D4.3 Decision Support, Benchmarking and Performance Indicator Monitoring Tools – Release 2

Dissemination level: Public
Submission date: 31st May 2021

Contents

This document is issued within the frame and for the purpose of the DEMETER project. This project has received funding from the European Union's Horizon2020 research and innovation programme under Grant Agreement No. 857202. The opinions expressed and arguments employed herein do not necessarily reflect the official views of the European Commission.

1 Executive Summary

DEMETER aims to lead the Digital Transformation of the European Agrifood sector based on the rapid adoption of advanced technologies, such as Internet of Things, Artificial Intelligence, Big Data, Decision Support, Benchmarking, Earth Observation, etc., in order to increase performance in multiple aspects of farming operations, as well as to assure the viability and sustainability of the sector in the long term. It aims to put these digital technologies at the service of farmers using a human-in-the-loop approach that constantly focuses on mixing human knowledge and expertise with digital information. DEMETER focuses on interoperability as the main digital enabler, extending the coverage of interoperability across data, platforms, services, applications, and online intelligence, as well as human knowledge, and the implementation of interoperability by connecting farmers and advisors with providers of ICT solutions and machinery.

To enable the achievement of those objectives, and to promote the targeted technological, business, adoption and socio-economic impacts, DEMETER is designing and developing targeted decision support systems to enable the delivery of tailored advisory services to the agricultural sector. This DSS combines data analytics from Work Package 2 with AI-based expert system, machine learning and benchmarking techniques to provide precision decision support to the users. This deliverable provides an update on the progresses performed to two basic activities of the DSS: AI-based analytic functions on one side and Benchmarking techniques and performance monitoring tools on the other side. Both activities serve as core building blocks of the DEMETER DSS for addressing the needs of the pilots. The AI building block services provide the intelligence to several DSS components which cover different situations (e.g., crop identification, irrigation needs or animal care). These should help the farmer with understanding the current state and, eventually, predicting future states. For Benchmarking, a minimum set of indicators covering pilots' activities has been established. Constraints of data availability at the farm level have been addressed. Based on these indicators, the Benchmark tools will provide feedback to the pilots and farmers about the agronomic, environmental, and economic sustainability of the practices adopted and of the technologies delivered within DEMETER. The benchmark system will allow the comparison of farms through three different components, which can be used by pilots: i) generic economic farm comparison (exploiting the data of FADN), ii) neighbouring benchmarking (a group of farms with similar environmental conditions and type of farming), iii) technology benchmarking (to evaluate the impact of a specific technology).

2 Acronyms and Abbreviations

ACS	Access Control System
AI	Artificial Intelligence
AIM	Agricultural Information Model
ANN	Artificial Neural Network
API	Application Programming Interface
BBCH	Biologische Bundesanstalt, Bundessortenamt und Chemische Industrie
BSE	Brokerage Service Environment
CAP	Common Agricultural Policy
DEH	DEMETER Enabler Hub
DOI	Digital Object Identifier
DOY	Day of the Year
DSS	Decision Support System
ET_0	Reference Evapotranspiration
FADN	Farm Accountancy Data Network
FAPAR	Fraction of Absorbed Photosynthetically Active Radiation
FMIS	Farming Management Information Systems
FoI	Feature of Interest
FTIR	Fourier-transform infrared spectroscopy
ICT	Information and Communications Technology
GDD	Growing Degree Day
ISSN	International Standard Serial Number
JSON	JavaScript Object Notation
JSON-LD	JavaScript Object Notation for Linked Data
KPI	Key performance indicators
LSTM	Long-Short Term Memory
LSU	Standard Livestock Unit
LU	Livestock Unit
ML	Machine Learning
MLP	Multi-Layer Perceptrons
MSE	Mean Squared Error
NDVI	Normalised Difference Vegetation Index
NN	Neural Networks
NUTS	Nomenclature of Territorial Units for Statistics
OPTRAM	OPtical TRApezoid Model
REST	Representational State Transfer
RF	Random Forest
RMSE	Root Mean Square Error
SKOS	Simple Knowledge Organization System Primer
SMART	Specific, Measurable, Achievable, Relevant, Time-bound
SNN	simulated neural networks
SOSA	Sensor Observation Sampling Actuator
STR	SWIR Transformed Reflectance
TSM	Time Series Models
UAA	Utilised Agricultural Area
URL	Uniform Resource Locator

3 List of Authors

Company	Author
ICE	Oscar Garcia (Editor)
	John Beattie (Editor)
AGRICOLUS	Diego Guidotti (Editor)
	Susanna Marchi
UMU	Manuel Mora
ATOS	Sergio Salmerón
VITO	Bart Beusen
VICOM	Izar Azpiroz
MIMIRO	Harald Volden
ENG	Antonio Caruso
DNET	Nenad Gligoric

4 Document History

Version	Date	Change editors	Changes
0.1	19/01/2021	Oscar Garcia (ICE)	First draft with TOC
0.2	21/01/2021	John Beattie (ICE) Oscar Garcia (ICE) Diego Guidotti (AGR)	Comments to first draft + final TOC
0.3	22/03/2021	Bart Beusen (VITO) Izar Azpiroz (VICOM) Manuel Mora (UMU) Sergio Salmerón (ATOS) Diego Guidotti (AGR) Harald Volden (MIMIRO) Antonio Caruso (ENG) Nenad Gligoric (DNET)	Input to section 6
0.4	22/03/2021	Diego Guidotti (AGR)	Input to section 7
0.5	09/04/2021	Oscar Garcia (ICE)	Input to section 5 and 8
0.6	23/04/2021	Oscar Garcia (ICE)	Final check before peer review
0.7	07/05/2021	Filipe Neves (INESC Porto) Martin Klopfer (OGC)	Peer review
0.8	14/05/2021	Oscar Garcia (ICE)	Updates to peer review
0.9	21/05/2021	Diego Esteban (ATOS)	Final review
1.0	31/05/2021	Oscar Garcia (ICE)	Final version to be submitted

5 Introduction

This deliverable summarises the progresses made since the initial version D4.1 was released, reporting on WP4 tasks related to AI-Decision-making, Benchmarking and Performance Indicator Monitoring Tools.

Aims of these tasks were:

- to deliver building blocks of decision-making systems that serve the specific needs of the DEMETER pilots; these building blocks use AI-based expert systems, machine learning and benchmarking techniques to provide tailored advice in specific agro-management environments.
- to integrate the data, services and platform adopted in the pilots to support the creation of a benchmarking system that can be used at farm level to evaluate the productivity and the sustainability of the practices adopted and to test and evaluate the efficacy of the developed digital solution in reducing costs, improve the production and support the long-term sustainability.

The document is structured as follows:

Section 6 provides a description of the progresses carried out in the areas for artificial intelligence technologies, thus covering the advances and progress made since DEMETER D4.1 (Decision Support, Benchmarking and Performance Indicator Monitoring Tools – Release 1) was submitted. This content is related to the AI-based Decision-Making models being implemented and integrated within the different DEMETER DSS components.

Section 7 describes the implementation of the benchmarking system of DEMETER. Since the DEMETER benchmarking system aims to provide end-users with tools to evaluate the productivity and the sustainability of the practices adopted, as well as the efficacy of the developed digital solutions, these components will enable the comparison for individual and peer to peer learning, linked to the impact of operational processes brought by DEMETER. This section reports the advances and progresses made during this period regarding benchmarking indicators to be implemented and integrated within the DEMETER Decision Support System (DSS) Benchmarking components.

Finally, Annex A briefly describes some of the ML and AI libraries used for the different modules developed for the DSS components. The descriptions are preceded by definitions of the types of datasets and the metrics used to evaluate the performance of the models used in deployment and in training (where appropriate).

6 Progress on AI Decision Making models

Starting from the initial algorithms proposed in D4.1, this section covers the **advances and progress** made during this period to the AI-based Decision-Making models to be implemented and integrated within the different DEMETER DSS components.

As a generic statement, to access a referenced piece of code that has been uploaded to DEMETER GitLab, permission needs to be requested and granted by DEMETER in a case-by-case basis.

6.1 AI model for Component 4.A.1: Plant Yield Estimation

The yield prediction model currently implemented and available on GitLab[1], uses a smoothed Sentinel-2 timeseries (daily NDVI values) to predict potato yields on field level, with data from AVR harvesting machines as ground truth data to train the model. At this moment, only the prediction step is fully implemented, using a pre-trained neural network to do the prediction for potato fields.

Data-driven or non-parametric models have the ability to model very complex systems, but they require a large amount of training data. Complicated models with many features compared to the training examples are likely to overfit. The application of ML methods combined with sensing technologies, conducted on small areas with small samples of data, leads to a low ability to generalise the learned parameters to areas with different characteristics. The availability of large datasets from diverse sources is necessary to achieve better generalisation. This component will be used in pilot 3.4 where ground truth data for potato yields will be available through yield sensors on AVR harvesting machines. Data from this pilot will be used to train a prediction model for potatoes, although the same workflow can also be used to train a model for other crops.

The workflow is organised in the following steps:

1. Generate the crop growth curve:
    - The input is an AIM description of the field (crop type, polygon, planting date) for which the yield prediction should be run.
    - A Crop growth curve can be retrieved from a Sentinel-2 timeseries service, with data for cloudy days interpolated using a smoothing algorithm (e.g., Whitaker smoothing [1]).
    - Data is generated from starting date to the date on which the prediction is done (2-3 weeks before harvest).

2. Predict yield:
    - Input is a smoothed timeseries of daily NDVI values.
    - The timeseries is converted to a fixed length array by padding with zeros at start and end of the array.
    - Timeseries values are NDVI values, so the values are already between 0 and 1.
    - The fixed length timeseries array is fed as input into the regression model, i.e., an ensemble (bagging) of 10 multi-layer perceptrons (MLP, neural network of 3 layers, see Annex A.5 for a brief introduction to MLP), each with 1 output neuron whose value represents a scaled-down value for the predicted yield.
    - The output of the regression model is then unscaled using a fixed scaler valid only for storage potatoes.

[1] https://gitlab.com/demeterproject/wp4/decisionsupport/4.a.cropgrowthstatusyield/4.a.1-plantyieldestimation/vito_yield_prediction

3. Flask webservice:
    - The yield prediction service is embedded in a Flask application that can be run on localhost, and which accepts an NDVI timeseries in AIM JSON-LD format as payload.

The generation of the smoothed NDVI timeseries is not yet uploaded to GitLab. In the current implementation, the timeseries is generated by calling the CropSAR [2] service of VITO, which uses data fusion of Sentinel-1 and Sentinel-2 to produce NDVI values for cloudy days. Since this is a restricted service, access can only be granted when authentication protocols are in place. As an alternative, the standard Terrascope timeseries service[2,3] can be used, where NDVI values on cloudy days are interpolated using a standard smoother or using e.g. Whitaker smoothing. The generation of NDVI timeseries using the standard timeseries service will be implemented as part of this component.

The implemented model only uses the NDVI timeseries as input, which serves as an indicator of growing conditions. Any deviation in the NDVI curve marks the deficiency of some parameter and thus will have an effect on the crop yield. To be able to predict the yield several weeks before harvest, meteorological conditions (air temperature, rainfall) will also need to be considered. Therefore, we aim to integrate meteorological input directly as an extra layer in the input array, to simulate crop yield under varying predicted meteorological scenarios for the last weeks before harvest.

6.2 AI model for Component 4.A.2: Plant Phenology Estimation

The olive phenology state (winter buds, flowering…) can be represented numerically with the BBCH scale [3], which is used to identify the phenological development stages of plants. The web service of this component, currently available in the GitLab repository[4], provides a BBCH value for a given Day Of Year (DOY) symbolising a concrete natural olive phenology state. This service can be launched locally following the steps in GitLab to mount the dockerised system. The prediction is provided thanks to a previously trained random forest (see Annex A.2 for a brief introduction to Random Forest) model and the connection to the Copernicus API.

The workflow of this service is organised as follows:

1. Obtain climate measurements from Copernicus API:
    a. This service requires the geographical coordinates (latitude and longitude) and the day observation to perform the olive phenology prediction. This information is provided by the user through the interface.
    b. The service is connected to the Copernicus API. Once the aforementioned inputs are selected, the service launches a query to extract the temperature measurements from Meteostat weather API.

2. Olive Phenology Prediction:
    a. The simplest model has two inputs: The Day Of Year (DOY) and the Growing Degree Day (GDD). In this case, we use the ALLEN formula [4] to compute the GDD from maximum and minimum temperature measurements registered since January 1st.
    b. The pre-trained random forest (RF) model is applied to the selected DOY and the computed Growing Degree day.

[2] https://services.terrascope.be/timeseries/v1.0/ts
[3] https://docs.terrascope.be/#/Developers/WebServices/TimeSeries/TimeSeriesService, operated by VITO
[4] https://gitlab.com/demeterproject/wp4/decisionsupport/4.a.cropgrowthstatusyield/4.a.2-plantphenologyestimation/olivephenologyprediction

c. The result is the BBCH rescaled, an integer representing a specific olive phenology state. The corresponding physical meaning is extracted from an integrated table.
d. To visualise the evolution of such parameters, the BBCH is computed for all days (from the first of January until the Day Of Observation), creating a time series containing three columns: DOY, Growing Degree Day and the BBCH. This time series is converted to the corresponding AIM format to transfer this data to Knowage.

3. Dash (Flask based web service):
   a. The olive phenology prediction service is embedded in a Flask application that can be run locally after downloading it from GitLab[4].

Below, the results of *Analysis of Copernicus' ERA5 Climate Reanalysis Data as a Replacement for Weather Station Temperature Measurements in Machine Learning Models for Olive Phenology Phase Prediction* [5] are presented to demonstrate the efficiency of considering ERA5 temperature measurements to predict the olive phenology state. The scenario 2 represents the simplest model.

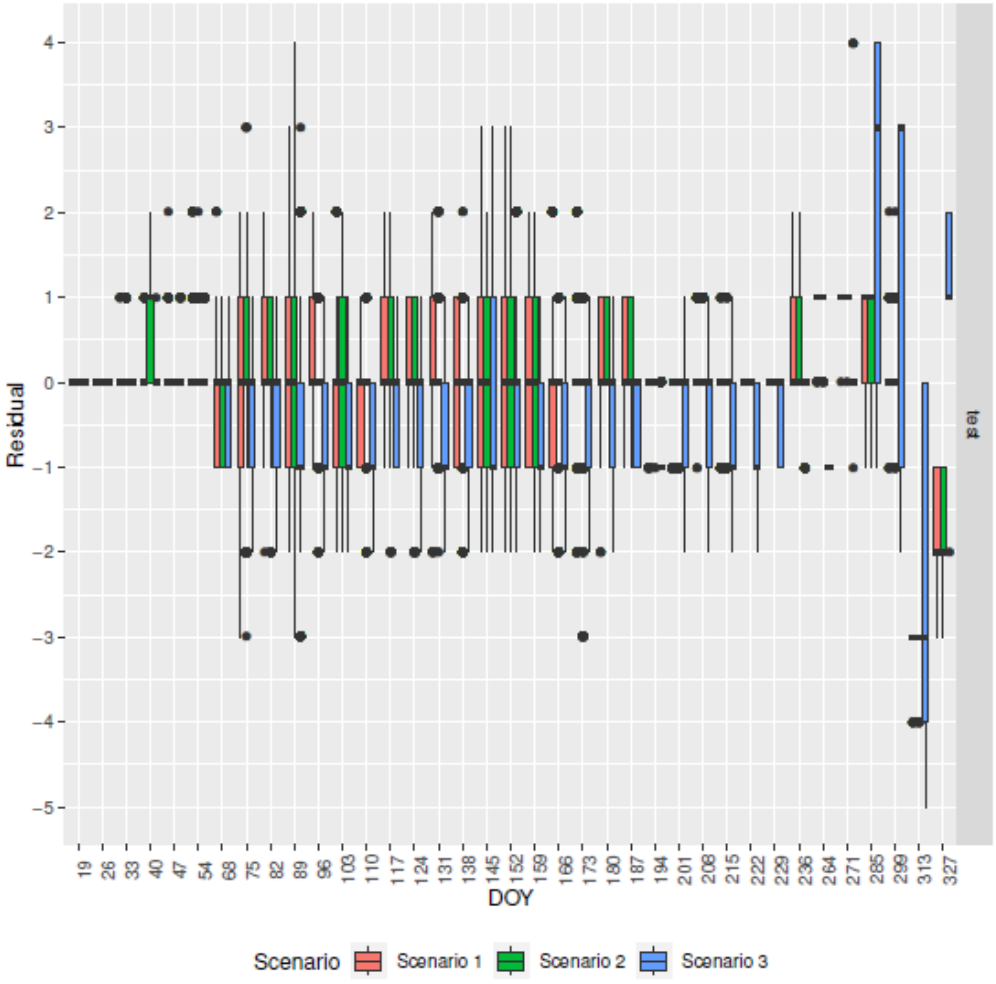

Figure 1: Comparison of the residuals by DOY for the different models in the scenarios from Table 1

Scenario	Data group	Feature set	Selected Model
Scenario 1	Weather station data	DOY, GDD (Allen)	Random Forest
Scenario 2	ERA5	DOY, ERA5_GDD (Tavg)	Random Forest
Scenario 3	Weather station data	GDD (Allen)	Agricolus baseline

Table 1: Scenarios for comparing selected ML models to Agricolus' baseline model

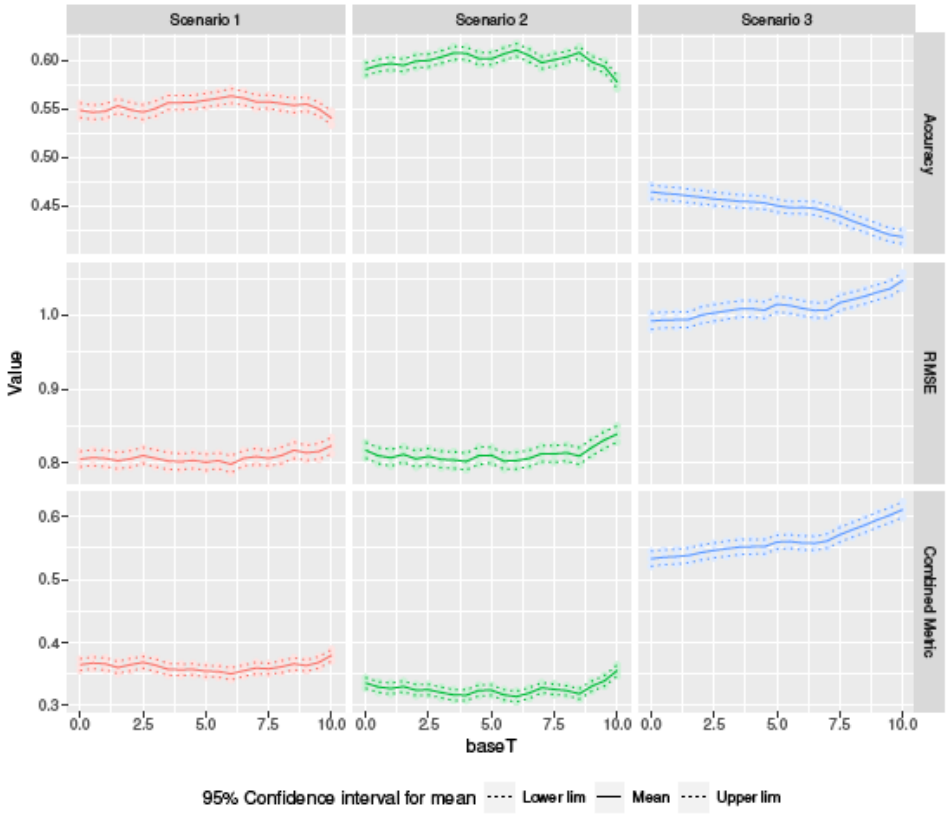

Figure 2: Base temperature optimisation for the scenarios described in Table 1: confidence intervals for Accuracy, RMSE and combined metrics

A most complex model based on the extra-tree regressor model has also been published ([6]), and as all the input parameters belong to Copernicus ERA5 climate data, it will be soon updated.

