax	No scaling		
Random Forest	Number of estimators	[10, 100] (step=10)	20	30.99 seconds	12.15
	Criterion	MSE, MAE	MAE		
	Maximum depth	[1, 20]	7		
	Minimum samples split	[2, 20]	4		
	Minimum samples leaf	[1, 10]	12		
	Prunning parameter (alpha)	[0.6, 1.0] (step=0.01)	0.61		
	Maximum samples	[45, 65]	46		
	Best features	[1, 15]	Top 6		
	Best scaling	No scaling, Std., MinMax	No scaling		
Gradient Boosting	Loss	LS, LAD, Huber, Quantile	LAD	55.82 seconds	11.86
	Learning rate	[0.1, 0.5] (step=0.05)	0.15		
	Number of estimators	[10, 200] (step=10)	80		
	Subsample	[0.1, 1.0] (step=0.1)	0.5		
	Criterion	MSE, Friedman MSE, MAE	MSE		
	Best features	[1, 15]	Top 12		
	Best scaling	No scaling, std., MinMax	MinMaxScaler		
Support Vector	Kernel	Linear, Polynomial, RBF	Polynomial	3.11 seconds	11.06
	Degree	[1,5]	4		
	Gamma	Scale, auto	Scale		
	Coeficient	[0.0, 2.0] (step=0.05)	0.90		
	Regularization parameter	[0.0, 2.0] (step=0.05)	1.18		
	Epsilon	[0, 1] (step=0.001)	0.614		
	Best features	[1, 15]	Top 7		
	Best scaling	Std., min-max	StdScaler		

Table 6. Hyperparameter tuning results

After having tuned the hyperparameters, and selected the best values, we then continued with the model selection and model assessment. As described in the theoretical framework section, in model selection, we estimate the performance of different models to choose the best one, and in model assessment, having chosen a final model, we estimate its prediction or generalization error on new data. For the model selection, we performed a k-fold cross-validation process on the train set as presented in Figure 16. Table 7 shows the results of the average prediction error measured in three different metrics for each model.

Model	MAE (ton/ha)	MAPE (%)	RMSE (ton/ha)
Decision Tree	13.3	16.1%	16.2
Random Forest	12.2	14.8%	15.9
Gradient Boosting	10.9	12.6%	13.7
Support Vector	10.6	12.3%	13.6
Stacking	9.4	10.5%	12.4

Table 7. Results of the prediction error for model selection.

The current solution for the company for predicting productivity relies on the intuition of business experts and the average results of past campaigns. Considering that this baseline solution presents an average error of 15 ton/ha from campaign to campaign, most of the models above, show a better result than the current scenario. The *Decision Tree* model presented the worst performance in the three metrics. This result was expected since DT is a simple model whose predictive performance is not competitive with other unsupervised learning techniques. However, its metrics were not so far away from those of the *Random Forest* which was expected to perform better. *Gradient Boosting* and *Support Vector* models had the best and very similar performance.

To have a better performance, we decided to combine the predictions of some of our models into a stacked model. As mentioned earlier, stacking is designed to ensemble a group of strong learners, and such models tend to perform as well or better than any of the individual learners in the staking. For deciding which models to combine, which features to include, and the scaling method to apply, we used again Optuna to optimize the choice. The selected strategy for the stacking was to combine our two best models *Gradient Boosting* and *Support Vector* as base learners and to have a Ridge Regression as our final model or super learner. Figure 18 shows the structure of the stacking solution.

