

Machine Learning approaches for Ocean Colour

Dr. Ana Ruescas^{1,2} & Dr. Gustau Camps-Valls²
^{1,2}Brockmann Consult & IPL Universitat de València

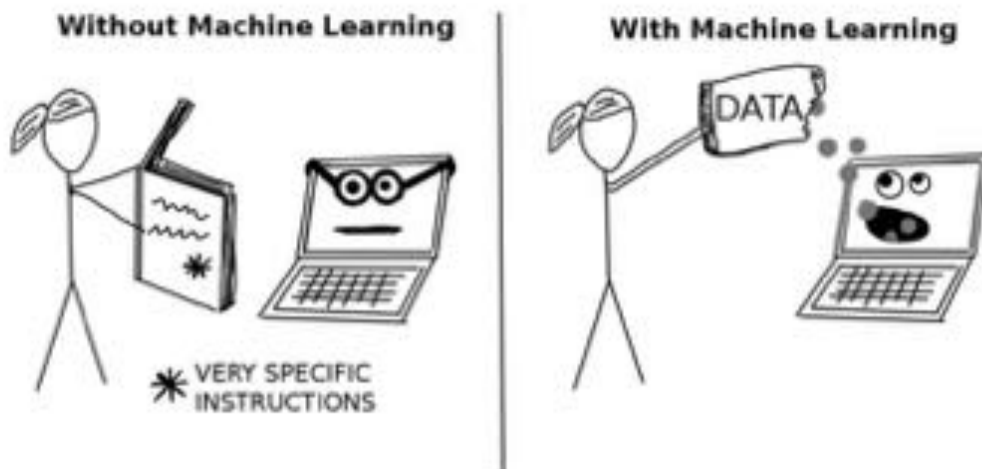
IOCCG Summer Lecture Series 2024





- What is machine learning?
- Image processing chains
- Approaches for:
 - Feature extraction
 - Classification
 - Regression
- Conclusion
- Hands-on: CDOM estimation

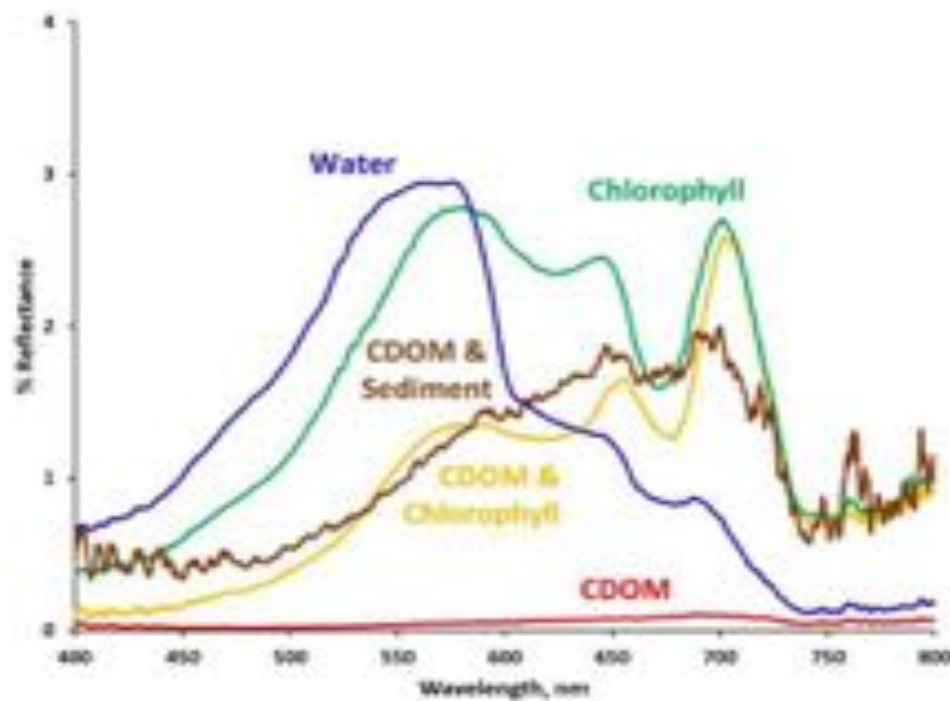
- Machine learning (ML) is a set of methods that computers use to make and improve predictions or behaviours based on data.
- This short pptx will focus on supervised ML: we have a dataset for which we already know the outcome of interest and want to learn to predict the outcome for new data.
 - If the output is categorical, the task is called classification.
 - If the output is numerical, it is called regression.
- Clustering tasks (unsupervised learning) or reinforcement learning are not the object here.