Another fact to consider is the query duration. In fact, the query with respect to this API can last more than 45 minutes, so recently, in order to provide a fast BBCH prediction, the weather API Meteostat has been connected. In the future, there will be two options: more precise prediction based on Copernicus data and fast prediction based on Meteostat temperature measurements. In addition, the weather API Meteostat provides temperature forecasting for five future days, and therefore the BBCH forecasting for those days.

6.3 AI model for Component 4.A.4: Crop Type Detection

The goal of this component is to detect the crop type for a given polygon and a given timeframe (growing season of the crop), using satellite data as input. Detecting crop type with satellite data allows us apply detection to any region in the world. There are however several factors that make this task very challenging:

- Different planting periods.
- Specific field management practices.
- Crop species that look similar.
- Regional differences in growing season.
- Clouds in satellite images that obscure our view on the field.

By using the right satellite data, and combining input from different satellites, we try to tackle these issues and produce a crop type map from space.

The model for crop type detection in this component is implemented as a Recurrent Neural Network using the TensorFlow[5] deep learning framework. A recurrent architecture is chosen because we need to take into account not just individual images, but timeseries of images. Instead of looking at individual pixels in the field, the timeseries is composed of data averaged over the parcel, e.g., 1 NDVI value per field per timestep. Key to a good identification is to look at differences in the timeseries when crops start growing, flowering, maturing and eventually get harvested. It is in the timeseries analysis that the difference between different crop types become apparent. As an example, the difference in growing curve for winter wheat and potato are depicted in the Figure 2. As one can see, winter wheat peaks much earlier in the season than potato.

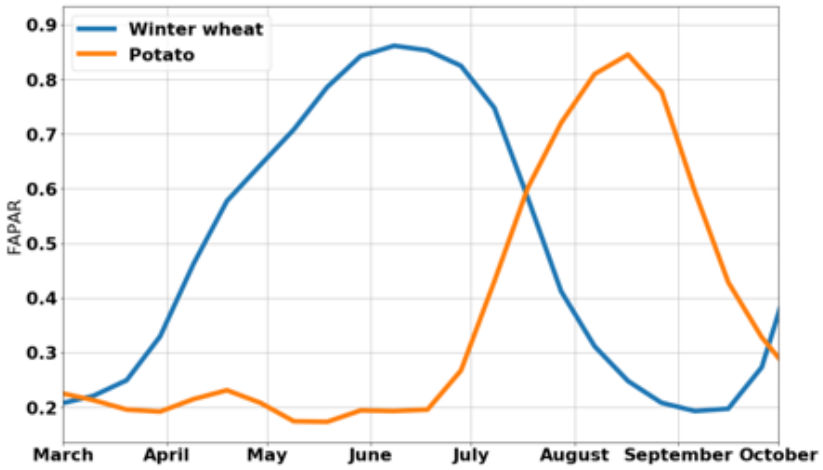

Figure 3: Growing curve for winter wheat and potato

As input, we use a combination of Sentinel-1 and Sentinel-2 data. While optical imagery from Sentinel-2 provides us with information on biophysical plant properties, the images may be obscured by clouds. We therefore also use Sentinel-1 radar data that can look through clouds, providing us with a reliable source of information at regular time intervals. In addition, Sentinel-1 data provides us with information on plant structural properties as well (e.g., elongated plant structures of maize vs. low closed canopy structures of potato).

[5] https://www.tensorflow.org

The input to the recurrent neural network is thus:

- A Sentinel-2 timeseries on field-polygon level, with start and end of the timeseries fixed (e.g., March – October for crop detection in Belgium). The timeseries holds information from all Sentinel-2 bands, augmented with the NDVI:
  - Bad data points (cloud or shadow not detected by atmospheric correction module) need to be identified and removed.
  - Missing data for cloudy days needs to be interpolated, e.g., using a Whittaker smoothing algorithm.

- A Sentinel-1 timeseries, on field-polygon level, with start and end of the timeseries exactly the same as for the Sentinel-2 timeseries. The timeseries should contain info on Sentinel-1 Vertical (VV) and Horizontal (VH) polarisation signals, and both for Descending and Ascending orbits.

The 2 timeseries are first resampled to 5-day intervals, and then fed into a stacked LSTM network (Long-Short Term Memory, a specific type or recurrent neural network). Each input is fed into a separate LSTM stack, while the outputs of both LSTM stacks are concatenated in the second layer of the network. The final layer (output layer) is a classifier implemented as a dense neural network, with the number of output neurons equal to the number of crop types we wish to detect.

The input timeseries will be collected using OpenEO[6]. However, the workflow for downloading the input data will be split from the workflow doing the prediction. This will give the end-user the option to gather his input data from other sources, while still being able to use the AI model for training or prediction. To facilitate the choice of input data sources, timeseries input to the model is expected to conform to the DEMETER Agricultural Information Model (AIM) format.

The AI model implementation is completed, but the rest of the workflow is still under development. The AI model has been tested with historical datasets available on disk, but not yet with new input from OpenEO. The next steps will focus on the use of OpenEO as a service to gather the necessary inputs from Sentinel-1 and Sentinel-2. To train the model on crop type prediction, crop type information from the Flemish Department on Agriculture and Fisheries has so far been used, transformed into an AIM format.

6.4 AI model for Component 4.B.2: Reference Evapotranspiration Prediction

The aim of this algorithm is to combine time series model predictions using ML with the Reference Evapotranspiration (ET_0) calculated using the *Penman-Monteith* method ([10]) with weather predictions from several meteorological services to obtain the final forecast, value that can be used then to schedule a dynamic irrigation planning (see Figure 4).

[6] https://docs.terrascope.be/#/Developers/WebServices/OpenEO/OpenEO

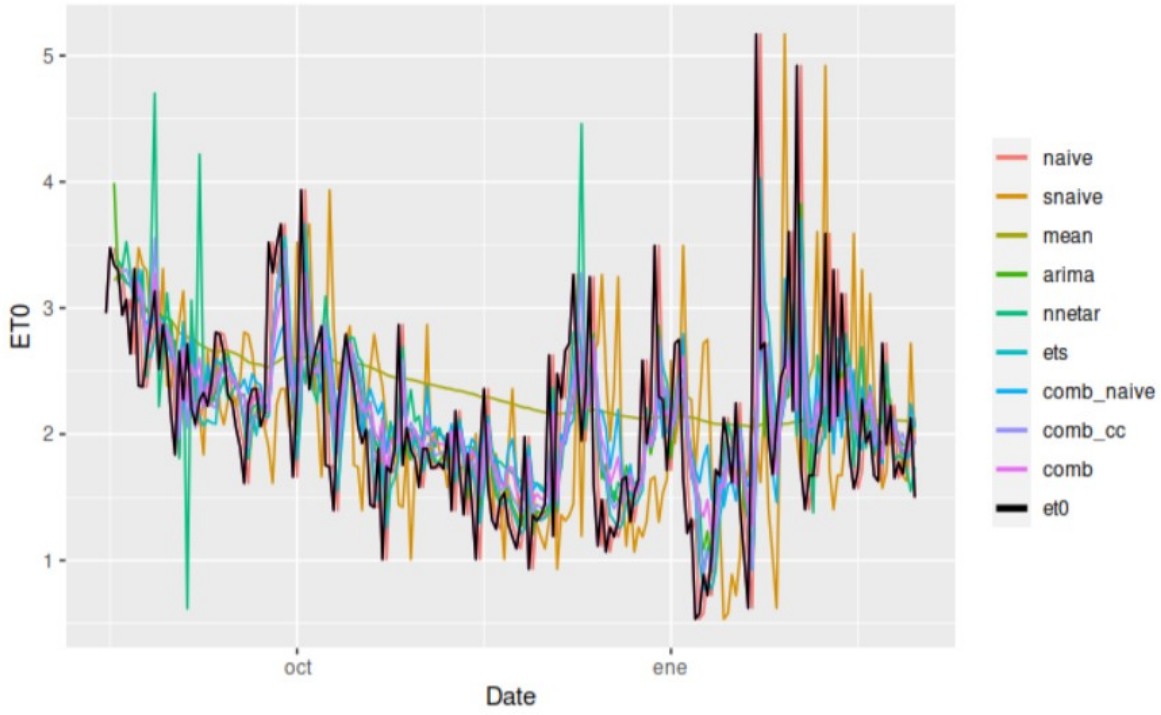

Figure 4: Predictions of several forecast models (1 day ahead) across 200 days where real ET_0 is displayed in black

This algorithm is run daily, and each implemented time series prediction model (TSM) is executed storing its ET_0 predictions for the next days in a database. Then, the most suitable TSM is decided comparing most recent day's stored data (ET_0 predictions in previous execution) with the most recent day's real ET_0 calculation (*Penman-Monteith* method) selecting the one that least underestimates it.

More concretely, in a daily basis the following steps are performed to obtain an ET_0 prediction ensemble:

1. Obtain meteorological data of the previous day(s).
2. With the prior data, compute the ET_0 and insert it into the historical database.
3. For each of the available models, compute the prediction of ET_0 for the following days and insert it into the database.
4. Obtain meteorological forecasts from well-known sources (i.e., OpenWeather) for the following days.
5. Given that data, compute ET_0 and insert it into the database.
6. Analyse the historical accuracy of all the previous models and select which is the more suitable to choose as valid.
7. Select the prediction computed in steps 3 and 4 of the model that has been selected in step 5.
8. Publish the prediction.

Next, Figure 5 represents a flow chart with the different steps is shown:

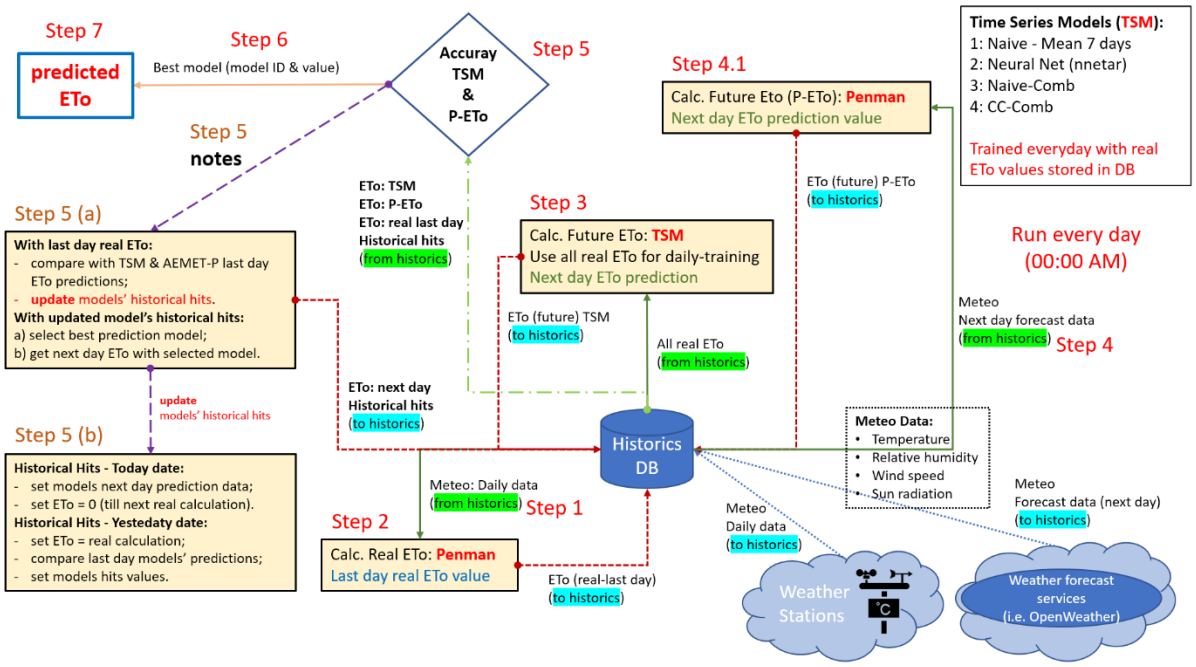

Figure 5: ET_0 prediction steps

For this algorithm, a diverse selection of TSMs has been considered:

- Based on naive assumptions like the mean of ET_0 in the last 7 days (*mean*).
- Complex autoregressive neural nets (*nnetar*).
- Combinations of those models:
  - Combination model for the naive models (*naive-comb*).
  - Combination model for the most complex ones (*cc-comb*).

- ET_0 predictions computed using the *Penman-Monteith* method with weather forecasts (i.e., OpenWeather) among others will also be added as additional models.

This process is performed on a daily basis building a database that is used in each iteration to make the current predictions and to assist the future ones.

In the next table an example that represents the RMSE (*Root Mean Square Error*) ET_0 values of each model in each week during the last 10 weeks can be seen. Here it is possible to observe that the best weekly model (marked in red) is not fixed, so it is our work to select each time which is the most suitable model for each of the ET_0 predictions we offer.

ET0 RMSE of several models in the last weeks (OpenWeather excluded)

week	naive	snaive	mean	arima	nnetar	ets	comb_naive	comb_cc	comb
2020-12-21	0.367	0.589	0.443	0.282	0.286	0.294	0.260	0.273	0.265
2020-12-28	0.590	0.770	0.478	0.570	0.541	0.555	0.457	0.526	0.500
2021-01-04	0.422	1.502	1.134	0.610	0.600	0.554	0.975	0.573	0.770
2021-01-11	0.379	0.950	0.380	0.281	0.256	0.408	0.369	0.288	0.305
2021-01-18	1.246	1.023	1.123	1.187	0.984	1.231	0.938	1.107	1.019
2021-01-25	0.626	0.971	0.423	0.589	0.432	0.633	0.497	0.546	0.489
2021-02-01	1.004	0.991	0.709	0.956	0.904	1.003	0.840	0.884	0.875
2021-02-08	1.098	0.874	0.614	0.740	0.745	0.789	0.609	0.714	0.673
2021-02-15	0.254	0.786	0.332	0.317	0.293	0.304	0.366	0.291	0.337
2021-02-22	0.457	0.318	0.334	0.307	0.263	0.323	0.312	0.314	0.293
2021-03-01	0.428	0.507	0.315	0.374	0.410	0.369	0.333	0.335	0.251

Figure 6: RMSE of several models in the last weeks (the best weekly model is depicted in red)

It is known that farmers use the average weekly ET_0 to plan their short-term irrigation for next days, and hence the goal with this new approach is to provide a more systematic and quantitative ET_0 prediction.

This prediction takes into account both a weather forecasting service (the availability of multiple weather forecasting services would allow the use of data fusion with their predictions for a same location), and different predictive models based on time series to select the most accurate prediction while minimising water consumption.

6.5 AI model for Component 4.B.3: Soil Moisture Estimation

We use the data provided by the infield soil moisture probes (one-point data) to generate a model capable of quantifying the amount of water on the plot surface (2D data), complementing the soil probes information. For this purpose, an implementation of an optical trapezoidal model (OPTRAM) ML algorithm ([9]) has been performed that is fed by both soil moisture probe data and Sentinel-2 multispectral images. The output of this model can then be integrated into the irrigation model to better estimate the amount of irrigation water that should be supplied to the crop to keep it at field capacity (see Figure 7). In this figure, the model is able to determine in physical units the amount of water on the surface of a plot through satellite multispectral images and a soil moisture probe.

Establishing soil moisture not only in a specific point of the plot but in the plot extension (2D) can be relevant to generate localised irrigation in those areas of the plots that more need it, with consequent water savings. Due to the technical constraints of this model, its moisture detection range is limited to the surface of the plot, so we are analysing if it is possible to correlate its results to deeper soil levels.

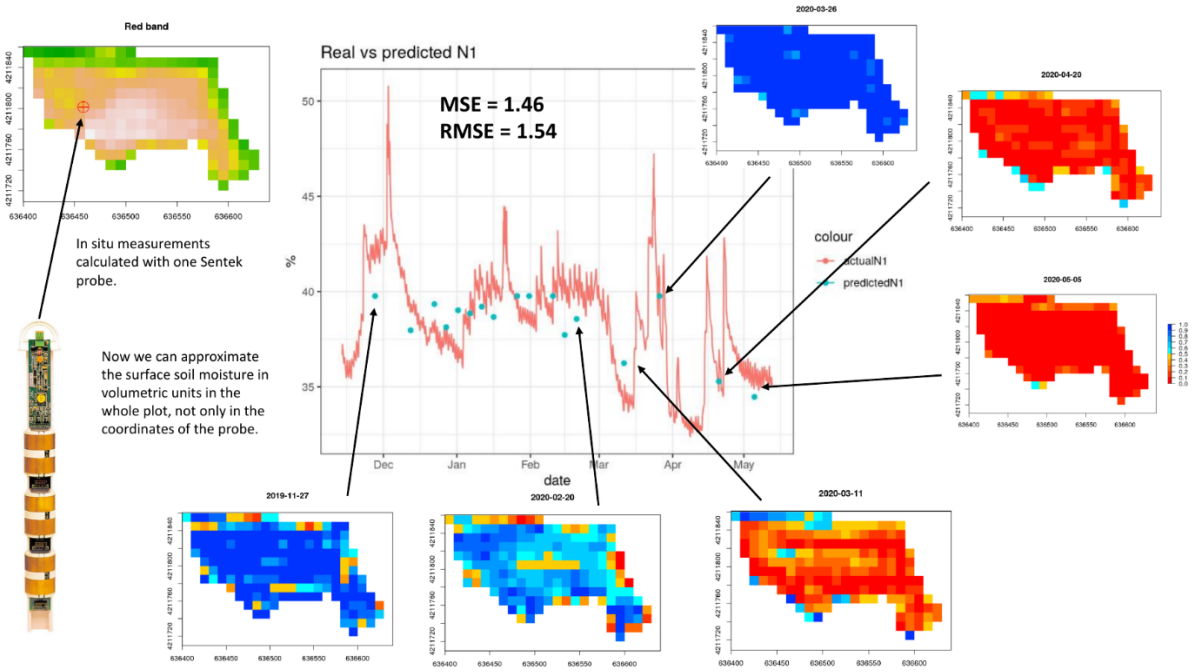

Figure 7: Summary of OPTRAM results

To build this model in a plot, the following steps are performed:

1. Collect Sentinel-2 satellite multispectral images for the longest possible time interval.
2. For the same time interval, retrieve the corresponding data from the ground soil moisture probes.
3. With the Sentinel-2 images build the shortwave-infrared transformed reflectance (STR) and normalised difference vegetation index (NDVI) space (shown in Figure 8).
4. Fit the wet (*STRw*) and dry edges (*STRd*) of the previous space.
5. Calculate the normalised soil moisture content $W = f(STR, STRd, STRw)$ for each of the available pixels of the plot, including the pixels belonging to the location of the ground probes.
6. Perform linear regression (see Annex A.4 for a brief introduction to Linear Regression) analysis to obtain the soil constants minimum dry (ϑd) and maximum wet soil moisture (ϑw).
7. Compute surface soil moisture $\vartheta = f(W, \vartheta d, \vartheta w)$ for the ground truth datapoints (shown in figure above in "*Real vs predicted N1*" central plot).