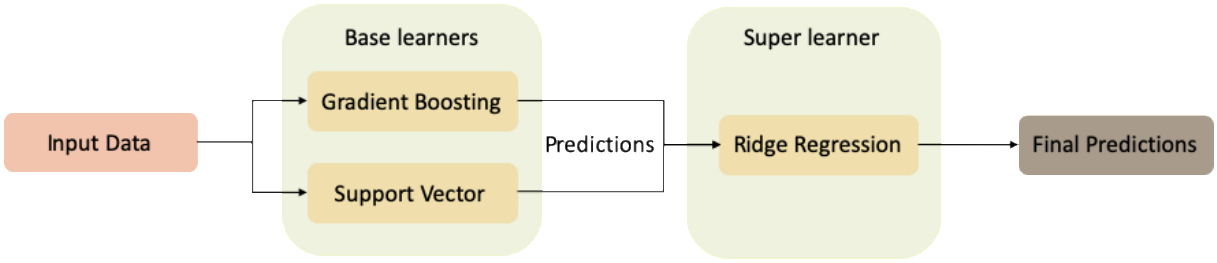

Figure 18. Structure of the stacking solution

The Ridge Regression is a method for shrinking the coefficients of linear regression by imposing a penalty on their size. It works by minimizing a penalized residual sum of squares (Hastie, Tibshirani, & Friedman, 2017). The Ridge Regression fits well as the final model in staking because it has good performance in scenarios where the variables are highly correlated. In our case, the variables are the predictions of the base models which present indeed a high correlation. Besides the base and final models for the stacking, the optimization process selected the Top 12 features and the MinMaxScaler method for data scaling. Table 7 shows the result of the prediction error for the stacked model. For the three metrics, the performance was better than any previous model. For this reason, we selected the stacked model as our final one.

After having selected the final model, we moved to the stage of model assessment, where we estimated the prediction or generalization error of the final chosen model on new data. For that purpose, we used the test set that was not used in either the parameter tuning or the training and represented 20% of the total data as shown in Figure 16.

Table 8 presents the generalization error of the final chosen model on new data. Again, we showed the three metrics of MAE, MAPE, and RMSE that have high relevance due to their interpretability. Regarding the MAE metric, it implies that on average the distance between the prediction and the real value of productivity was 9.02 ton/ha. As to MAPE, it tells us that on average, the prediction of the productivity is off by 9.5%. The RMSE is also a measure of the average distance between the prediction and the actual value. It is typically higher than the MAE due to the squared distances that penalize more the highest differences.

Model	MAE (ton/ha)	MAPE (%)	RMSE (ton/ha)
Final model	9.02	9.5%	13.5

Table 8. Generalization error of the final chosen model

In conclusion, our model could predict tomato crop yield with an approximate error of 9 ton/ha considering that the average productivity is 94.5 ton/ha. This is a very good result bearing in mind that we were in a small data scenario and that it was the first approach we performed. Additionally, it presents a significant improvement regarding the current solution based on the business expert intuition and the average results of past campaigns that present an error of 15 ton/ha from campaign to campaign.

Another important stage after having chosen and assessing the final model was to interpret it. For that purpose, we first evaluated the importance each feature had on our model. Figure 19 presents a SHAP summary plot that combines feature importance with feature effects. Each point on the plot is a Shapley value for a variable and an observation. The features are ordered according to their importance and the color of the point represents the value of the feature from low to high. The feature with the highest importance in our model is Temperature and the one with the lowest importance is Organic Matter. High values of Temperature, Electrical conductivity, Dew Point, and Potassium-Magnesium Ratio harm productivity. On the contrary, high levels of Leaf wetness, Zinc, and Relative humidity seem to have a positive impact on the crop yield.