[C. Molnar, Interpretable Machine Learning, 2021]

Molnar, C. (2022). Interpretable Machine Learning: A Guide for Making Black Box Models Explainable (2nd ed.). christophm.github.io/interpretable-ml-book/

- Different materials produce different electromagnetic radiation spectra
- The spectrum shows absorptions and emissions at different wavelengths
- The spectral information contained in an image pixel can therefore indicate the various components in the water



[Olmanson et al., 2016]

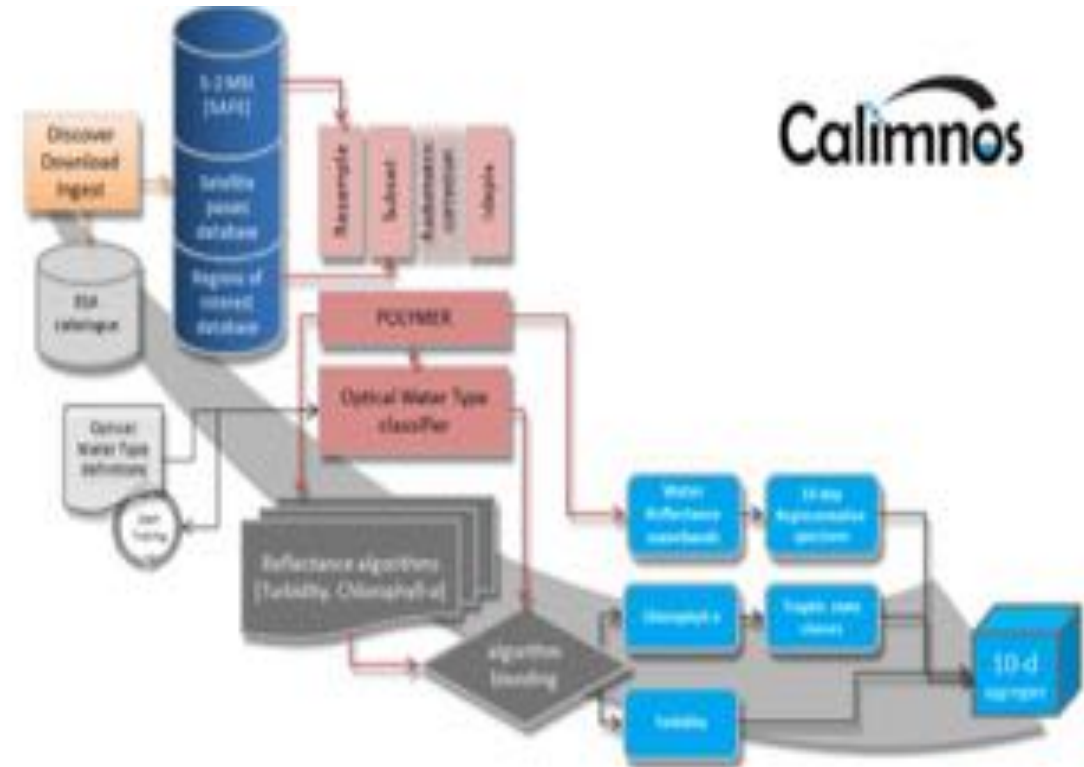


[Ocean Optics Web Book]



Standard processing chain

- Many steps and by-products from signal/image acquisition to the final products.
- A wide diversity of problems and dedicated tools



Calimnos

[Stolzer et al., 2020]

- How to select the best features that describe the problem
- Extract the best combination of spectral bands
- Automatically find groups of pixels in the image for screening, detection...
- Estimate geo-physical variables from the spectra (e.g. chl-a)
- Assign semantic classes to objects/regions in the scene (e.g. phytoplankton types, water classes)
- High dimensional data: multi-temporal, multi-angular and multi-source
- Non-linear, non-Gaussian feature relations
- Few supervised (labelled) information is available (high cost)
- Tons of data to process in (near) real time

Try to find a subset of the input variables (also called features or attributes).

Extracting features from RS images is essential to:

- Compress information for storage/transmission
- Reduce spatial and spectral redundancy
- Make image processing algorithms more robust (to noise, dims.)
- Visualize data characteristics
- Understand the underlying physical relations

Extracted features can be either:

1) Spectral:

- Physically based: erosion, dilation, filters → **not explained here**
- Statistical multivariate methods: linear and non-linear

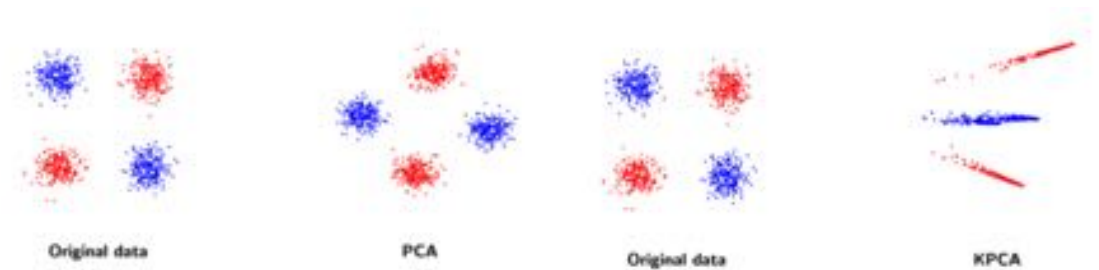
2) Spatial/contextual :