In the next figure (Figure 8), the historic data of all ground soil moisture probe levels, where each level is 5 cm deeper than the previous one and N1 is the closest level to the surface, can be seen. The cross-dots displayed on the N1 time series represent the precise time the satellite Sentinel-2 took a picture. Since the exact position of the infield probes is known, it is possible to calculate what the exact soil moisture value in that moment and point was. These points are called "ground truth data points" and they will be compared with the model predicted values (see Figure 5).

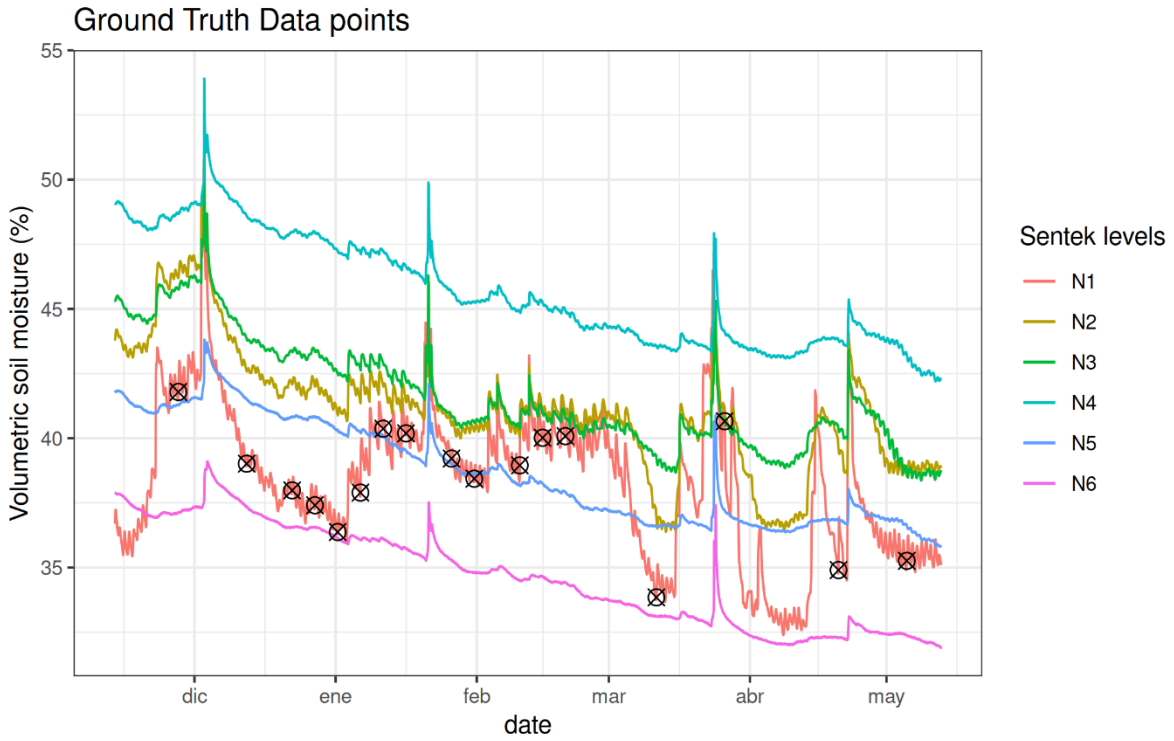

Figure 8: Soil moisture probe (6 levels) time series values (N1 is the closest level to the surface) and Sentinel-2's time series of captured images (cross-dots on N1)

In the next figure (Figure 9) the NDVI (the normalised difference vegetation index) vs the STR (the SWIR transformed reflectance) can be seen. Pixels corresponding to the soil probe location are coloured in red.

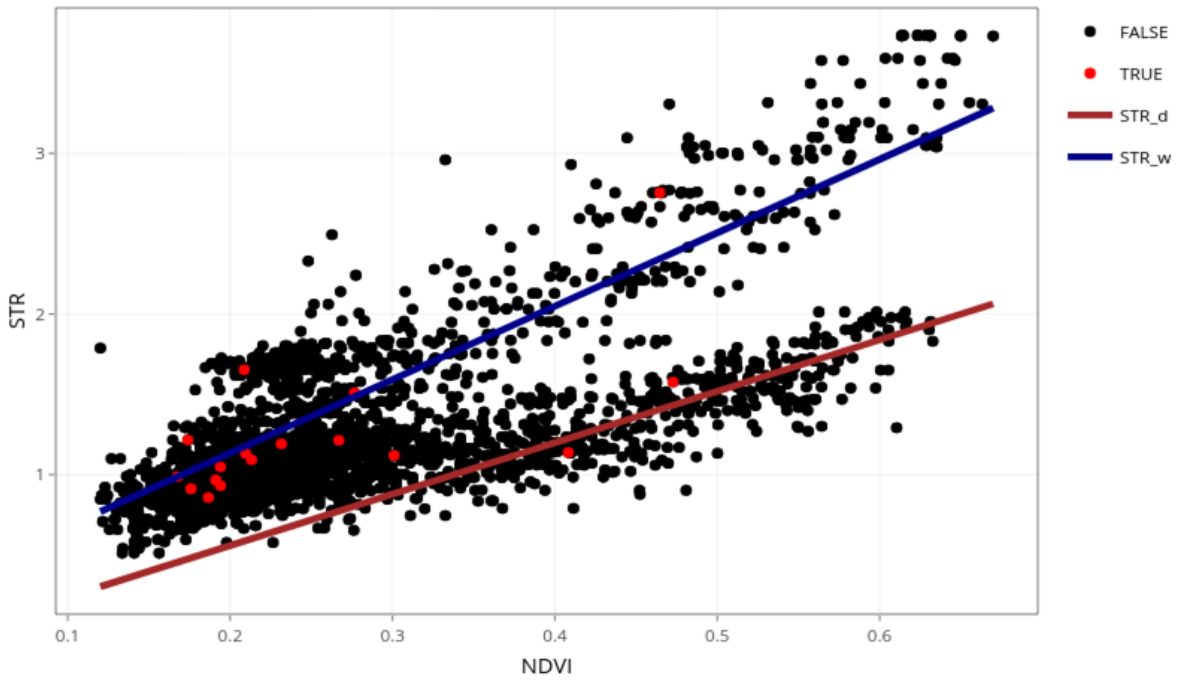

Figure 9: The NDVI-STR space

6.6 AI model for Component 4.B.4: Crop Water Status Anomalies Detection

The aim of this model is to identify possible anomalies in the crop extension (2D) related to the plants' water status. This method for crop anomaly detection is based on multispectral analysis of images provided by Sentinel-2 satellite. The images corresponding to the crop during several seasons are compared with the latest image obtained to classify, using ML techniques, the pixels into several categories according to the expected behaviour extracted from the historic data of the same crop in the same or adjacent plots.

This method for crop anomaly detection is based on multispectral analysis of images provided by Sentinel-2 satellite. The images corresponding to the crop during several seasons are compared with the last image obtained to classify, using ML techniques, the pixels in several categories according to the expected behaviour extracted from the history of the same crop or the adjacent ones.

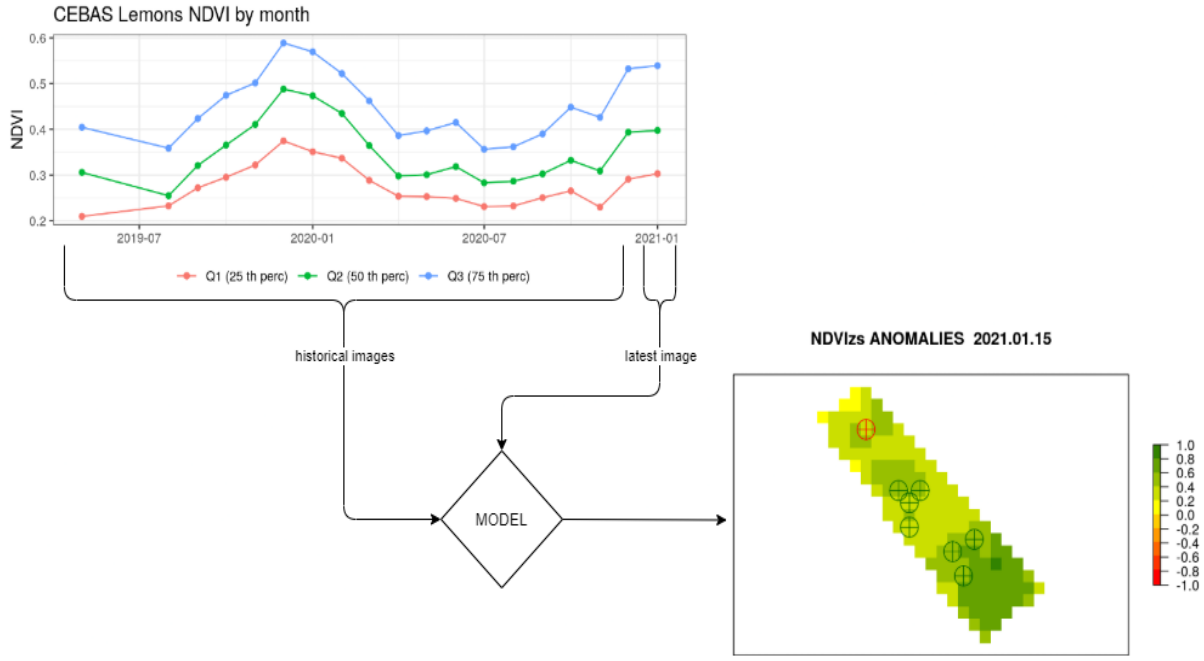

Figure 10: Example of the process of estimating the state of the plot

Crop water status can be determined with various vegetation indices and temporal ranges depending on the quantity and quality of available imagery.

One of the best-known metrics for determining an anomaly in unsupervised scenarios is the Z-score map, which measures the number of standard deviations a point is away from the mean. In this way we can calculate the Z-score for each of the pixels of the most recent image compared to the corresponding pixels of previous seasons, which forms the mean or normal crop condition.

$$NDVI_{Z-score} = \frac{NDVI_n - \mu_{NDVI_{1\ldots n-1}}}{\sigma_{NDVI_{1\ldots n-1}}}$$

Figure 11 : Formulae for the Z-score metric

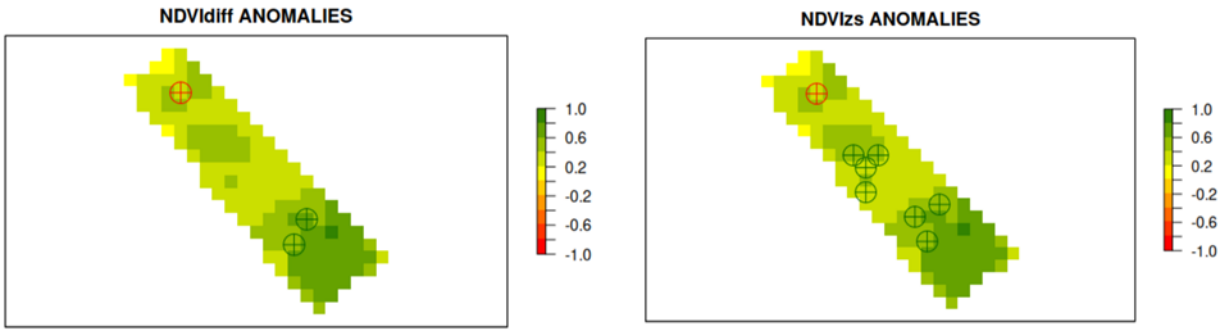

Figure 12: Output example for 2 different techniques for anomaly estimation

The figure above presents the anomaly outputs for two techniques applied to the same inputs, note that the two coincide in the arrangement of the anomalous points, which is a good indication that there are indeed anomalies.

To build this model in a plot, the following general steps are performed:

1. Collect Sentinel-2 satellite data for the longest available time period.
2. Collect the latest Sentinel-2 image.
3. Use the previous inputs to feed the specified unsupervised metrics.
4. Filter out the inactivity seasons depending on crop type.
5. Adjust for cloudy images.
6. Aggregate and display the resulting anomalies (if any) for the visual inspection for the decision makers.

We are currently working on the refinement and validation of the models. Due to the difficulty of finding well-documented anomalies in the pilot crop we will create our own anomalies based on real data. Given an NDVI image we take several points on the map and replace their values with those corresponding to a phenological stage totally different from the current one. In this way, we simulate the effect that there are several lemon trees that are not developing correctly and, thus, we can assess the specificity and sensitivity of the model.

6.7 AI model for Component 4.E.1: Pest Estimation with Sterile Fruit Flies

Sterile fruit flies counting is a task that involves several challenges. The main goal of this component (framed in the context of pilot 3.3) relies in counting sterile and non-sterile fruit flies from images. The images are expected to be taken from automatic traps that will make use of ultraviolet light to see dye applied to the sterilised flies (as it fluoresces). With that in mind, the component was designed to count the different flies in each image captured in the images to have an estimation of the sterile/non-sterile flies' ratio in the field. That component was proposed to make use of Neural Networks (NN, see Annex A.3 for a brief introduction to NN) in order to identify the different flies in the images.

Due to the initial lack of training images from the pilot, the component was initially proposed as a generic element counting component to be applied in pilot 3.3.

Regarding the advances so far, the initial efforts have been put into two different tasks in parallel: i) an initial version of the generic component for element counting from images, which has been developed in Work Package 2; and ii) the collection of training data for the component (which, due to the timing, is being carried out under lab conditions, with around 500 labelled pictures taken). Now

that an initial local version of the generic counting component has been implemented, our efforts are being put into the integration of that component with the DEMETER architecture in parallel with the creation of the component 4.E.1. Additionally, more images need to be collected from the pilot in order to evaluate the functioning of the component, since some issues were identified with the current images provided by the pilot: i) the images are taken under lab conditions (see Figure 13), which are expected to be different to the real conditions from the pilot (and the models trained with those images might not be a good fit for the pilot conditions), ii) due to the nature of the images, the labelling process is slow and the number of images is not high, iii) according to the pilot feedback, it is hard to distinguish between sterile and non-sterile fruit flies even for the experts (which makes it even more difficult for the models to identify the flies correctly). It is difficult to address the aforementioned issues prior to the automatic trap deployment in the pilot, so these issues are expected to be addressed when the pilot status advances, getting images from the real traps (first and second issues introduced). Once the final images that will be used in the pilot are captured, they should be labelled according to the experts' criteria (second and third issues introduced).

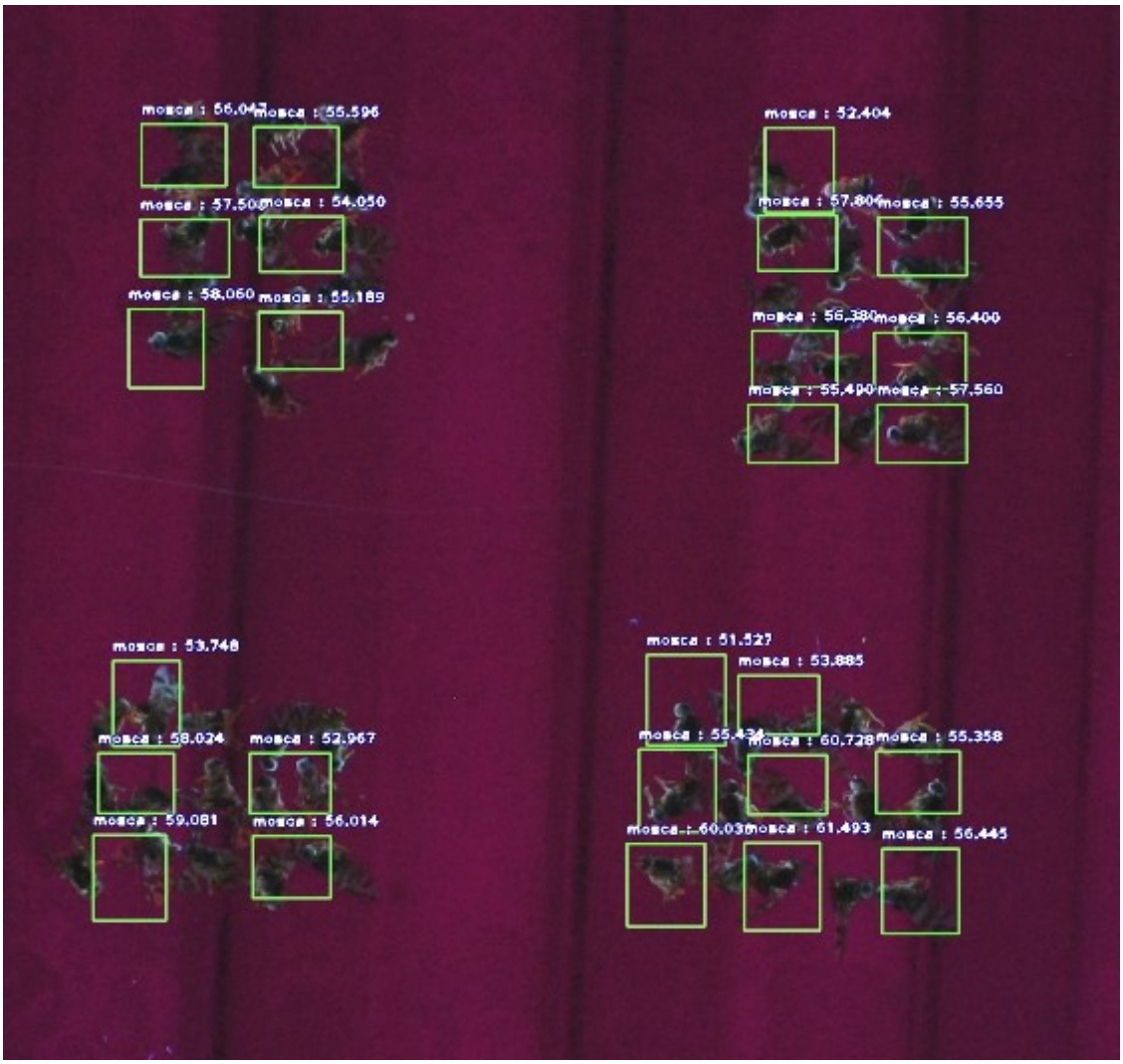

Figure 13: Flies detected in the initial testing images

6.8 AI model for Component 4.E.2: Estimate Temperature Related Events

The target of this AI model is the prediction of pest events and stages based on the weather condition. The source code of this component is available at DEMETER's GitLab repository[7].