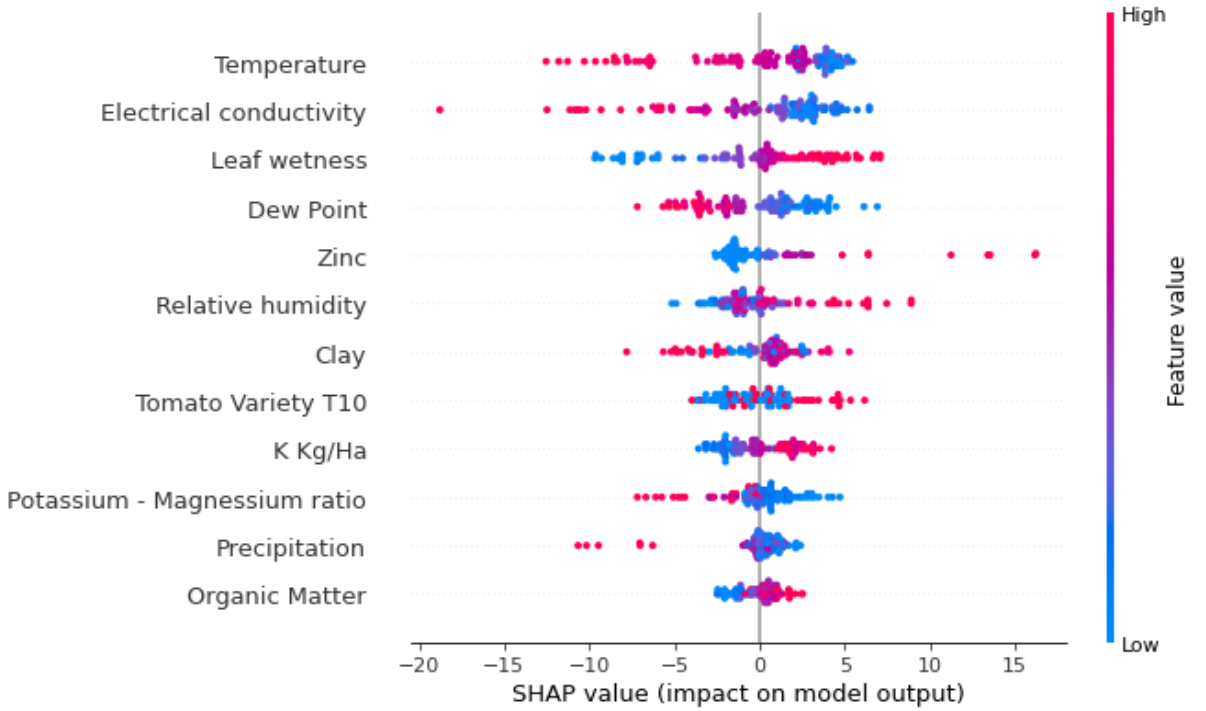

Figure 19. SHAP summary plot

Another valuable plot for interpretation is the *Accumulated Local Effects* (ALE) plot, which describes how features influence the outcome of a model on average. Figure 20 shows the ALE plot for each feature, since the variables were scaled with the *MinMaxScaler* the interpretation was not direct and we had to transform back some values to the original ones. Next, we describe the effect that each feature has on the average prediction of productivity.

- *Potassium (K) kg/ha*: values of the fertilizer between 118kg/ha and 191 kg/ha have a positive effect on the average prediction of productivity. Applying less than 118 kg/ha of Potassium fertilizer has a negative impact and values greater than 191 kg/ha seems to have no effect.
- *Leaf wetness*: the effect seems to oscillate but in general, the fact that the leaves are exposed to liquid moisture on average less than 336 minutes (6.7 hours) a day seems to have a little positive effect on productivity average prediction. It can be related to the effect that high moisture has on the development of fungus and diseases.
- *Clay:* values greater than 39% of clay have a positive effect on the average prediction. As mentioned before, the percentage of clay is positively correlated with a higher presence of nutrients on the soils.
- *Precipitation:* only very low values of precipitation have a positive effect on the average prediction of productivity. Values greater than 0.59 mm have a negative effect.
- *Electrical conductivity:* this soil parameter has a strong effect on the prediction. Values of electrical conductivity greater than 617 (mS/m) have a negative impact on the average crop yield prediction. It is also a measure of the salinity of the soil, and it's an important indicator of soil health since the excess of salt may cause a variety of problems for plants.
- *Temperature:* this weather parameter also has a strong negative effect on the average prediction of productivity after values of 20.5 ºC.

- *Potassium-magnesium radio: it has a negative effect on the average yield prediction for values greater than 0.22.*
- *Relative humidity: for values greater than 71% there is a positive effect on the average prediction.*
- *Dew point:* it has a similar effect to *Temperature. Values greater than 14.3ºC have a negative impact on the average prediction.*
- *Zinc:* concentrations of this nutrient larger than 18.74 mg/Kg have a positive effect on the average productivity prediction.
- *Tomato Variety T10:* the fact that the tomato variety is or not the *T10* seems to not affect the predictions much.
- *Organic Matter:* it also seems that this soil parameter does not affect the average prediction so much. However, it seems that values of Organic Matter greater than 1.96% have a negative effect.