Temperature directly affects the phenological development of pests, therefore temperature can be used for building weather-driven models that simulate pest development and the appearance of the different pest stages. Standard methods to simulate the main events in pest development cycle are based on the day-degree calculation, which is the heat unit accumulation above a base temperature.

The component is based on docker containers to simplify the installation on premises at pilot level. The model prediction is based on a machine learning model (i.e., Random Forest) requiring as input daily maximum and minimum temperature data. To obtain air temperature data, we have adopted the Copernicus API to get the ERA5 dataset.

To test the component, we have chosen a target species and specific pest events. The model has been developed on olive fruit fly which is the key pest of the olive tree agroecosystem. As target event we have chosen the peak of flight of each generation of the pest in the summer. Data used to train the model were collected by olive fruit fly monitoring activity in a set of farms in Tuscany (central Italy).

To build this model, the following steps are performed:

- Prepare a training set with latitude, longitude, and the day of the year, in which a peak of flight was observed.
- Get the weather data from the Copernicus ERA5 API.
- Calculate the day-degree indicator on the Copernicus data.
- Build the model by running a random forest algorithm, which uses the day-degree as the input and the pest peak of olive fruit fly flight as the target.
- The result of the model is saved in the component folder to be used by the prediction web services.

The component is a web service used to predict the peak of flight using as input weather data. The workflow of the component is:

- Getting the latitude and longitude of the farm from the farm AIM data model.
- Connecting with Copernicus ERA5 data service to collect the weather data for the point of interest.
- The day-degree accumulation is calculated from the ERA5 data using the Allen formula ([4]).
- The random forest pre-trained model is used to predict for that specific date if the peak has been reached.
- The output of the model has a daily time step as well as the status of the probability level of the random forest algorithm.
- The result of the model is formatted according with the AIM data format, using the Observation Class from the Sensor-Observation-Sampling-Actuator ontology (SOSA).
- The API is used by the Knowage visualisation component to make the results accessible to the final user.
- A Python web application developed with the Flask framework has been used to publish the result of the API.

[7] https://gitlab.com/demeterproject/wp4/decisionsupport/4.e.pestanddiseasemanagement/4.e.2-estimatetemperaturerelatedpestevents/temperaturerelatedpestevent

6.9 AI model for Component 4.F.1: Estimate Milk Production

Our hypothesis is that AI/ML as a method will create more robust or precise forecasting models. Our objective is to develop individual cow specific lactation curves as a basis for predicting future milk yield.

Data structure. The milk yields are predicted on a 15-day interval over a total of 345 days. The input data is a combination of previous milk yields, with the same periods and intervals as the output, and a variety of continuous and discrete variables. Among these are the breed, calving number, time of year, feed consumption and days in milk.

Machine learning algorithm. The algorithm used in our forecast model is named CatBoost[8]. This algorithm is a well-established gradient boosting regressor with special support for categorical features. Gradient boosting regressors work by combining lots of weak learners (small decision trees) to one big model. The model is trained tree-by-tree and therefore allows the choosing of new trees and the values in each tree to be dependent on previous trees. The choosing of weights on the base trees is done by gradient descent on the loss function. The loss function is a MultiRMSE. Which is the mean RMSE for each prediction (23 steps in our model).

Hyperparameter tuning. The hyperparameter tuning was done using grid-search on the following parameters:

- 'depth': [5,8,3],
- 'bagging_temperature': [1.2,1,0.8],
- 'l2_leaf_reg': [3,2],
- 'learning_rate': [0.1,0.05,0.01],
- 'grow_policy': ['Depthwise','SymmetricTree'].

The best score is chosen by the following function:

$$score = \frac{\sum_{i=1}^{N} \left| \sum_{d=1}^{dim} (a_{i,d} - t_{i,d}) \right|}{N}$$

Where:

- *N* = number to predicted (rows in dataset),
- *dim* is the number of timesteps ahead to predicted (columns in dataset),
- *a* is the true value and t is the predicted.

This function can be referred to as the Mean Absolute Cumulative Error. Figure 14 demonstrates below an example of observed and predicted milk yield for an individual cow by the ML algorithms.

[8] Open-source Gradient Boosting library - https://catboost.ai/

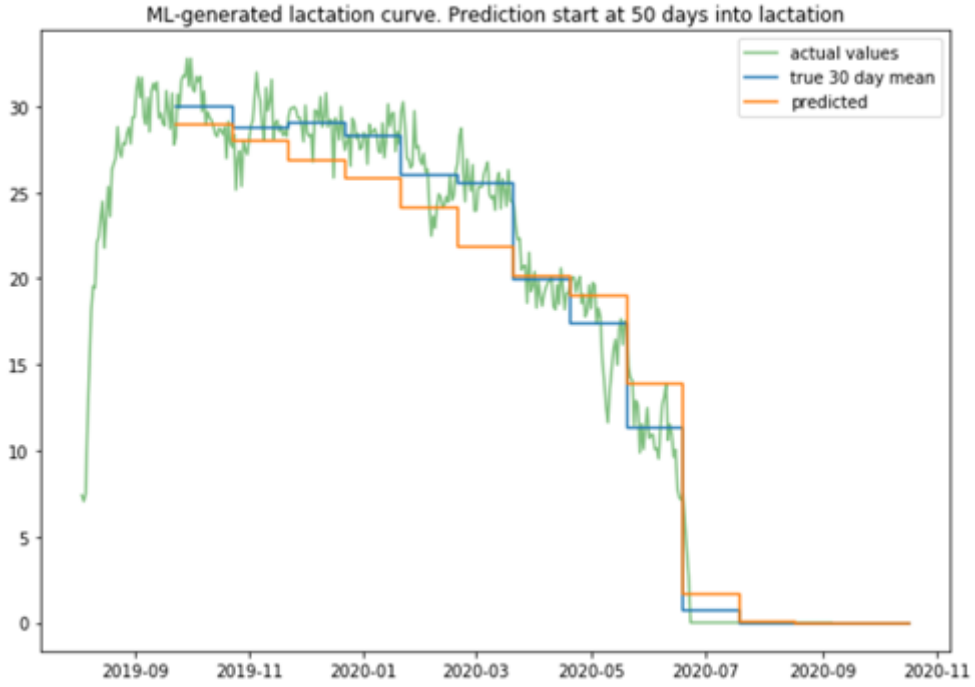

Figure 14: Observed vs. Predicted milk yield for an individual cow

6.10 AI model for Component 4.G.1: Estimate Animal Welfare Condition

Animal welfare represents a fundamental aspect of sustainability in livestock production. The use of technologies of AI to discover fundamental information such as those relating to possible diseases that could affect the cow's herd or an individual animal, represent on the one hand a challenge for the farmers (producers) in the digital transformation of their information systems or FMIS and, on the other, a very important innovation that revolutionises the way of doing business. Just think of the use of antibiotics, which could be reduced significantly by means of careful monitoring by the producer, supported by a decision support system on animal welfare. It is beyond doubt, that integrating an AI technology that influences the production method of meat and its derivatives (such as milk and dairy products), represents an advantage for everyone, including consumers. Obviously, measuring the well-being of animals using digital algorithms is not a simple undertaking; it requires a good collaboration between the ICT system integrator and the farmer, good data quality, and equipment like sensors that are responsible for data production. This combined technology set makes the result of an AI algorithm elaboration sufficiently close to reality and provides the necessary outputs to the DSS. Data presentation happens in a special dashboard to convey to the farmer the most important messages: herd health is a reflection of animal welfare. The parameters related to the physical stress for instance, are, very often, the mirror for much more important pathologies. Any attempt to assess well-being through the use of AI algorithms and decision support systems must take into account authoritative sources, such as latest animal welfare research.

The combination between internal algorithm's business logic and the technology used to implement it represents the real key to the success or failure of a machine learning algorithm on animal welfare. The machine learning algorithm on Animal Welfare allows the evaluation of the health status of the cows analysed, to determine the degree of well-being, in terms of nutrition, hygiene, rest and movement and, consequently, also to evaluate their productivity. Among all the machine learning algorithms, we have selected the Random Forest (see annex A.2 for a brief introduction); one of the

most relevant and effective for the topic addressed. Since the AI part of the component in Estimate Animal Welfare Condition concerns the health status of animals (cows), it is clear that this case must be addressed using a classification method.

The training data stream contains the data concerning the **fat/protein ratio**, the **electrical conductivity**, the **total days at rest** and the **total daily rest**, accompanied by the health status by pathology (**ketosis**, **mastitis**, and **lameness**) assigned "manually" by the farmer. By processing this flow, the Random Forest algorithm learns in what circumstances a cow is healthy and in which circumstances it is sick. In particular, the algorithm analyses the aforementioned characteristics as follows:

- Lameness, it analyses activity at rest: if the cow rests too little or too much, it may get too tired and limp or may already be lame and have difficulty getting up.
- Ketosis, it analyses the relationships between fats and proteins: if the diet is too unbalanced on fats, this can favour the onset of a ketosis status.
- Mastitis, it analyses the values of electrical conductivity: if the electrical conductivity of the milk taken exceeds a certain threshold, it is very likely that the cow is suffering from mastitis.

The following Figure 15 shows a dashboard with a table containing the data concerning the fat/protein ratio, the electrical conductivity, the motor activity, and the total daily rest, accompanied by two parameters: the health status by pathology (ketosis, mastitis, and lameness) assigned "manually" (Actual Column) and the health status by pathology assigned by the algorithm (Predicted Column), determined by the latter through training performed by studying the values assigned "manually".

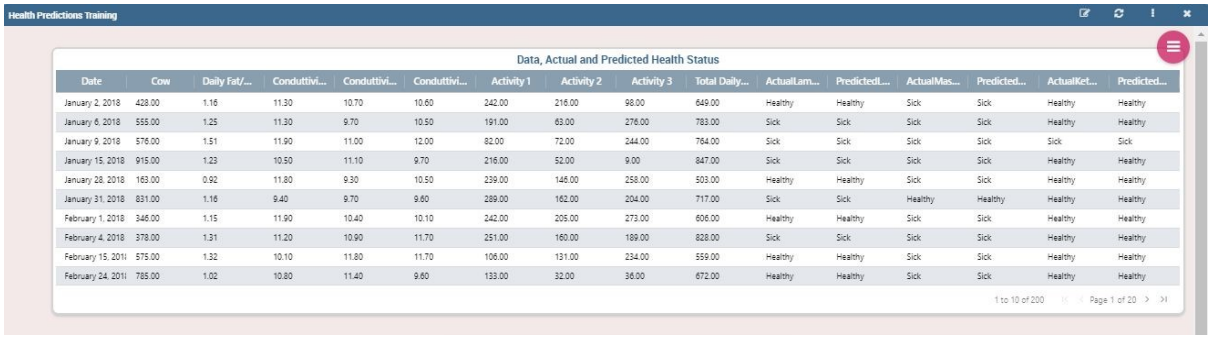

Figure 15: Health Predictions Data Table – Training

The dashboard also shows, for each pathology, two pie charts, to compare the health status assigned "manually" with those determined by the algorithm and a table showing the metrics[9], described below:

- **True Positive Rate** indicates how many "really healthy cows" (the true positive values) have been identified compared to the sum of the latter and the cows "wrongly classified as sick" (the false negative values).
- **False Positive Rate** indicates how many cows "wrongly classified as healthy" (the false positive values) have been identified compared to the sum of the latter and the cows "really sick" (the true negative values).
- **Precision** indicates how many "really healthy cows" have been identified compared to the sum of the latter and the cows "wrongly classified as healthy".

[9] The meaning of the metrics is explained in Annex A.1.2

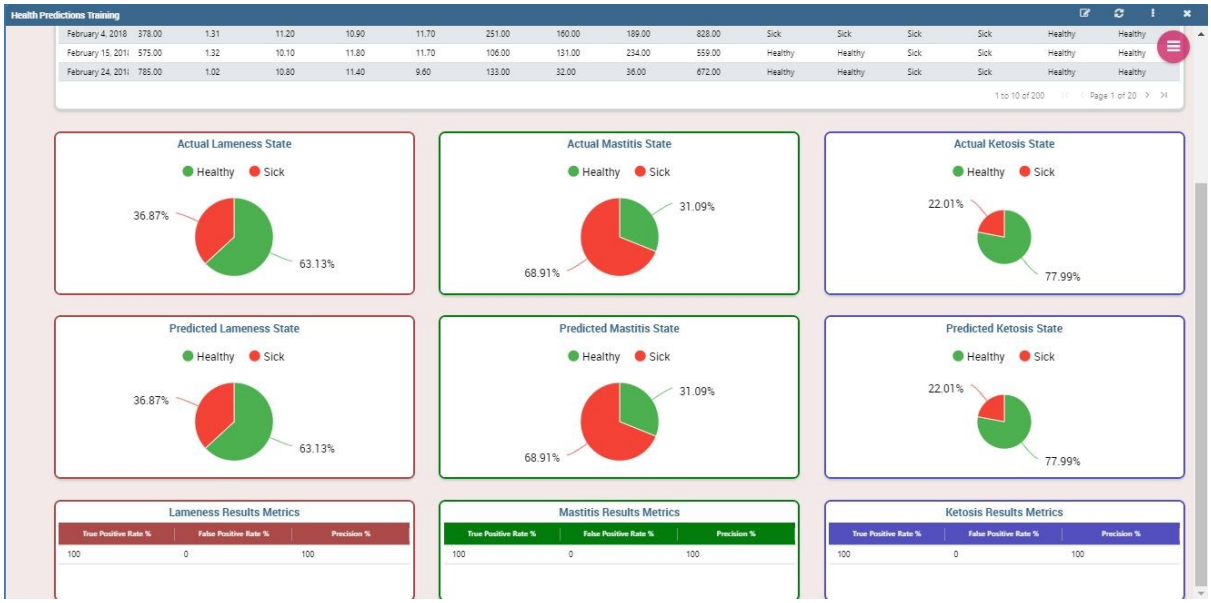

Figure 16: Health Predictions Dashboard Charts – Training

After obtaining a high degree of precision (see Figure 15, "Predicted column"), it is possible to see the new predicted values by the algorithm.

By observing the table and the forecasts made by the algorithm, it is possible to determine which parameters must be checked to improve the quality of the cow life, and this is precisely the objective of Animal Welfare: monitor to correct any anomalies that affect the health status. If the percentage of cows with mastitis exceeds a certain threshold, the hygiene procedures must be reviewed; if this happens for the ketosis ones, the feeding must be modified, making it more energetic and if the percentage of cows affected by lameness is too high, it is necessary review the housing spaces.

6.11 AI model for Component 4.G.2: Poultry Well-Being

Poultry well-being detection requires sensing and quantification of many different parameters, such as environmental (quality of parameters from the air and in field) and physical (stress due to the power loses, level of movements of the poultry, potential illness detection, food intake, etc.). DNET provides novel solutions in the poultry domain based on agroNET[10], with a biosafety guide and an objective-based feed mixture module as the main features. The solution captures real time measurements of parameters and their integration with the analytics algorithms, as well as parameters taken from the ambient sound in the chicken coop. As there was a need for more accurate estimation of the poultry well-being that could indicate proper progress of the chickens in terms of health (process done manually), a video-based algorithm is designed and developed to monitor chicken activity and progress to provide additional intelligence.

The main objective of this algorithm for poultry well-being is to provide additional support to the existing platform for poultry monitoring, which is based on micro-electronics devices and to integrate the algorithm into a smart cloud-based system. The current AI is using sound processing algorithm to detect chicken stress, while the extension done is video-based detection of chicken well-being - the main parameters are classified chicken size from images and the chicken movements on farms to facilitate improvement of the breeding process. The proposed AI algorithm is much more complex and

[10] agroNET – www.agronet.solutions

requires careful deployment of edge devices to capture representative datasets, existing devices with video capturing capability, edge data processing and machine learning. This algorithm is developed in Python and runs directly on the edge device (i.e., the camera).

Figure 17 shows the output of the poultry well-being algorithm running on the camera, that estimates chicken size and movements. The algorithm calculates weight estimation and activity using images captured with camera. The annotated dataset is compiled from 3500 images captured for 35 days. The algorithm is based on round estimation of the area that represents the shape of each chicken. The algorithm was trained by using sets of less than 1000 images for chicken recognition and 150 images for weight estimation.

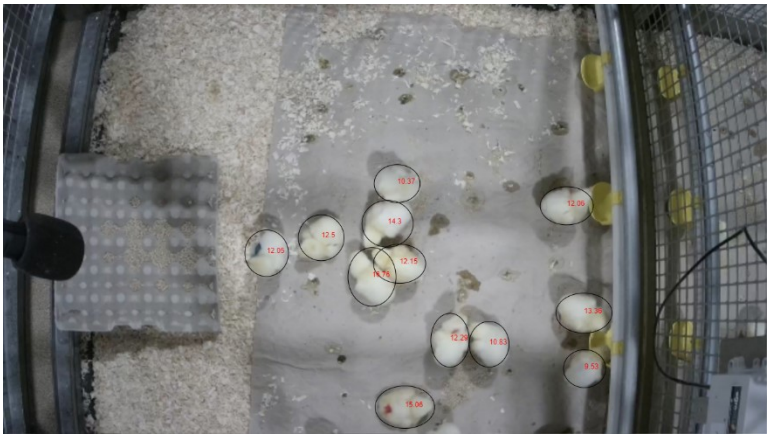

Figure 17: Image vision algorithm running of the edge (camera) for estimation of the chicken size

The main process of the algorithm development and integration in the existing system was as follows:

1. Functionality of the existing ML by adding video processing and implementing extraction of image features functionality on the device.
2. Integration with digital farming platform agroNET to enable acquisition of sensor data on the cloud side.
3. Implementing algorithm from image on the edge side.
4. Final deployment in pilot and validation of the developed functionalities on 1 farm with 3 different flocks.