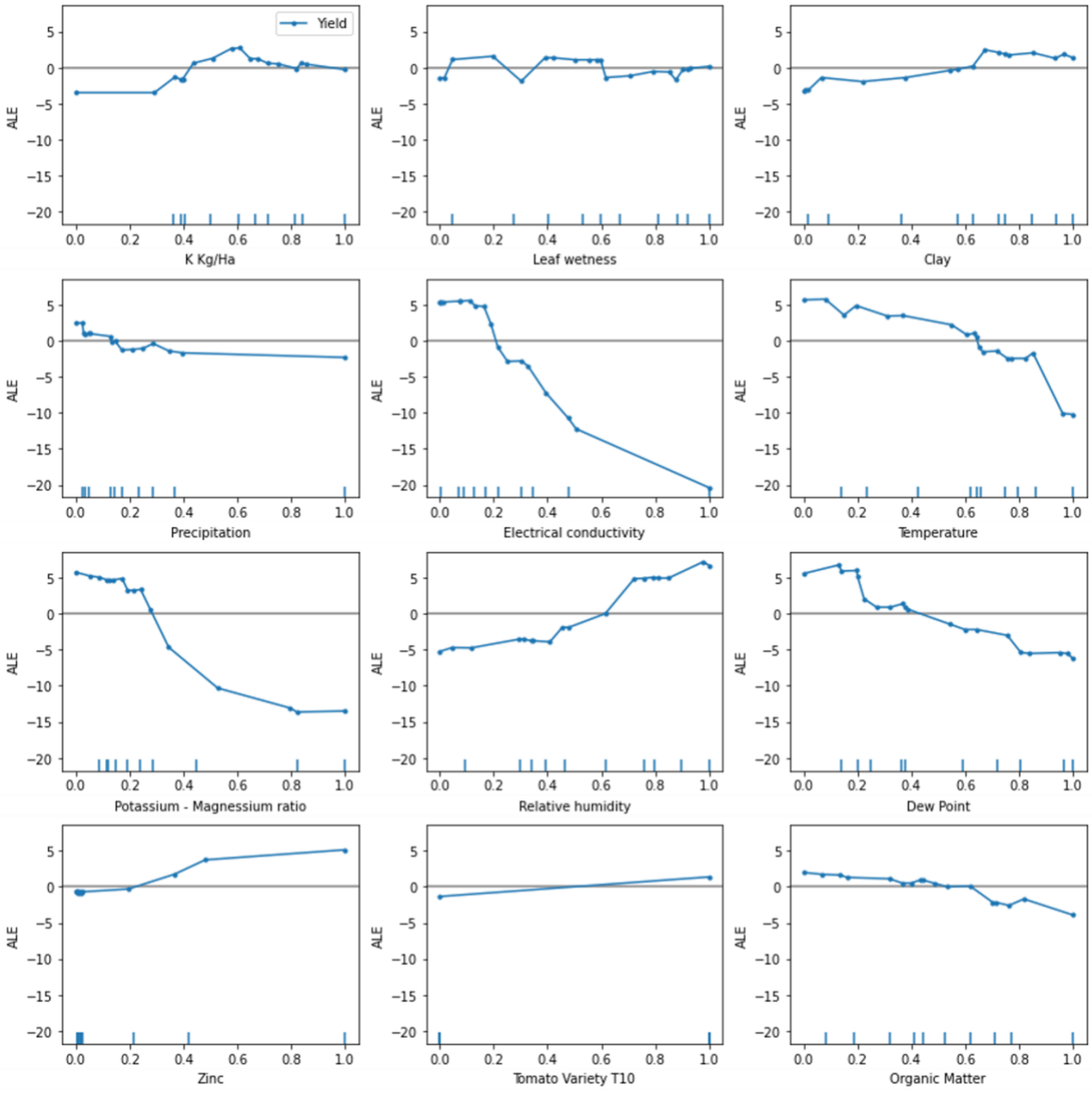

Figure 20. Accumulated Local Effects plot

In general, the interpretation of the model made sense, and most of the features showed the expected effect. For the company, it was very important that the decisions taken by the model were transparent and reflected reality. Even though several features included in the model are weather parameters that can't be controlled by the growers, the model can be a relevant tool for them when evaluating scenarios or estimating the effects that some parameters can have on productivity. As more data is collected the model can be refined, and the predictions will become more accurate as well as the interpretation.

5. CONCLUSIONS

This project was developed as a Data Science internship at a Portuguese company dedicated to the cultivation and processing of tomatoes. The objective was to apply ML approaches, to predict processing tomato crop yield given some attributes such as soil properties, weather conditions, and fertilizer application. Besides being robust enough for predicting tomato productivity, the model needed to be interpretable and transparent for the business. This was the first time that a project involving Machine Learning was carried out in the company, and it was the first step in its path towards Precision Agriculture, a strategy that helps farmers make more intelligent decisions based on information and technology.

For that purpose, we created a Data Science workflow based on the CRISP-DM methodology. This is a process model with six phases that provides a framework for designing, creating, testing, and deploying machine learning solutions. The first half of the project was dedicated to understanding the objectives from the business perspective, collecting and analyzing data sets from several sources, and cleaning, transforming, and preparing the data for modeling. The second part of the project consisted in applying and assessing various ML regression models for predicting productivity.

The models assessed were *Decision Tree Regression,* ensemble bagging models like *Random Forest Regression,* and boosting techniques like *Gradient Boosting Regression,* and finally, an approach that had shown good performance in scenarios of small datasets and capturing nonlinear relationships, the *Support Vector Regression*. Before the training process, we performed the Hyperparameter tuning and feature selection with *Optuna,* a Bayesian optimization framework for machine learning. For the training process, we selected a k-folds cross-validation strategy that better worked in our small data set scenario.

As expected, the *Decision Tree* model presented the worst performance in the MAE, MAPE, and RMSE metrics. *Gradient Boosting* and *Support Vector* models had the best performance. For improving the predictive power, we combined the predictions of our two best models into a stacked approach with a Ridge Regression as the final model. For the three metrics, the performance was better than any previous approaches. For this reason, we selected the stacked model as our final one. Finally, we estimated the generalization error of the final chosen model on new data. For that purpose, we used the test set that was not used in either the parameter tuning or the training and represented 20% of the total data. The result for the generalization error was 9.02 ton/ha for the MAE metric, 9.5% for the MAPE, and 13.5 ton/ha for the RMSE. This means that our model could predict tomato crop yield with an approximate error of 9 ton/ha.

Even though our final model was complex and not intrinsically interpretable, we were able to apply model-agnostic interpretation methods especially helpful to explain these kinds of “black-box” approaches. Specifically, we used the SHAP summary plot to better understand the feature importance and feature effects, and the *Accumulated Local Effects* (ALE) plot, to explain how features influence the outcome of the model on average.

In general, the objectives of the project were accomplished and the company was satisfied with the result of the model and its interpretation. They are willing to continue iterating the process and feeding the model with more data for the next campaigns. Hopefully, over time, the model will help growers make better decisions and optimize productivity.

6. LIMITATIONS AND RECOMMENDATIONS FOR FUTURE WORKS

The main limitation of our project was the small amount of data available. In agriculture, the data is very limited since it works by campaigns that happen once a year. It means that for having a considerable quantity of data, we need to wait several years. In our case, we had access to the data of four plantation campaigns, which allowed us to create a good enough model considering the amount of data, but that needs to be improved year by year.

For future works, the idea is to continue feeding, improving, and assessing the model with the data collected in each campaign. The main goal is that the growers can make better decisions and optimize productivity based on AI techniques. For that purpose, besides capturing more data and improving the model, it is important to familiarize the people with the ML techniques, since many of them are very skeptical about these kinds of approaches.

7. BIBLIOGRAPHY

American Metereological Society. (2020). *Glossary of Meteorology*. Retrieved from American Metereological Society: https://glossary.ametsoc.org/wiki/Welcome

Awad, M., & Khanna, R. (2015). Support Vector Regression. In *Efficient Learning Machines.* Springer Nature.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning* . Springer.

Boehmke, B., & Greenwell, B. (2020). *Hands-On Machine Learning with R.* Taylor & Francis Group.

Burkov, A. (2019). *The Hundred-Page Machine Learning Book.*

Data Science Project Management. (n.d.). *What is CRISP DM?* Retrieved from Data Science Project Management: https://www.datascience-pm.com/crisp-dm-2/

Encyclopedia Britannica. (2021). *Encyclopedia Britannica*. Retrieved from Encyclopedia Britannica: https://www.britannica.com/

Fischetti, T. (2018). *Data Analysis with R - Second Edition.* Packt.