To be able to finalise the algorithm some data is needed for the next cycle and algorithm upgrading:

- Daily weight per chicken.
- Daily activity – feeding time, medical treatment, among others.
- Marking of chickens – to assess starting time, e.g., from third week when chickens are bigger.

6.12 AI model for Component 4.H.1: Milk Quality Prediction

The smart dairy supply chain has become one of the most challenging and innovative areas where data analytics based on AI technologies can be applied. The algorithm used in this component focuses on the ability to predict milk quality by analysing a predetermined set of data collected during the lactations of dairy cows. Therefore, the use of a machine learning algorithm requires the training data be qualitatively representative, to allow the algorithm to “understand with some certainty and reliability” whether the analysed sample refers to high quality or low quality milk. The technique used

upstream, or Fourier-transform infrared spectroscopy (FTIR) method is used globally to predict several milk quality parameters. FTIR analysis was used for the extraction of data relating to the milk quality, as a technique to obtain reliable data to be provided to the algorithm at the training level (before) and prediction level (after).

The training data stream contains the data concerning the **FTIR** analyses accompanied by the grade of quality assigned "manually". By processing this flow, the Random Forest algorithm then learns in what circumstances is the milk of **high**, **medium**, or **low quality**. The algorithm analyses the following parameters: **caseins**, **density**, **fats**, **proteins**, **cryoscopic point**, **pactose**, **urea**. If the milk is not very dense and the cryoscopic point is too high, for example, the percentage of water present in the milk is probably excessive. And this indicates the presence of a milk of less than excellent quality. Same thing for fats and proteins that indicate the genuineness and nutritional value of milk. Low fat, for example, indicates a low nutritious milk. Too much uric acid indicates that the cow is on the wrong diet and is probably consuming too much protein.

The dashboard below shows, both for raw and processed milk, a table containing the FTIR (Fourier-transform infrared spectroscopy) analyses and, for each type of milk, two pie charts, to compare the quality grades assigned "manually" with those determined by the algorithm. Here too we find a table showing the metrics:

- **True Positive Rate** indicates how many "really healthy cows" (the true positive values) have been identified compared to the sum of the latter and the cows "wrongly classified as sick" (the false negative values).
- **False Positive Rate** indicates how many cows "wrongly classified as healthy" (the false positive values) have been identified compared to the sum of the latter and the cows "really sick" (the true negative values).
- **Precision** indicates how many "really healthy cows" have been identified compared to the sum of the latter and the cows "wrongly classified as healthy".

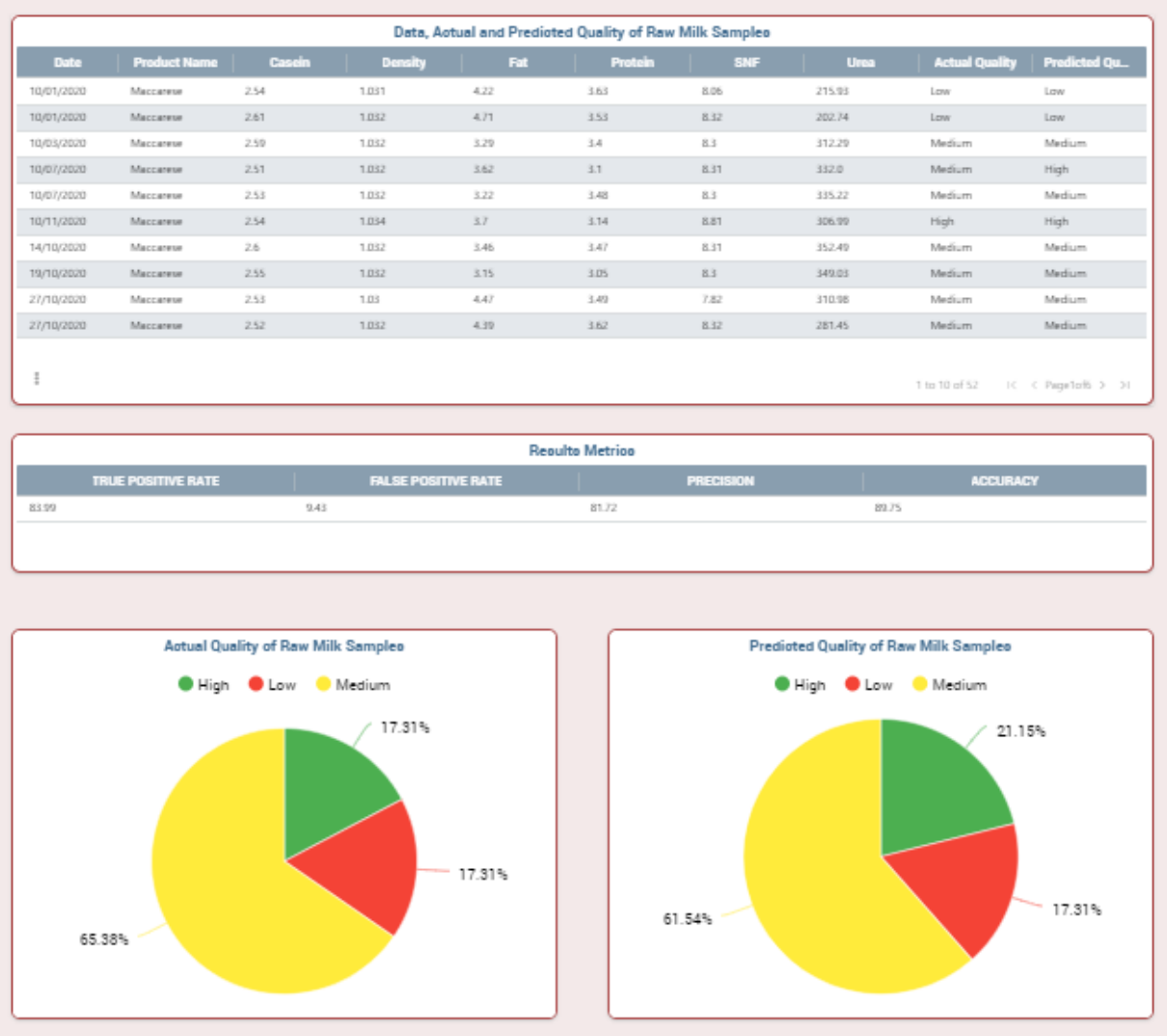

Figure 18: Quality of raw milk samples – Training

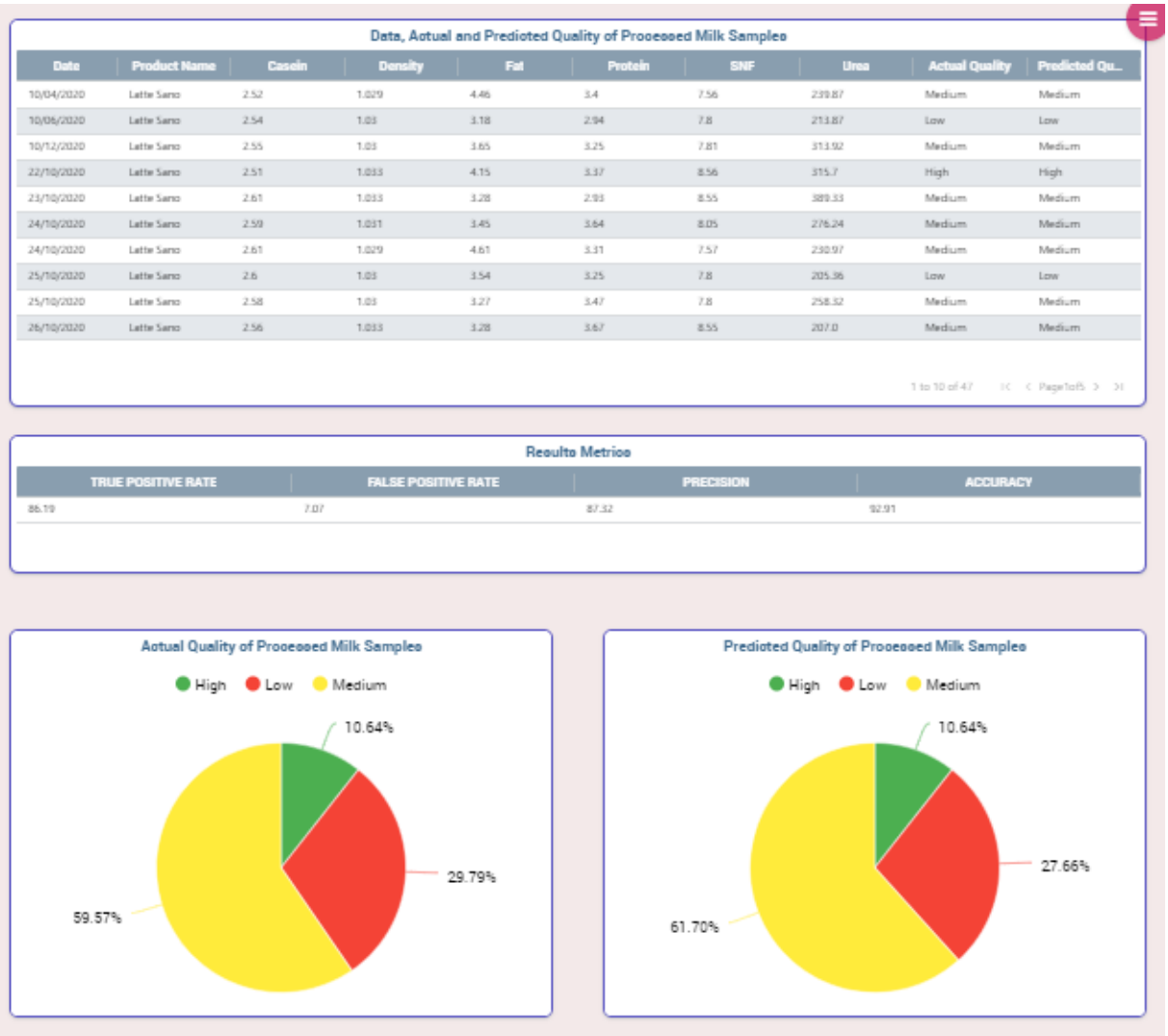

Figure 19: Quality of processed milk samples – Training

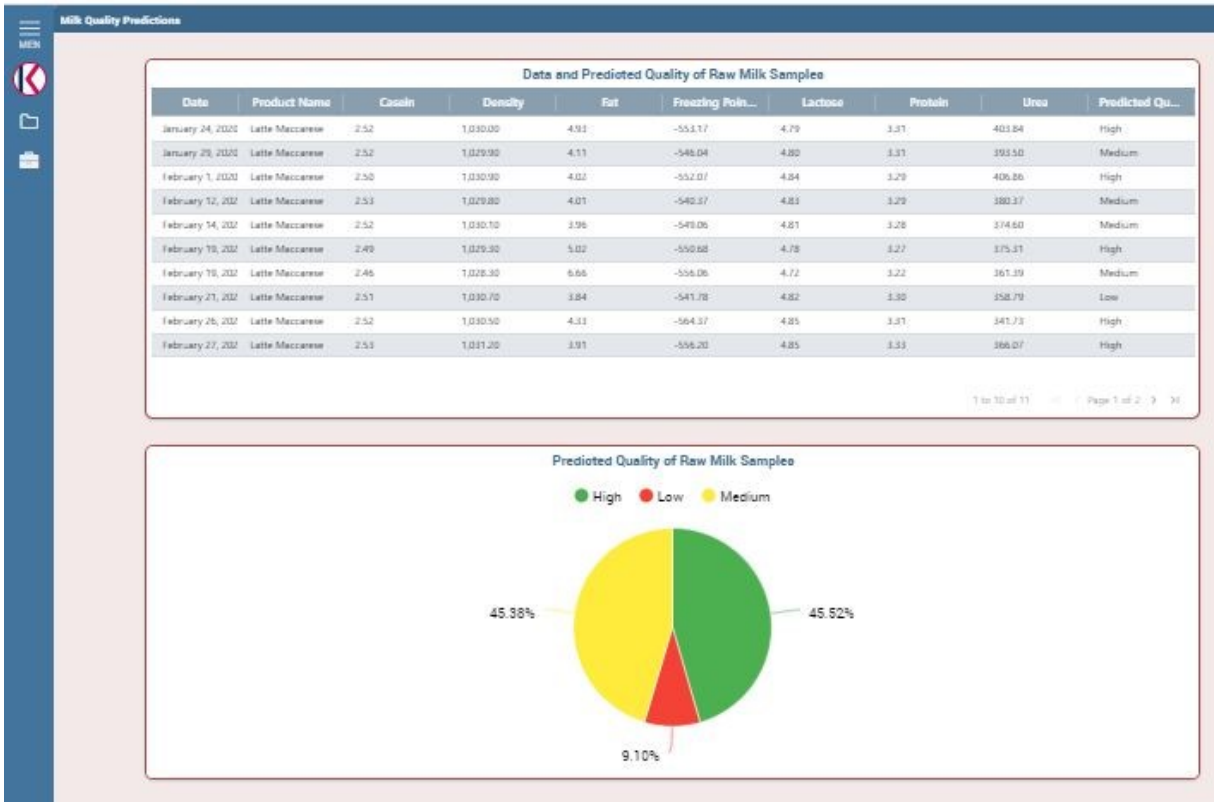

Figure 20: Quality of raw milk samples – Prediction

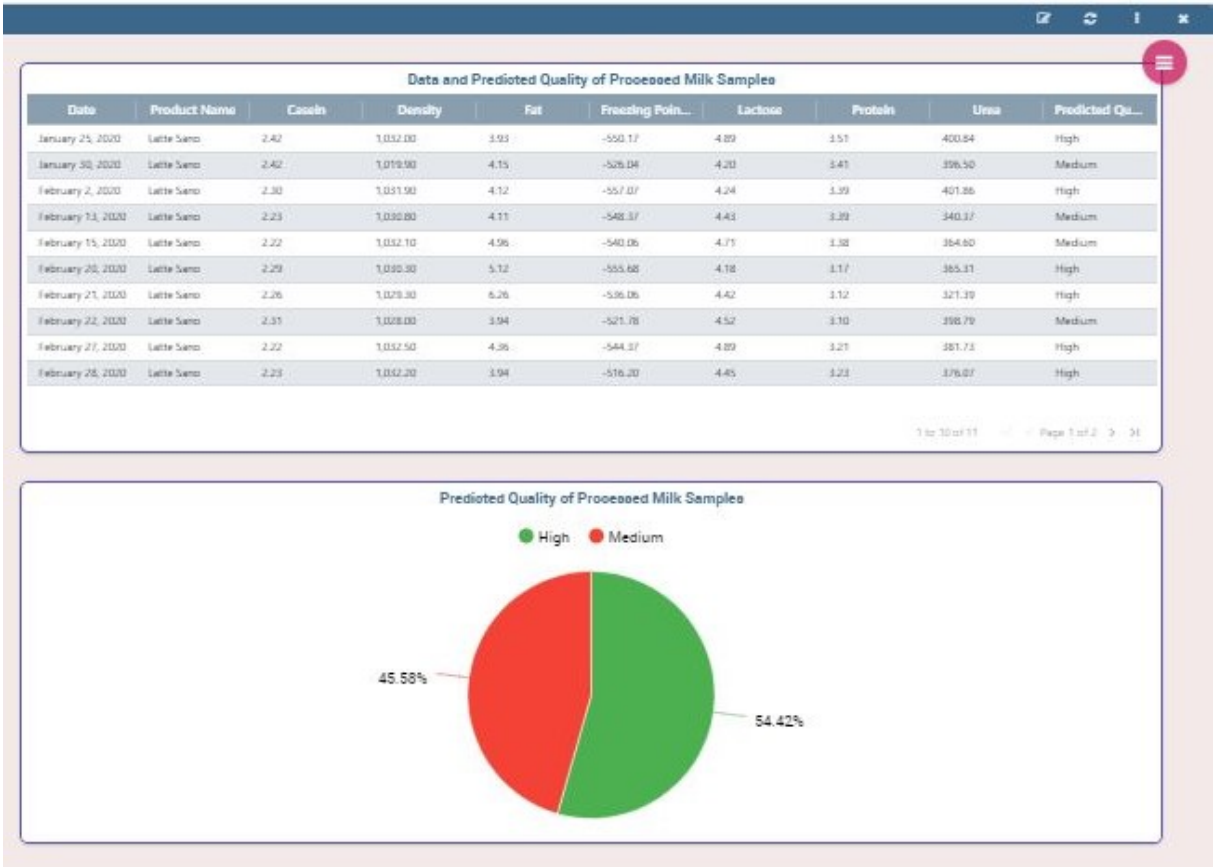

Figure 21: Quality of processed milk samples – Prediction

After obtaining a high degree of accuracy (see Figure 20 and Figure 21 "Predicted" columns), it is possible to see the new predicted values by the algorithm.

By observing the table and the predictions made by the algorithm, it is possible to determine which parameters must be checked to improve the quality of the milk, and this is precisely the aim of the AI: to monitor in order to correct any anomalies that affect the product quality.

7 Progress on Benchmarking and Performance Indicator Monitoring Tools

The DEMETER benchmarking system aims to provide end-users with tools to evaluate the productivity and the sustainability of the practices adopted, as well as the efficacy of the developed digital solutions. The benchmarking components will enable the comparison for individual and peer to peer learning, linked to the impact of operational processes brought by DEMETER.

In D4.1 a preliminary analysis has been carried out along with a survey addressed to pilot leaders with the aim of determining the main objectives of the benchmarking system and the data potentially available at the farm/pilot level for the calculation of indicators to be integrated in the benchmarking components. A first set of performance indicators to evaluate the economic, agronomic, and environmental sustainability of the practises and innovations delivered in the DEMETER pilots has been shared with pilots and reported in D4.1.

Starting from the initial indicators proposed in D4.1, this section reports the **advances and progresses** made during this period regarding benchmarking indicators to be implemented and integrated within the DEMETER DSS Benchmarking components.

The implementation of benchmarking tools and the selection of indicators require a strong cross-Work-Packages' interactions, as described in the following points:

- WP2 - the AIM data format for benchmarking components has been assessed to ensure the interoperability with other DEMETER components.
- WP3 - ensures the integration of the benchmarking components with the DEMETER core enablers.
- WP5 - the selection of indicators has involved the pilot community to assess data availability for their calculation and the alignment with the achievement of pilots' objectives.
- WP7 - a workshop on KPI was organised on March 19, 2021 by WP7/WP5 to define the metrics by which the pilots may measure the success of DEMETER project following a multi-actor and interactive approach. The results of the workshop (which will be described in D5.5) have been taken into account for benchmarking purposes.

7.1 Indicators

Selecting and measuring key performance indicators in the agriculture domain is not an easy task. A critical starting point for developing assessment methods is the selection of representative indicators being sufficiently comprehensive to be applicable in different geographic locations, while sufficiently sensitive to different impacts, crops, and production methods ([7]). The challenge of defining and measuring proper indicators has been undertaken in the implementation of the DEMETER benchmarking system to fulfil objective 04 "*Establish a benchmarking mechanism for agriculture solutions and business, targeting end-goals in terms of productivity and sustainability performance of farms, services, technologies, and practices based on a set of key performance indicators that are relevant to the farming community*". In other words, the target of DEMETER benchmarking system is to create a framework to manage a complex set of indicators able to meet the needs of the 20 DEMETER pilots in evaluating the achievement of pilots' objectives and the success in applying DEMETER technologies.

Therefore, taking into account the differences among the 20 pilots of DEMETER and trying to cover the pilot activities, a minimum set of indicators has been identified and selected addressing the following constraints: i) being concise, ii) easy to calculate and iii) scientifically sound.

According to FAO ([8]), in the process of selecting the possible indicators, a balance must be found between scientific accuracy and pragmatic decision-making. Indicators should be robust (methodological soundness), measurable (easily and clearly quantified), effective (capture or can relate to a broader range of aspects), acceptable (widely accepted, easy to interpret and cost effective).

In addition, as outlined in the KPI workshop of 19 March 2021, the indicators need to satisfy the SMART criteria where SMART stands for:

- **S**pecific: the area of action is clearly defined as well as what the indicators aim to measure - vague definitions which cannot be explained are difficult to measure and can lead to misinterpretation.
- **M**easurable: specify how to calculate the indicator and which data are required.
- **A**chievable: the indicator can be implemented in the benchmarking components allowing farm comparison, given the available data at the farm level.
- **R**elevant: to pilot and DEMETER objectives - the indicator must contribute to measuring the overall success of the delivered technologies.
- **T**ime-bound: an exact end point should be specified.

Hence, the selection process intends to assess indicators which should be available and measurable for most of the DEMETER pilots. The data required for indicator measurements should be gathered at the farm/field level, taking into account data made available by devices (i.e., farm management information system, app to record infield data), automatically recorded (i.e., infield sensors), provided as output by models (i.e., estimated yield, estimated irrigation water amount).

Keeping in mind the DEMETER project objectives (O1, O2, O3, O4), the indicators selected for assessing performance sustainability cover the following three main sectors:

- Agronomic (related to quantity and quality of the production, and the use of input).
- Environmental (related to biodiversity, environmental efficiency, and impact).
- Economic (related to farm profit and technical efficiency).

Each sector has been further divided into sub-sectors, and for each of these, the set of indicators has been kept to a *minimum*, to facilitate the application to the pilots which are focusing on different aspects of the agricultural domain and to meet the constraints of data availability at farm/pilot level.

The process of indicators definition and calculation does not end with this deliverable, as a matter of fact the DEMETER benchmarking framework will allow all the pilots to extend the current indicators list according to their needs, objectives, and data availability.

The selected indicators are to be considered as calculated yearly since the benchmarking components allow the comparison of the indicators on a yearly basis.

In the following paragraphs, a list of indicators for each sector and subsectors is provided and later the indicators have been cross referenced with the FADN database to facilitate the comparison with the European database.

7.1.1 Agronomic Indicators

Agronomic indicators aim to assess crucial elements of agricultural production coping with environmental changes: farm size, yield (levels, variability in time and space), yield quality, input use and animal wellbeing. The yield benchmarking is of particular interest for farmers and advisors since it may provide evidence of the possible improvements that can be obtained in yield or the gap that can be filled if adequate management decisions are undertaken, or if a new technology is applied. The agronomic sector consists of the following sub-sectors:

- **Structural** - indicators describing the farm structure in terms of activities, size, and morphology.
- **Yield - Crop** - describing a crop's quantitative parameters in terms of productivity. Keeping the focus on production, which is the main goal of the farmers, this group is the first informative clue on the activity behaviour and farm trend.
- **Yield - Livestock** - describing livestock's quantitative parameters.
- **Yield - Crop quality** - describing some simple quality parameters of the crops.
- **Yield - Livestock quality** - describing some simple parameters for quality of livestock.
- **Animal welfare** - this sub-sector can include behavioural, physical, physiological and production features.
- **Crop input** - describing the use of external resources and promoting the monitoring of the use of input.

In the following Table 2 we report the list of indicators for each sub-sector and the modality of calculation.

Indicator	Sector	Sub-sector	Calculation mode	Unit of measure
Total Surface	Agronomic	Structural	total surface in hectares (UAA - utilised agricultural area)	ha
Head of Livestock (standard unit)	Agronomic	Structural	number of head in livestock unit (LU)	LU
Head per hectares	Agronomic	Structural	livestock units divided by total area	$LU*ha^{-1}$
Field proximity	Agronomic	Structural	sum of the number of adjacent fields divided by total number of fields	-
Crop Yield	Agronomic	Yield - Crop	average crop yield per field; the actual indicator will be crop-specific (e.g., Olive Yield)	$t*ha^{-1}$
Sugar content	Agronomic	Yield - Crop quality	sugar content divided by mass unit; the actual indicator will be crop-specific (e.g., grape sugar content)	%
Fat content (oil yield)	Agronomic	Yield - Crop quality	fat content divided by mass unit; (e.g., olive fruit fat content)	%
Meat production	Agronomic	Yield - Livestock	meat yield	$kg*animal^{-1}$
Milk Production	Agronomic	Yield - Livestock	milk yield	$kg*cow^{-1}$
Honey production	Agronomic	Yield - Livestock	honey yield	$kg*hive^{-1}$
Milk protein content	Agronomic	Yield - Livestock quality	milk protein content divided by mass unit	%
Milk fat content	Agronomic	Yield - Livestock quality	milk fat content divided by mass unit	%
Animals per square meter	Agronomic	Animal Welfare	number of livestock unit divided by byre area	$LU*m^{-2}$
Somatic cells in milk	Agronomic	Animal Welfare	number of cells per volume (mL) of milk	$n*mL^{-1}$
Animal mortality	Agronomic	Animal Welfare	number of yearly deaths divided by livestock units	%
Irrigation water	Agronomic	Crop Input	total water consumption per area	m^3*ha^{-1}
Nitrogen distribution	Agronomic	Crop Input	total nitrogen input per area	$kg*ha^{-1}$
Number of pesticide treatments	Agronomic	Crop Input	average number of pesticide treatments	n

Table 2: List of selected agronomic indicators with sub-sector, the description of calculation and unit of measure

7.1.2 Environmental Indicators

This first list of environmental indicators has been collected with the aim of addressing relevant sustainability aspects and describing environmental concerns for agricultural production processes. Principal aspects taken into account have been farm biodiversity, the use of water for irrigation, the nutrient management, the use of agrochemicals in pest management:

- **Farm biodiversity** - is related to the biological variety and variability within the farm. Biodiversity is a critical resource and it is typically measured in terms of variation at different levels: genetic, species, ecosystem or landscape. Here, we mainly focus on farm biodiversity.
- **Water Environmental Efficiency** - aims at understanding the environmental impact and the efficiency of farm inputs and explores how these contribute to the production and habitat conservation. Here, we focus on irrigation water, which is explored in terms of efficiency.
- **Nutrient Environmental Efficiency** - aims at understanding the environmental impact and the efficiency of farm inputs and explores how these contribute to the production and habitat conservation. Here, we focus on nitrogen use efficiency.
- **Pesticide Environmental Efficiency** - aims at understanding the environmental impact and the efficiency of farm inputs and explores how these contribute to the production and habitat conservation. This category has a specific focus on pesticide use.
- **Erosion** - assesses the risk of soil erosion by abiotic factors. The indicator gives a useful picture of soil health through the assessment of land degradation.

Indicator	Sector	Sub-sector	Calculation mode	Unit of measure
Field density	Environmental	Farm Biodiversity	number of fields divided by total field area	$n*ha^{-1}$
Crop density	Environmental	Farm Biodiversity	average of the number of fields per type (arable, permanent crop, permanent grassland) on the total field area	$n*ha^{-1}$
Crop rotation period	Environmental	Farm Biodiversity	time interval between two crops in the field	years
Semi-natural surface	Environmental	Farm Biodiversity	semi-natural area divided by total area	%
Water use efficiency[11]	Environmental	Water Environmental Efficiency	yield divided by the irrigation water used	$t*mm^{-1}$
Water environmental efficiency[11]	Environmental	Water Environmental Efficiency	water distributed minus water suggested by models	mm
Nitrogen use efficiency[11]	Environmental	Nutrient Environmental Efficiency	yield divided by the nitrogen distributed	$t*kg^{-1}$
Nitrogen leached[11]	Environmental	Nutrient Environmental Efficiency	potential leachable nitrogen per area	$kg*ha^{-1}$
Nitrogen environmental efficiency[11]		Nutrient Environmental Efficiency	nitrogen distributed minus nutrient suggested by models	$kg*ha^{-1}$

Table 3: List of selected environmental indicators with sub-sector, the description of calculation and unit of measure

[11] Crop specific (e.g., wheat, rice, corn, grape, olive)

7.1.3 Economic Indicators

The indicators of this sector were selected following some key concepts of farm business analysis: profit (difference between the money that comes into the farm business from the sales of a product and the money that goes out to produce it), technical efficiency (measuring the farmer's skill and success in producing the highest possible level of output from a fixed amount of inputs) and economic efficiency (measuring the financial returns on resources used). The following sub sectors have been identified:

- **Farm balance** - includes indicators of costs and incomes related to the whole farm, independently from the activities that have generated them. These indicators are very relevant to the farmers for the evaluation of the economic performance of the whole farm activity.
- **Crop balance** - includes the specific income and costs derived from land cultivation and does not include animal breeding. These indicators inform farmers about the whole crop production sector performance, as well as the economic performance of the main crops.
- **Livestock balance** - similarly to crop balance, this sub-typology of indicators supports farmers in the control of specific incomes and costs derived from the breeding activities, not including crop production. This can help farmers to be informed about the whole breeding sector performance.
- **Input economic efficiency (water)** - indicators of economic efficiency of inputs allow to estimate the effectiveness of the input use in the farm practice. In particular, this sub typology is dedicated to water. In an era of water scarcity, the knowledge of the efficiency of the single water unit may be of great help to the farmers to optimise their irrigation strategy.
- **Input economic efficiency (nutrients)** - economic efficiency of inputs can be measured also for nutrients. The knowledge of the efficiency of the single fertiliser unit, and of its incidence on the total crop production costs, may be of great help to the farmers to optimise their fertilisation strategy.
- **Labour** - labour represents an important cost item for farmers and includes efficiency indicators, which can help farmers to control the labour costs and its efficiency for different activities.
- **Machineries** - machinery economic indicators help farmers to control the costs for machinery, analysing their incidence on the total costs.

Indicator	Sector	Sub-sector	Calculation mode	Unit of measure
Total income	Economic	Farm Balance	sum of gross operating profit, public contributions, sold services and warehouse changes	€
Variable costs	Economic	Farm Balance	sum of farm variable costs	€
Gross operating profit	Economic	Farm Balance	total gross production minus variable costs	€
Total gross production	Economic	Farm Balance	sum of the produced quantity multiplied by the selling price of each farm product	€
Water economic efficiency[12]	Economic	Input Economic Efficiency - Water	total water consumption divided by total gross production	$m^3*€^{-1}$
Fertilisation economic efficiency[12]	Economic	Input Economic Efficiency - Nutrient	total fertiliser costs divided by total gross production	%
Fertilisation distribution costs	Economic	Input Economic Efficiency - Nutrient	sum of total fertiliser costs, labour, and machinery costs of fertilisation	€
Labour cost	Economic	Labour	number of worked hours multiplied by the average salary cost per hour	€
Labour activity index	Economic	Labour	number of worked hours multiplied by the average salary cost per hour divided by the Utilised Agricultural Area (UAA)	$€*ha^{-1}$
Labour activity index - crop production[12]	Economic	Labour	number of worked hours on the crop fields multiplied by the average salary cost per hour divided by the crop area	$€*ha^{-1}$
Labour activity index - animal breeding	Economic	Labour	number of worked hours dedicated to breeding activity multiplied by the average salary cost per hour	€
Labour activity index - livestock unit	Economic	Labour	number of worked hours dedicated to breeding activity multiplied by the average salary cost per hour divided by the number of livestock unit (LU)	$€*LU^{-1}$
Labour activity index - milk production	Economic	Labour	number of worked hours dedicated to dairy activity multiplied by the average salary cost per hour divided by the milk production	$€*t^{-1}$

Table 4: List of selected economic indicators with sub-sector, the description of calculation and unit of measure

The whole list of indicators has been cross-checked with the FADN (Farm Accountancy Data Network) database. The FADN is a database of microeconomic data managed by the European Commission with the main purpose of gathering accountancy data from farms for the determination of incomes and business analysis of agricultural holdings, for statistical and political objectives. FADN relies on annual surveys carried out by the Member States of the European Union. Data is later harmonised. The survey does not cover all the agricultural holdings in the EU but only those due to their economic dimension could be considered commercial. Collected data includes: i) physical and structural data, such as location, crop areas, livestock numbers, labour force, etc, ii) economic and financial

[12] Crop specific (e.g., wheat, rice, corn, grape, olive)

data, such as the value of production of the different crops, stocks, sales and purchases, production costs, assets, liabilities, production quotas and subsidies, including those connected with the application of CAP (Common Agricultural Policy) measures.

In the following Table 5 we report the agronomic and economic indicators previously described, available in the FADN database and the corresponding code.

Indicator	Sector	Sub-sector	Calculation mode	FADN code	FADN description
Total surface	Agronomic	Structural	total surface in hectares	SE025	Total Utilised Agricultural Area
Arable surface	Agronomic	Structural	arable surface in hectares	SE026	Arable land
Permanent crop surface	Agronomic	Structural	permanent crop surface in hectares	SE027	Permanent crops
Grassland surface	Agronomic	Structural	permanent grassland surface in hectares	SE028	Permanent grassland
Cereal surface	Agronomic	Structural	cereal surface in hectares	SE035	Cereals
Vineyard surface	Agronomic	Structural	vineyard surface in hectares	SE050	Vineyards
Forage surface	Agronomic	Structural	forage crops surface in hectares	SE071	Forage crops
Head of livestock (standard unit)	Agronomic	Structural	number of head in livestock unit (LU)	SE080	Total livestock units
Cows	Agronomic	Structural	number of dairy cows	SE085	Dairy cows
Other cattle	Agronomic	Structural	number of other cattle	SE090	Other cattle
Poultry	Agronomic	Structural	number of poultry birds	SE105	Poultry
Wheat yield	Agronomic	Yield - Crop	average wheat yield	SE110	Yield of wheat
Corn yield	Agronomic	Yield - Crop	average corn yield	SE115	Yield of maize
Milk production	Agronomic	Yield - Livestock	milk yield	SE125	Milk yield
Milk per cow	Agronomic	Yield - Livestock	milk yield per dairy cow	SE126	Milk yield cattle dairy cows
Total income	Economic	Farm Balance	sum of gross operating profit, public contributions, sold services and warehouse changes	SE131	Total output
Variable costs	Economic	Farm Balance	sum of farm variable costs	SE281	Total specific costs
Gross operating profit	Economic	Farm Balance	total gross production divided by variable costs	SE410	Gross Farm Income
Total gross production	Economic	Farm Balance	sum of the produced quantity multiplied by the selling price of each farm product	SE135	Total output crops & crop production
Fertilisation distribution costs	Economic	Input Economic Efficiency - Nutrient	sum of total fertiliser costs, labour and machineries costs of fertilisation	SE295	Fertilisers

Indicator	Sector	Sub-sector	Calculation mode	FADN code	FADN description
Labour cost	Economic	Labour	number of worked hours multiplied by the average salary cost per hour	SE370	Wages paid
Machinery cost	Economic	Machineries	sum of the working hours multiplied by the cost per hour of the different machinery	SE340	Machinery & building current costs
Total gross production per hectare	Economic	Crop balance	sum of the product of the total production and selling price divided by the number of hectares of each crop	SE136	Total crops output / ha
Livestock income	Economic	Livestock balance	sum of the total production multiplied by the selling price of each animal product	SE206	Total output livestock & livestock products
Income per livestock unit	Economic	Livestock balance	sum of the total production multiplied by the selling price of each animal product divided by the number of livestock units in the farm	SE207	Total livestock output / LU

Table 5: List of selected economic indicators with sub-sector, the description of calculation and unit of measure

7.2 Benchmarking Components

The benchmarking system has been developed following the specifications already described in the Deliverables 4.1 and 4.2. The system is based on four components:

- I0 - Indicator Engine: the system that performs all the basic routines for indicator management and it is a common component used by the following three components.
- I1 - Generic Farm Comparison: a generic tool usable by all European farms with a minimum set of requested inputs.
- I2 - Neighbour Benchmarking: a tool usable by a group of farmers wishing to share anonymously a set of data to create indicators allowing local benchmark.
- I3 - Technology Benchmarking: a tool helping farmers and stakeholders in evaluating the impact of a technology.

All the benchmarking components have been developed and are available on the DEMETER GitLab repository[13].

The whole system is available as a set of docker containers, allowing its easy installation on premises and its integration with the other DEMETER components. All the components have been integrated in the same Docker Container considering that they are strictly connected and adopt common libraries. In this way it is easier to install, and the same calculated indicators can be used for multiple benchmarking activities.

The benchmarking components are based on the following containers:

- **db**: the container is based on an image with PostgreSQL database, version 11.5 with PostGIS version 2.8; the database will store all the data needed by the components and that needs to be stored (e.g., the average neighbour indicator value); the database has multiple schemas which will be explained in the specific components description.
- **pgadmin**: the database web interface for maintenance and direct access for reporting and advanced analysis.
- **api**: the main container with all the APIs and the data analysis tool.
- **nginx**: the web server proxy.

The benchmarking component interacts with the following DEMETER enablers:

1. **ACS** - Access Control System – providing authentication and authorisation mechanisms; all the benchmarking data, excluding the list of the public indicators definition will be available only for users with a valid OAUTH2 token provided by the DEMETER ACS.
2. **BSE** - Brokerage Service Environment – providing mechanism for the discovery of services.
3. **DEH** - DEMETER Enabler Hub – the benchmarking component has been registered on the DEMETER Enabler Hub.
4. **KnowageForDashboards** - the user interfaces for the benchmarking component have been developed in Task 4.3 and are available in the KnowageForDashboards Enablers.