Food and Agriculture Organization of the United Nations. (1984). *Fertilizer and plant nutrition guide*. Retrieved from Food and Agriculture Organization of the United Nations: http://www.fao.org/3/aq355e/aq355e.pdf

Food and Agriculture Organization of the United Nations. (2009, 10 12). *How to feed the world in 2050*. Retrieved from Food and Agriculture Organization of the United Nations: http://www.fao.org/fileadmin/templates/wsfs/docs/expert_paper/How_to_Feed_the_World_in_2050.pdf

Forbes. (2021, 01 07). *Forbes Technology Council*. Retrieved from Forbes: https://www.forbes.com/sites/forbestechcouncil/2021/01/07/artificial-intelligence-and-precision-farming-the-dawn-of-the-next-agricultural-revolution/?sh=6bfa74751dbe

Hastie, T., Tibshirani, R., & Friedman, J. (2017). *The Elements of Statistical Learning.* Springer.

IBM Cloud Education. (2020, 07 15). *IBM Cloud Education*. Retrieved from IBM: https://www.ibm.com/cloud/learn/education

James, G., Witten, D., Tibshirani, R., & Hastie, T. (2013). *An Introduction to Statistical Learning with Applications in R.*

Kumar Srivastava, A., Safaei, N., Khaki, S., Lopez, G., Zeng , W., Ewert, F., . . . Rahimi, J. (2021, 05 04). Comparison of Machine Learning Methods for Predicting Winter Wheat Yield in Germany.

Larose, D., & Larose, C. (2014). *Discovering Knowledge in Data: An Introduction to Data Mining, 2nd Edition.* Wiley.

Meroni, M., Waldner, F., Seguinia, L., Kerdiles, H., & Rembold, F. (2021, 04 27). Yield forecasting with machine learning and small data: what gains for grains? *Agricultural and Forest Meteorology*.

Molnar, C. (2021). *Interpretable Machine Learning. A Guide for Making Black Box Models Explainable.*

Oracle. (2021). *Accumulated Local Effects*. Retrieved from Oracle Accelerated Data Science: https://docs.oracle.com/en-us/iaas/tools/ads-sdk/latest/user_guide/mlx/accumulated_local_effects.html

Queensland Government. (2016). *Soil Properties*. Retrieved from Queensland Government: https://www.qld.gov.au/environment/land/management/soil/soil-properties

Shahhosseini, M., Hu, G., & Archontoulis, S. (2020). Forecasting Corn Yield With Machine Learning Ensembles. *Frontiers in Plant Science*.

Srinivasan, A. (2006). *Handbook of Precision Agriculture Principles and Applications.*

Stirrup, J., & Oliva Ramos, R. (2017). *Advanced Analytics with R and Tableau.*

The Economist. (2016, 06 09). *The future of Agriculture*. Retrieved from The Economist: https://www.economist.com/technology-quarterly/2016-06-09/factory-fresh

United States Department of Agriculture. (n.d.). *United States Department of Agriculture*. Retrieved from Soil Quality Kit: https://www.nrcs.usda.gov

University of Georgia. (2017, 01 30). *Commercial Tomato Production Handbook*. Retrieved from University of Georgia Extension: https://extension.uga.edu/publications/detail.html?number=B1312&title=Commercial%20Tomato%20Production%20Handbook#:~:text=Soil%20Requirements%20and%20Site%20Preparation,that%20tend%20to%20stay%20wet.