[13] https://gitlab.com/demeterproject/wp4/benchmarking/

Core Enablers Used

In Figure 22, the whole schema of the technical infrastructure of the benchmarking components is reported.

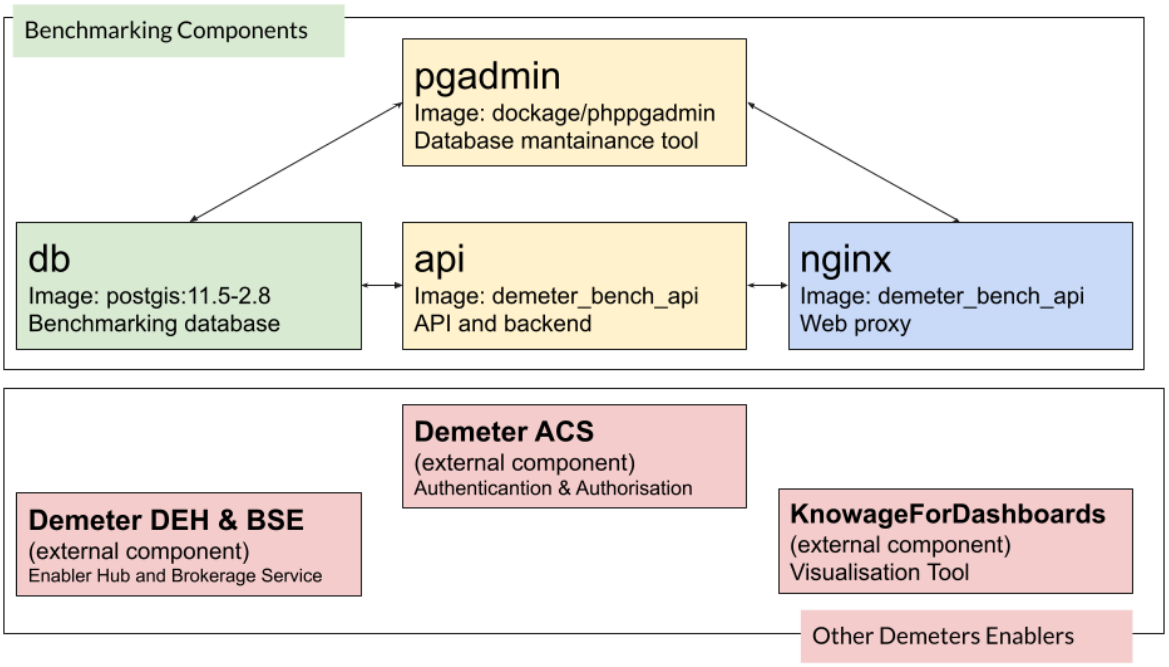

Figure 22: Schema of the DEMETER Benchmarking Technical Infrastructure

To run the docker in the local environment, the user needs to execute the command `docker-compose up -d`. After starting the application, the user can access:

- http://localhost:7050/: the component web interface, which is only for testing purpose. The main user interface has been developed in Task 4.3 and is available in the DEMETER Visualisation tool.
- http://localhost:7050/api/swagger.json: the swagger file that documents all the benchmarking APIs.

In the DEMETER GitLab repository, the detailed installation and deployment instructions on the cloud are available. In the .env variables it is possible to define the needed environmental variables:

- database name, user, password, and port:
  - POSTGRES_DB
  - POSTGRES_USER
  - POSTGRES_PASSWORD
  - POSTGRES_PORT
- API URL and secret key
  - API_URL
  - APP_SECRET_KEY
- DEMETER ACS URL and OAUTH2 parameters
  - REDIRECT_URI
  - IDM_BASE_URL
  - OAUTH2_CLIENT_ID
  - OAUTH2_SECRET

7.2.1 Component I.0: Indicator Engine for Benchmarking Purpose

The Indicator Engine's main target is to manage the indicators to assess the current agronomic, economic, and environmental sustainability with data available at the farm level allowing it to:

- Publish and keep track of the list of indicators.
- Allow pilots integrators to update and extend the list of the indicators following the framework explained in the previous section.
- Store the results of the indicators if the values are needed for benchmarking.

The benchmarking components are based on the DEMETER Agricultural information Model (AIM) and all the input and output variables of the components respect the defined standards. Input and output are formatted in JSON-LD formats.

The main data element of the benchmarking tools is the indicator (**KpiIndicator**), which describes a specific indicator belonging to a specific sector; the formal definition of the element context is available at DEMETER GitLab[14]. The KpiIndicator is a subclass of Semantic Sensor Network Ontology Observable Property concept.

A KpiIndicator is characterised by the following elements:

- @id - unique identifier of an indicator.
- schema.name - English name of the indicator.
- schema.description - description of the indicator.
- sector - generic type of the indicator (Economic, Production, Environmental).
- unit - preferred unit of measure of the indicator.

Follows an example of an indicator in JSON format:

```
{
   "@id": "https://w3id.org/demeter/agri/ext/kpiIndicator#totalEconomicOutout",
   "@type": "KpiIndicator",
   "schema.name": "Economic Output",
   "schema.description": "Economic Output - total value of the production",
   "sector": {
     "@id":    "https://w3id.org/demeter/agri/ext/kpiIndicator#sectorScheme-Economic"
   }
}
```

The indicator section is expandable to accommodate the indicator sub sections as defined by the previous sections on indicators; the sector is defined as a concept using the SKOS vocabulary (Simple Knowledge Organization System Primer). Using these methods, the sector can be expanded adding concept-related sub sections creating an indicators hierarchy.

The actual value of an indicator for a specific farm and a specific main data element of the benchmarking tools is the indicator (**KpiIndicatorValue**); the formal definition of the element context is available at DEMETER GitLab[15]. The KpiIndicatorValue has been created as an expansion of the

[14] https://gitlab.com/demeterproject/wp2/agriculturalinformationmodel/domainspecificontologies/-/blob/master/extensions/jsonld/kpiIndicator-context.jsonld
[15] https://gitlab.com/demeterproject/wp2/agriculturalinformationmodel/domainspecificontologies/-/blob/master/extensions/kpiIndicator.ttl

Semantic Sensor Network Ontology Observation concept already adopted in other DEMETER contexts to express numerical value with spatio-temporal references.

The KpiIndicatorValue contains:

- **hasFeatureOfInterest**: reference to a Feature of Interest (FoI); in the benchmarking contact the feature of interest is a reference to a farm where the indicator is calculated; it is possible also to use the FoI at a more specific scale if the benchmarking user needs to calculate the performance for a specific section of the farm (e.g., a field, or an area where a specific technology is adopted); only the unique identifier of the farm is needed, so that it is not stored other specific type of the information about the farm.
- **hasResult**: reference to the measured value by the indicator; for describing the value:
    - usually, a reference to a QuantityValue element, all the indicators will be quantitative data.
    - unit: the unit of measure of the observations.
    - numericValue: the value of the observed indicator.

- **observedProperty**: the reference to the KpiIndicator.
- **resultTime**: time reference of the indicator observation; in the standard implementation the indicators are calculated on a yearly-based so the visualisation interface will average all the indicators within the same years; but the interface also allows the definition of monthly or daily indicators for specific contexts.
- **referenceValue**: the value of the indicators to be used as a benchmark (e.g., the average olive_yield of my neighbours); usually this data is not uploaded by the user, but it is calculated by the components, averaging all the value of the FoI involved in the same group; the groups definition is described in I2 and I3.

Below is an example of an indicator value in JSON format:

```
{
   "@type": "KpiIndicatorValue",
   "hasFeatureOfInterest": {
        "@id": "urn:demeter:farm1"
   },
   "hasResult": [
     {
        "@type": "QuantityValue",
        "numericValue": 6.67,
        "unit": {
          "@id": "demeter:KilogramPerHectar"
        }
     }
   ],
   "observedProperty": {
     "@id": "https://w3id.org/demeter/agri/ext/kpiIndicator#NitrogerLeached"
   },
   "referenceValue": 6.67,
   "resultTime": "2020-10-01T00:00:00+00:00"
}
```

In the database a dedicated schema has been created, called indicator, to store all the needed values for all the benchmarking components. The storage of the indicators values needs to:

- Calculate the benchmarking values for Neighbour and Technological Benchmarking.
- Store the results to provide to Knowage visualisation tool a REST API needed by Knowage to get the data for the user interface.

At the moment, there are three main tables:

- **indicator**: listing the DEMETER standard indicators (during install all the basic indicators defined in 6.1 are uploaded in the DB) and all the other custom indicators uploaded in the system by the pilots using the API described in next D4.4.
- **foi**: the feature of interest (e.g., farm) related to the indicator value.
- **indicator_value**: the value of the indicator, with the reference to the indicator and the foi.

7.2.2 Component I.1: Generic Farm Comparison

The Generic Farm Benchmarking component has been designed to provide to each farm a set of basic economic indicators, which can be used for a general benchmark of the farm activities. The main objective is to reduce the amount of data required to calculate the indicators. The system is based on the database[16] of Farm Accountancy Data Network (FADN) to have a set of references values to be used to benchmark the farm activities with a set of similar farms belonging to the FADN network.

To perform the analysis, a schema has been created in the Benchmarking database containing the following datasets:

- Spatial reference of the European Administrative division[17]; the geometry allows the extraction of the administrative division and the relative average data from the FADN.
- FADN data extracted from the FADN website.

For FADN, we have created a script that downloads the FADN data using the YEAR.COUNTRY.REGION.SIZ6.TF8 export data formats and import them in the PostgreSQL database.

The FADN file contains the value of about 200 indicators for a group of homogeneous farms. The farm homogeneity is based on:

- YEAR: year of the economic balance.
- COUNTRY_REGION: a set of regions covering all Europe.
- SIZ6: the size of the farm.
- TF8: a description that groups farm together in 8 macro-categories:
    - Fieldcrops.
    - Horticulture.
    - Wine.
    - Other permanent crops.
    - Milk.
    - Other grazing livestock.
    - Granivores.
    - Mixed.

[16] https://ec.europa.eu/agriculture/rica/database/consult_std_reports_en.cfm
[17] https://ec.europa.eu/eurostat/web/gisco/geodata/reference-data/administrative-units-statistical-units/nuts

The algorithm is based on the following steps:

- Acquisition of farm data definition in the AIM data format.
- Extraction of farm location and search of the correspondent FADN regions.
- Extraction from the farm data of a set of basic structural indicators; those indicators provide a generic description of the farm dimensions and typology:
    - SE025 - Total Utilised Agricultural Area.
    - SE026 - Arable land.
    - SE027 - Permanent crops.
    - SE028 - Permanent grassland.
    - SE035 - Cereals.
    - SE050 - Vineyards.
    - SE071 - Forage crops.
    - SE080 - Total livestock units.
    - SE085 - Dairy cows.
    - SE090 - Other cattle.
    - SE105 - Poultry.

- Search, within the specific region, what is the "closest" combination of Size (SIZ6) and typology (TF8) that minimises the distance between the farm structural indicators and the average indicators value; a multidimensional-distance algorithm to search the closest combination has been adopted.
- Extract from the selected DB row a sub-set of indicators (described in the previous section) for the last 10 years; the global parameters (e.g., total input and output) will be shown to farmers allowing them to define their performance according with the average of similar farms in the same area.

We have defined a minimal set of easily available data allowing the farm to get an estimated reference of the economic farm performance indicators: expected output, expected input and expected profit, along with an estimation of the input and output division in general areas. The system does not ask farmers to share the actual farm economic data, the component shows only the reference value, and thus excludes the requirement to enter sensitive information of the farm. Obviously, pilot integrators can grab the reference values and transfer them to their farm accountability system.

The inputs needed are:

- Location of the main farm centre: the farm can decide to share only the administrative division of the farm centre (NUTS3) if it does not want to share the geographic location.
- Farm structure:
    - Average surface of the main crop groups (farmer can also decide to share all the plot geometries; the system will calculate the surfaces).
    - Number of livestock units by species expressed in standard livestock unit (LSU) (if present).

Following an example of the required data in the AIM formats (the JSON is oversimplified just to highlight the main data used by the model) describing a farm, in Tuscany with 2.1 ha of wheat and 5.2 ha of olive.

```
{
   "@type": "Farm",
   "hasGeometry": { "asWKT": "POINT(11.5,42.9)" },
   "containsPlot":[
   {
     "@type": "Plot", "area": 21000, "category": "arable",
     "crop": {"cropSpecies":{  "name": "Wheat",        }      }
   },
   {
     "@type": "Plot", "area": 52000,"category": "permanent",
     "crop": {"cropSpecies":{"name": "Olive" } }
   }, ]
}
```

The component output is a set of relative KpiIndicator and KpiIndicator values using the same formats described in the I0 component. Through the interface (see Figure 23) the user can access the data for the last available years:

- Access a set of indicators; if actual farm data is provided the status shows the differences.
- It is possible to move across years and access historical data to assess the variation in time of the indicators.
- The system shows how the total input and output are divided in sub-categories (e.g., the incidence of fertiliser, or work force in costs, and the crop-related revenues).

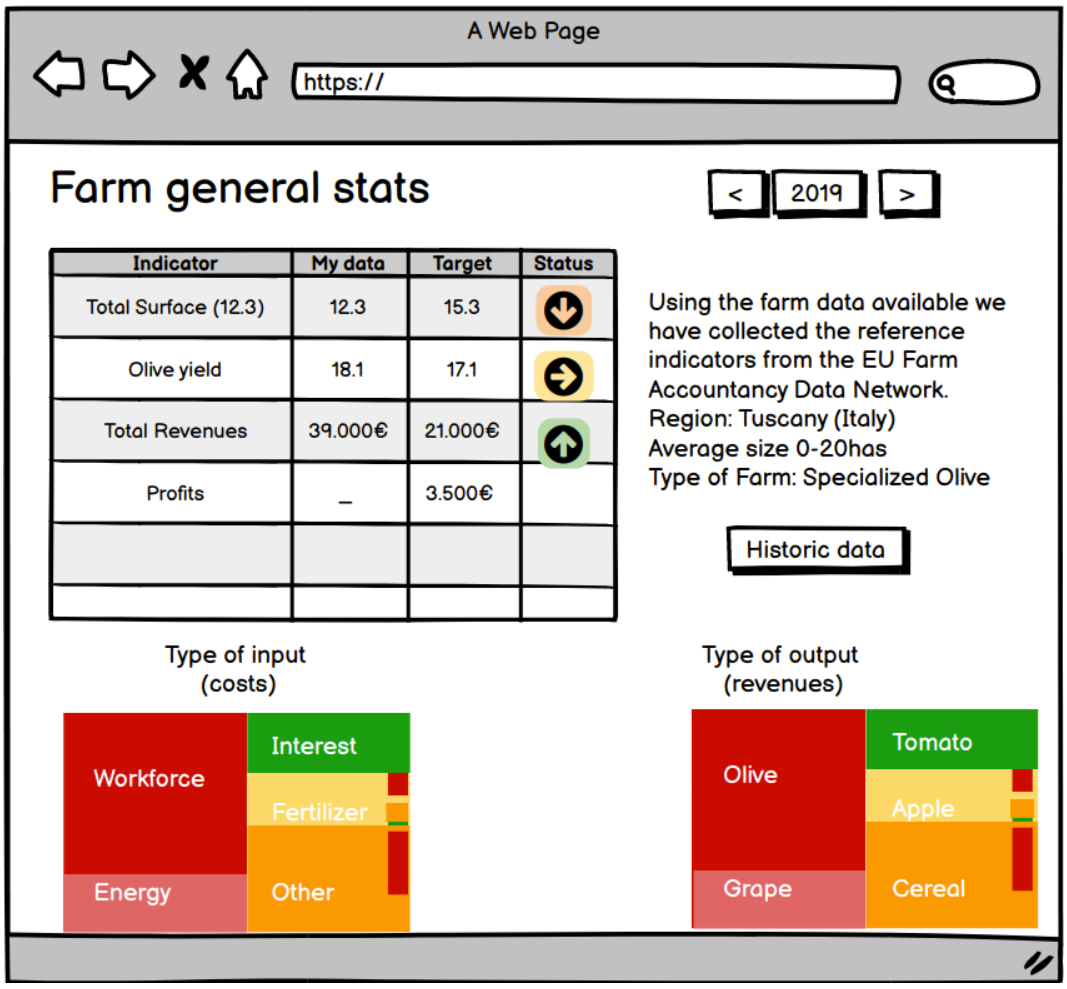

Figure 23: Wireframe of the component I.1 for generic farm comparison

7.2.3 Component I.2: Neighbour Benchmarking

The component's objective is to allow a group of farms (e.g., a DEMETER pilot, cooperatives, consortia, other organisations belonging to a specific area) to share data and compare performance. The methodology to create a neighbour benchmarking tool is the following:

- **Group creation**: a group coordinator (usually a system integrator) can create a group in the Benchmarking system; the creator is the owner of the group; several groups can be created, and a group has the following features:
    - Group name.
    - Mode: the group can be open (all valid users can push their own data) or closed (only a set of users can participate at the benchmarking).
    - Users: an array of the email of the participants (if the group is closed).
    - Indicators: an array of the indicators associated with that group; if the array is empty all the system indicators can be collected; the indicator should be added to the system (if needed) using the I0 REST API.
    - Reference_method: the coordinator can choose how the reference value will be calculated; the following options are available:
        - Average: average of the group values; default value.
        - Median: median of the group values.
        - Top 25 percentile: top 25 percentiles.

- **Farm association**: a user with the valid credential can associate the farm to the right group; the system checks if the association can be done (according with group features) and accepts the farm in the group; the system produces a guide for that specific farm allowing data entry and the benchmarking.
- **Data entry**: the user can push in the farm space a new set of indicators. The system has an UPSERT method of data entry meaning that it will use the indicator id and the time reference as identifiers, if an indicator value already exists for that farm the system updates the current value.
- **Benchmarking**: the farm can access with their specific URL the result of the indicators showing how the farm performs compared to other farms in the group.

In Knowage, the farm can access a dashboard (see Figure 24) to navigate through Benchmarking results:

- In the first page a summary shows the average benchmarking results for the three main sectors of the indicators: Agronomic, Economic and Environmental.
- It is then possible to analyse the result of the specific indicators.
- For multi-year data it is possible to visualise how the performance varies over time.