Yara. (2021). *Crop Nutrition Tomato*. Retrieved from Yara United States: https://www.yara.us/crop-nutrition/tomato/managing-tomato-taste/

8. ANNEXES

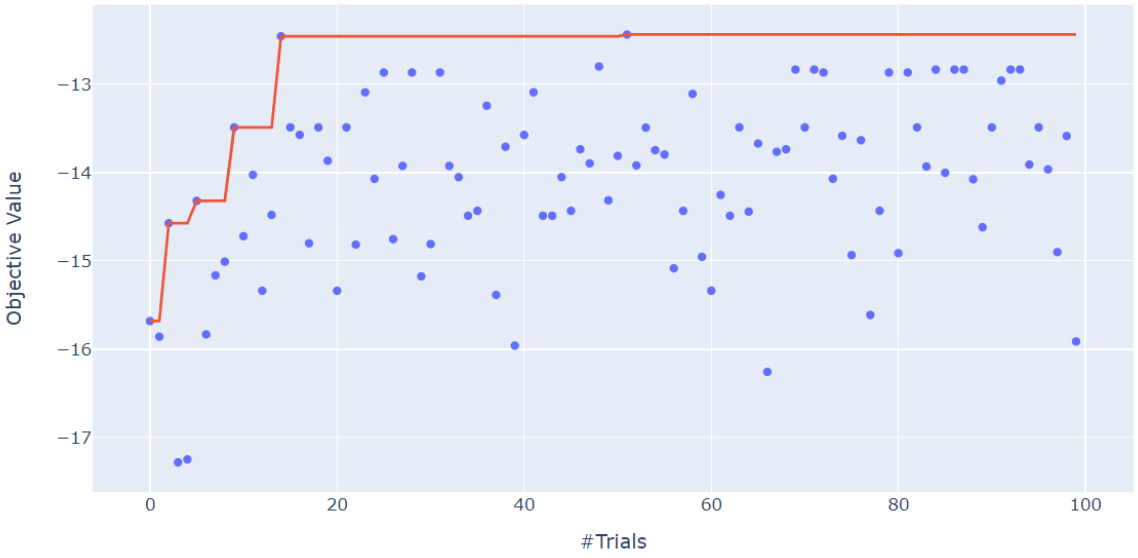

Figure 21. Optimization history plot Decision Tree

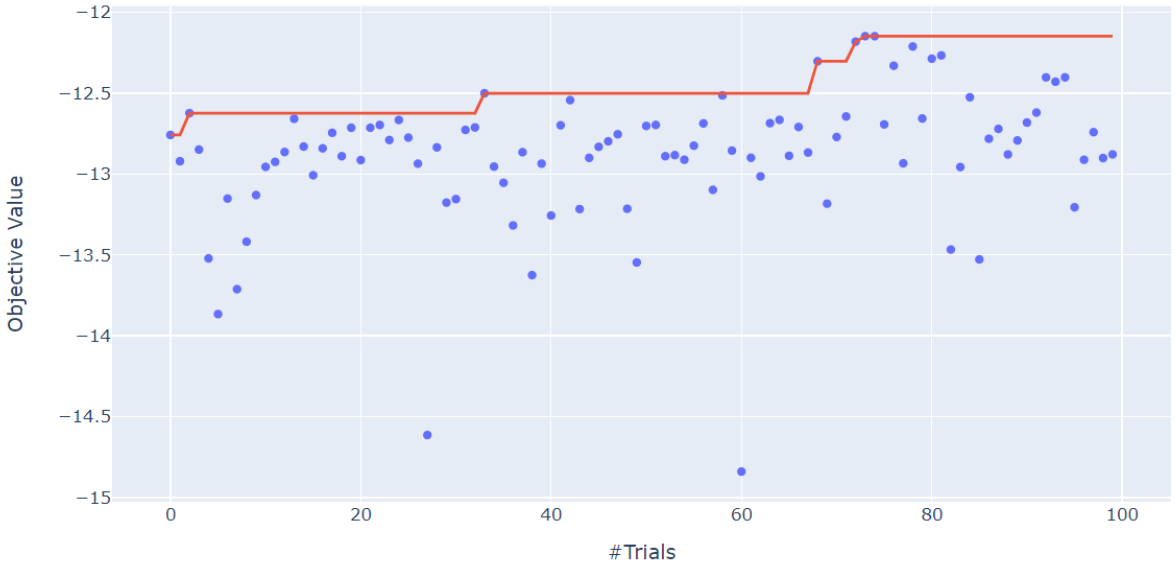

Figure 22. Optimization history plot Random Forest

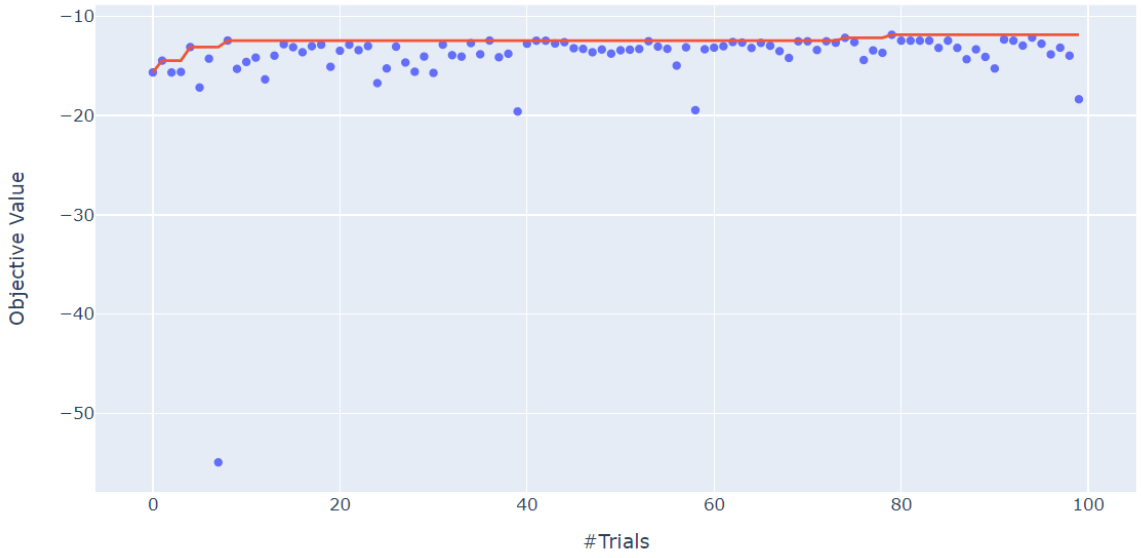

Figure 23. Optimization history plot Gradient Boosting

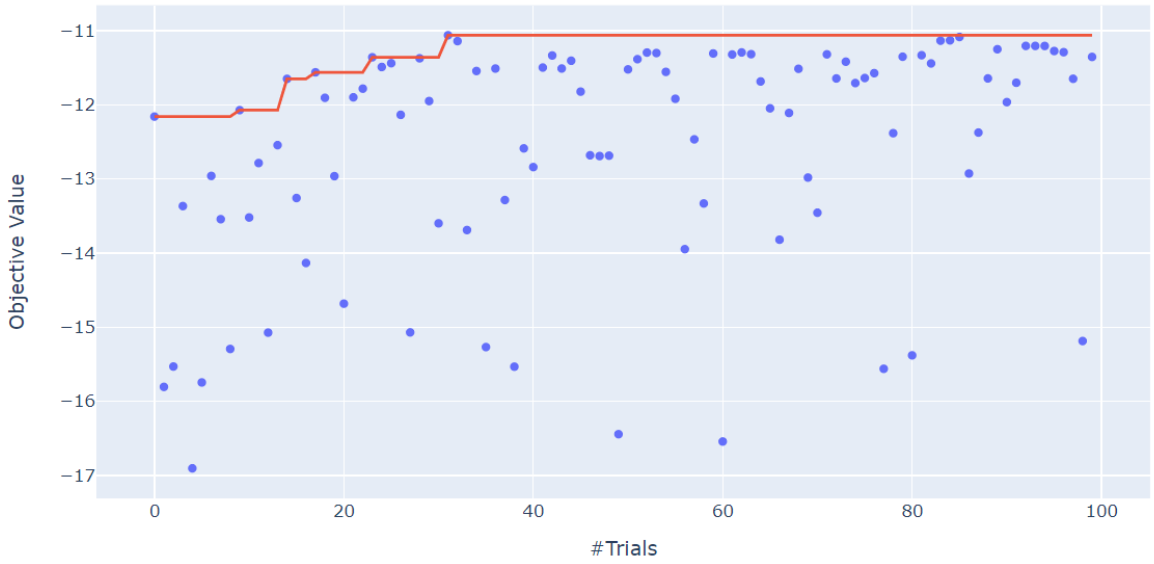

Figure 24. Optimization history plot Support Vector