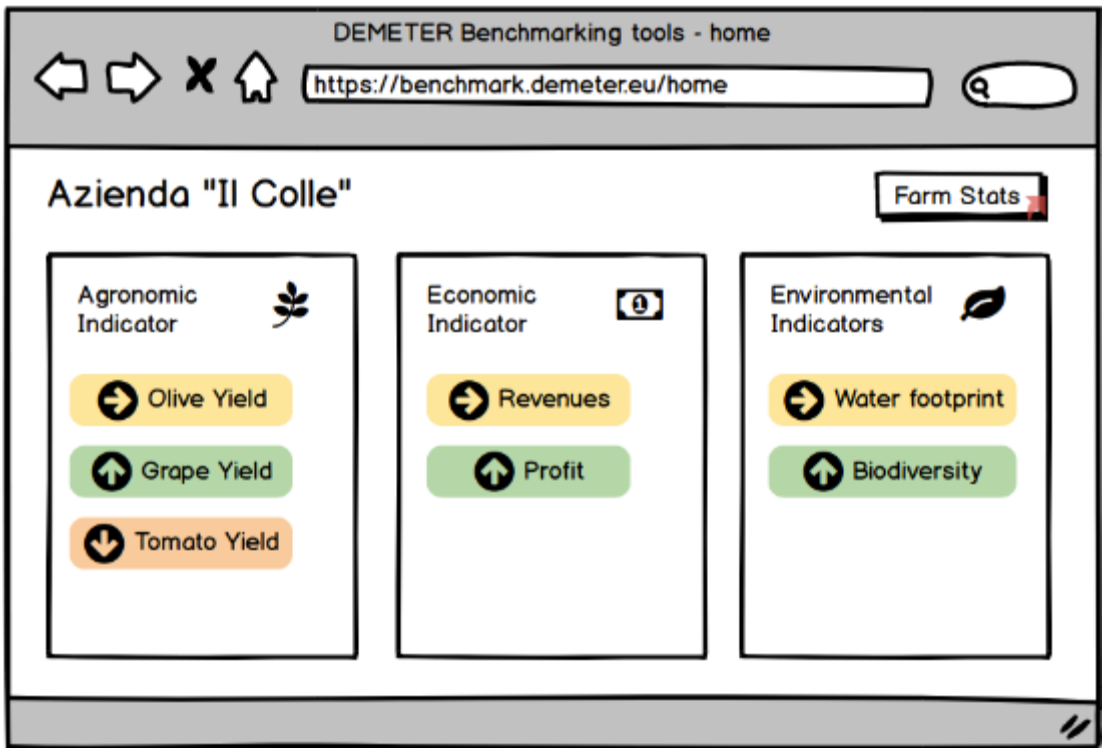

Figure 24: Wireframe of the component I.2 for neighbouring benchmarking

7.2.4 Component I.3: Technology Benchmarking

This component's target is to support the general DEMETER Objective 3: "*Establish a benchmarking mechanism for agriculture solutions and business, targeting end-goals in terms of productivity and sustainability performance of farms, services, technologies, and practices based on a set of key performance indicators that are relevant to the farming community*".

The component has two potential uses:

- A reusable component allowing a farmer or a group of farmers to evaluate the performance of a technology from the agronomic, economic, and environmental point of view.
- Use the developed component as a benchmarking mechanism for DEMETER technology, based on data collected by the farms of the several pilots to validate the achievement of the DEMETER project KPIs.

The methodology to create a benchmarking for a specific technology is the following:

- **Technology creation**: to test a specific technology a coordinator should create a new entry in the benchmarking system, allowing a group of farmers to enter data about the use of the technology; some mechanism will be inherited by the neighbour group (e.g. open/close mode, define a list of participating users, indicator definition, etc…) with some specific information:
    - technology_type: the type of technology (e.g., sensor, a digital service, a Decision Support Systems, etc…).
    - the level of adoption: for the comparison it will be created at least two distinct sub-groups:
        - adopter: data collected in farm using the technology.
        - non-adopter: data collected in farm not using the technology.

- the coordinator can create also more groups (e.g., partial adopters); it is important to stress that a single farm can participate in more than 1 group (e.g., adopt a technology from a specific year or test the technology on a part of the farm).

- **Data entry**: a user with valid credentials can push a new set of indicators into the technology space; the indicators need to be inserted in a specific level of adoption of the technology; each user will have a separate space and have no access to other users' data. The system has an UPSERT method of data entry meaning that it will use the indicator ID and the time reference as identifiers if an indicator value already exists for that technology and level the system updates the current value.
- **Benchmarking**: the coordinator and all the users belonging to the group can access the summary of the results; the individual data is not shown, the users access only the averages related to each adoption level (see Figure 25).

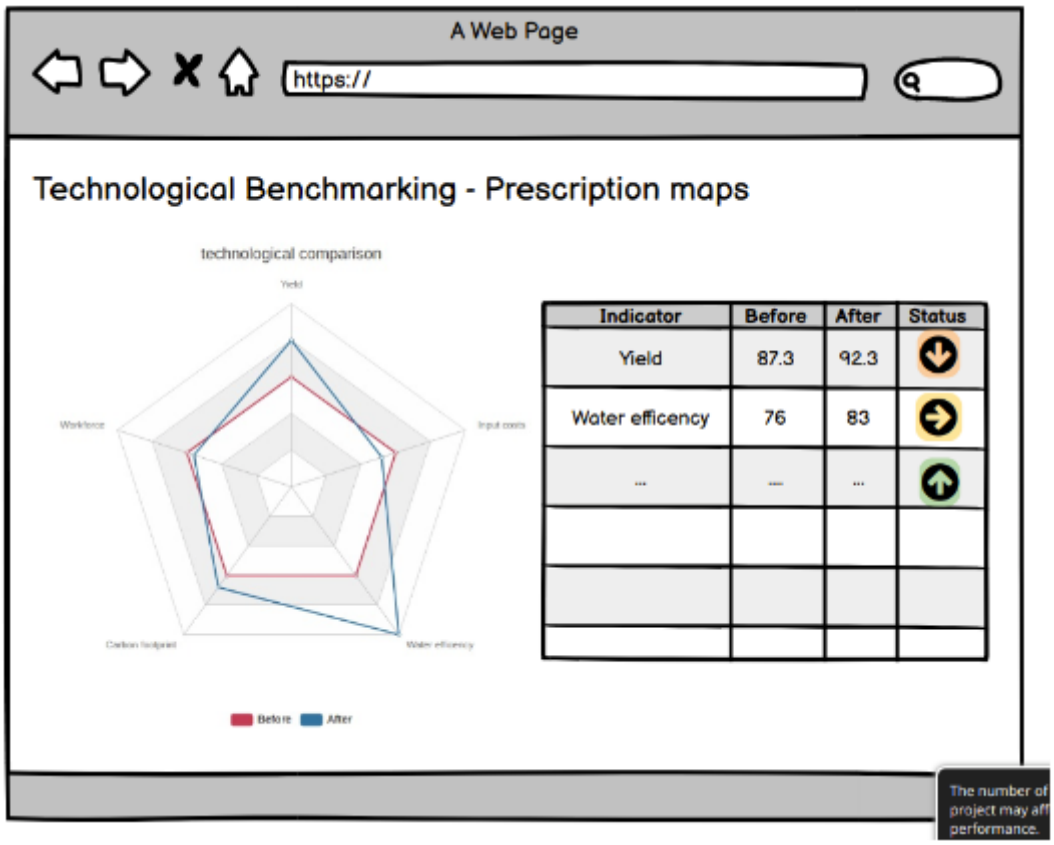

Figure 25: Wireframe of the component I.3 for technology benchmarking

8 Conclusions

This document reports on the progress achieved for two core building blocks of the DEMETER Decision Support, namely the Benchmarking and AI-based Decision Support Tools:

- On one hand, the progress achieved in the areas for artificial intelligence technologies. This is related to the AI-based Decision-Making models being implemented and integrated within the different DEMETER DSS components.
- On the other hand, the implementation of the benchmarking system of DEMETER. The DEMETER benchmarking system provides tools to evaluate the productivity and the sustainability of the practices adopted, and the efficacy of the delivered digital solutions. These tools will allow the comparison of individual and neighbouring farms and for the impact of operational processes brought by DEMETER.

The AI modules developed, adapted and/or deployed will help in solving the needs from the different pilots at the same time that they are integrated with the DSS components. These AI modules have been developed as generic as possible to allow their application in other pilots' sites whenever possible.

Considering the three pillars of sustainability in agriculture (agronomic, economic and environmental), a set of indicators has been defined and will be integrated in the benchmarking components with the aims to support stakeholders in evaluating the productivity and the sustainability of the practices and to assess the efficacy of the digital solution.

Further developments will be taking place as the pilots are providing validation feedback based on their field experience. The details of the validation and the updated description of both AI modules, KPIs and Benchmarking will be documented in the Deliverable 4.5 "Final Release and Support Report for Decision Enablers, Advisory Support Tools and DEMETER Stakeholder Open Collaboration Space" due to be released M38 (October 2022).

9 References

[1] Gallagher, N. (2018). Whittaker Smoother. Description of the Whittaker smoother with weighting to allow local smoothing.

[2] VITO (2019). CropSAR guarantees continuous crop monitoring. On-line - https://vito.be/en/news/cropsar-guarantees-continuous-crop-monitoring.

[3] Sanz-Cortés F., Martinez-Calvo J., Badenes M. L., Bleiholder H., Hack H., Llacer G., Meier, U. (2002). Phenological growth stages of olive trees (Olea europaea). *Annals of Applied Biology*. 140 (2): 151–157. doi:10.1111/j.1744-7348.2002.tb00167.x. ISSN 1744-7348.

[4] Allen C. (1976). A modified sine wave method for calculating degree days. Environmental Entomology, 5, 388-396.

[5] Oses N., Azpiroz I., Marchi S., Guidotti D., Quartulli M., Olaizola I. G. Analysis of Copernicus' ERA5 Climate Reanalysis Data as a Replacement for Weather Station Temperature Measurements in Machine Learning Models for Olive Phenology Phase Prediction. *Sensors*. 2020; 20(21):6381. doi:10.3390/s20216381.

[6] Azpiroz I., Oses N., Quartulli M., Olaizola I. G., Guidotti D., Marchi S. Comparison of Climate Reanalysis and Remote-Sensing Data for Predicting Olive Phenology through Machine-Learning Methods. *Remote Sensing*. 2021; 13(6):1224. doi:10.3390/rs13061224.

[7] Neset T. S., Wiréhn L., Opach T., Glaas E., Linnér B. O. (2019). Evaluation of indicators for agricultural vulnerability to climate change: The case of Swedish agriculture. Ecological Indicators, 105, 571-580

[8] Femia A., Hass J., Lumicisi A., Romeiro A. (2017). Agri-Environmental Statistics and Indicators: A Literature Review and Key Agri/Environmental Indicators; Global Strategy Technical Report, No. 27; Food and Agriculture Organization of the United Nations: Rome, Italy.

[9] Sadeghi M., Babaeian E., Trenton E. F., Jones S., Tuller M. (2017). The optical trapezoid model (OPTRAM): A novel approach to remote sensing of soil moisture applied to Sentinel-2 and Landsat-8 observations. Remote Sensing of Environment. 198. 52-68. 10.1016/j.rse.2017.05.041.

[10] FAO (1990). Expert consultation on revision of FAO methodologies for crop water requirements. Annex V, Penman-Monteith Formula. www.fao.org/3/x0490e/x0490e06.htm.

Annex A ML and AI libraries

This Annex introduces some of the ML and AI libraries used for the different modules developed for the DSS components. This introduction is preceded by defining the types of datasets and the metrics used to evaluate the performance of the models selected and trained.

A.1. General statements

A.1.1. Data types' definition

For AI and ML algorithms to be correctly executed and provide meaningful outcome a set of input data is needed for them to learn how to behave. However, since we want to have as independent model as possible the input data needs to be divided across different data sets to ensure the independence of the model, i.e., with a low bias, and avoid the overfitting of those models. As such, the input data will be split largely across three types of data: training, testing and validation data.

The model is initially fit on a **training** dataset, which is a set of examples used to fit the parameters of the model. In practice, the training dataset often consists of a (big) subset of the input dataset. After the adjustment of the model, especially during supervised learning, the current fitted model is run with a second (not-so-big) subset of the input dataset, called **validation** dataset. The validation dataset provides an unbiased evaluation of a model fit on the training dataset while tuning the model's parameters.

Finally, the **test** dataset is the third (also-not-so-big) subset of the input dataset which is used to provide an unbiased evaluation of a final model fit on the training dataset.

A.1.2. Metrics

The metrics used are calculated as follows:

True Positive Rate indicates how many true positive values have been identified compared to the sum of both true positive and false negative values:

$$\frac{True\_Positive}{True\_Positive\ + False\_Negative}$$

False Positive Rate indicates how many false positive values have been identified compared to the sum of both false positive and true negative values:

$$\frac{False\_Positive}{False\_Positive\ + True\_Negative}$$

Precision indicates how many true positive values have been identified compared to the sum of both the true and false positive values:

$$\frac{True\_Positive}{True\_Positive\ + False\_Positive}$$

A.2. Random Forest

Random Forest[18] is one of the most relevant and effective algorithms for the different topics addressed in DEMETER in general and the DSS in particular. As its name indicates, this method combines many decision trees into a single model. Individually, the predictions made by the decision trees may not be accurate, but combined together, the predictions will, on average, be closer to the outcome. The final result returned by the Random Forest is nothing more than an average of the numerical results returned by the different trees in the case of a regression problem, or the class returned by the largest number of trees in the case in which Random Forest is used to solve a classification problem.

To understand better how this method works, it is necessary to distinguish two operations of the algorithm: **training** and **prediction**. During training, each tree in a Random Forest learns from a random sample of data points. The idea is that by training each tree on different samples, although each tree may present a high variance with respect to a particular set of training data, overall, the entire forest will have a lower variance, but not at the cost of increasing the distortion. In practice, the Random Forest combines hundreds or thousands of decision trees, each training on a slightly different set of observations. Furthermore, during training, all historical data is given as input to the model along with the data relevant to the domain of the problem and the real value that we want the model to learn to predict. The model learns any relationships between the data, known as *features*, and the values to predict, called *target*. The training is carried out until the prediction results, highlighted in the metrics, are satisfactory. If the training dataset produces an unsatisfactory prediction, then it is necessary to either retrieve more representative data or change the **features** of the dataset. Once the suitable training model has been identified, it will be possible to move on to the prediction without training: the algorithm, based on the last training model, will make its predictions on other data, having the same structure as the training dataset, therefore the same fields' features, but with different values. Much as humans learn from example, the decision tree also learns through experience, excepts it does not have any previous knowledge it can incorporate into the problem. After enough training with quality data, the decision tree will far surpass our prediction abilities. By processing the training flow, the Random Forest algorithm learns under what circumstances to assign a classification. In fact, in addition to the characteristics to be analysed in the flow, there will also be a "manually" enhanced column, containing the real classification. This column is the reference for the algorithm, to understand how to classify the characteristics; it will verify, at the end of processing, how many evaluations actually correspond to reality and how many are wrong, producing a statistic. This statistic allows us to understand if the training flow is adequate or not: if the number of evaluations is accurate, it means that the processed dataset contains good quality information; conversely, if the number of evaluations is not accurate, it will be necessary to have more precise data to obtain a more accurate evaluation. Once a good level of training has been reached, the algorithm can be subjected to the prediction phase; in this phase it will elaborate a flow structured like the training flow, in which the column concerning the "manual" evaluation will be empty, since the algorithm will be able to autonomously predict the classification to be assigned. Machine learning at the beginning of the classification process does not seem to give the desired results, but it must be remembered that the more selective the features and quality data are during the training phase, the better the learning process will be and the more realistic the forecast (prediction), which will meet the desired results.

[18] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

A.3. Neural Networks

Neural networks, also known as artificial neural networks (ANNs) or simulated neural networks (SNNs), are a subset of ML and are at the heart of deep learning algorithms. Their name and structure are inspired by the human brain, mimicking the way that biological neurons signal to one another.

Neural networks are a set of algorithms that are designed to recognise patterns. They interpret sensory data through a kind of machine perception, labelling or clustering raw input. The patterns they recognise are numerical, contained in vectors, into which all real-world data, be it images, sound, text, or time series, must be translated. Neural networks can adapt to changing input; so, the net generates the best possible result without needing to redesign the output criteria.

Neural networks help us cluster and classify. You can think of them as a clustering and classification layer on top of the data you store and manage. They help to group unlabelled data according to similarities among the example inputs, and they classify data when they have a labelled dataset to train on. Like any ML algorithm, Neural networks can also extract features that are fed to other algorithms for clustering and classification; so, you can think of deep neural networks as components of larger ML applications involving algorithms for reinforcement learning, classification, and regression.

A.4. Linear Regression

Linear regression is a basic and widely used type of predictive analysis. The overall idea of regression is to examine two things:

- Does a set of predictor variables do a good job in predicting an outcome (dependent) variable?
- Which variables in particular are significant predictors of the outcome variable, and in what way do they–indicated by the magnitude and sign of the beta estimates–impact the outcome variable?

These regression estimates are used to explain the relationship between one dependent variable and one or more independent variables. The simplest form of the regression equation with one dependent and one independent variable is defined by the formula:

$$y = c + b * x$$

Where:

- y = estimated dependent variable score,
- c = constant,
- b = regression coefficient,
- x = score on the independent variable.

Three major uses for regression analysis are:

- First, the regression might be used to identify the strength of the effect that the independent variable(s) have on a dependent variable.
- Second, it can be used to forecast effects or impact of changes. That is, the regression analysis helps us to understand how much the dependent variable changes with a change in one or more independent variables.
- Third, regression analysis predicts trends and future values. The regression analysis can be used to get point estimates.

A.5. Multilayer Perceptron

A multilayer perceptron (MLP) is a class of feedforward artificial neural network. A MLP consists of at least three layers of nodes: an input layer, a hidden layer, and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. MLP utilises a supervised learning technique called backpropagation for training. Its multiple layers and non-linear activation distinguish MLP from a linear perceptron. It can distinguish data that is not linearly separable.

The multilayer perceptron is the *hello world* of Deep Learning: a good place to start when you are learning about deep learning. Multilayer Perceptron is commonly used in simple regression problems. However, MLPs are not ideal for processing patterns with sequential and multidimensional data.

They are composed of an input layer to receive the signal, an output layer that makes a decision or prediction about the input, and in between those two, an arbitrary number of hidden layers that are the true computational engine of the MLP. MLPs with one hidden layer are capable of approximating any continuous function.

MLPs are often applied to supervised learning problems: they train on a set of input-output pairs and learn to model the correlation (or dependencies) between those inputs and outputs. Training involves adjusting the parameters, or the weights and biases, of the model to minimise error. Backpropagation is used to make those weight and bias adjustments relative to the error, and the error itself can be measured in a variety of ways, including by root mean squared error (RMSE).