- Standard image processing descriptor → **not explained here**

- Measured spectral signal at the sensor depends on the illumination, the atmosphere, and the surface observed
- Physically-inspired features before applying a machine learning algorithm
- Adapt standard feature extraction methods, such as PCA, to include knowledge about the physical problem

Statistical multivariate methods:

- Dimensionality reduction is sometimes essential before classification or regression
- Most of the spectral feature extractors are based on multivariate analysis: “project data onto a subspace that maximize explained variance, minimize correlation, minimize error, etc.”
- Linear (PCAs) and non-linear methods (KPCAs)





- Find features maximizing the variance of the data
- The Python code:

PCA transformation

```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
prinComp = pca.fit_transform(input_data)
prinDf = pd.DataFrame(data = prinComp,
                      columns = ['PCA_1', 'PCA_2'])
```

Advantages and disadvantages:

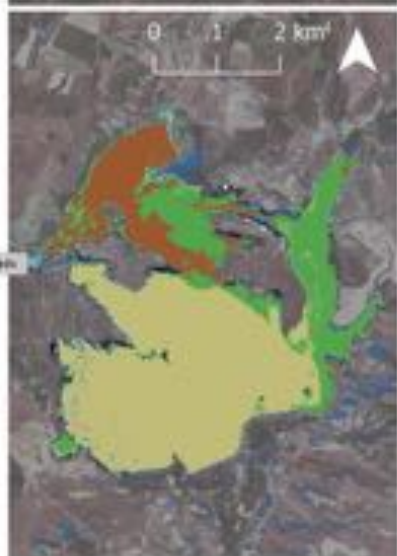
- ✓ Simplicity
- ✓ Easy to understand
- ✗ Unsuitable for non-linear problems (kPCAs?)
- ✗ More dimensions than points?

Kernel methods in machine learning:
<https://isp.uv.es/courses/>



Classification problems

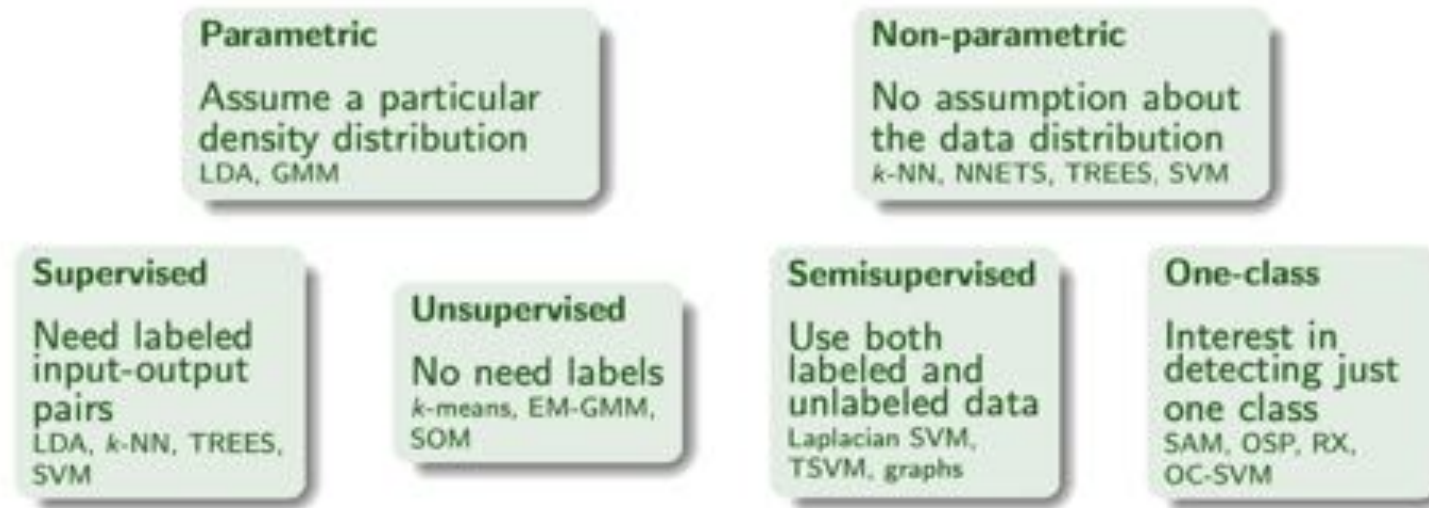
How to pass from here...



...to here?

Challenging problem!

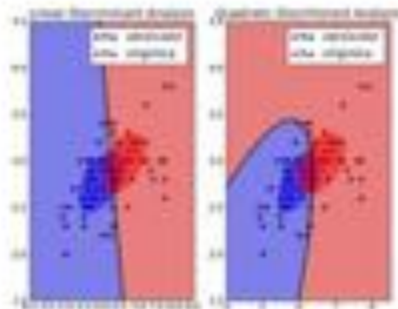
- What is a class? How many classes in the scene?
- What is "turbid" water?
- Do I have enough labelled data?
- Can my classification model be
- generalized?
- Labelled data is expensive
- Different atmospheric and illumination effects on several scenes
- Do I need atmospheric correction?
- Expert knowledge is needed pre- and postprocessing



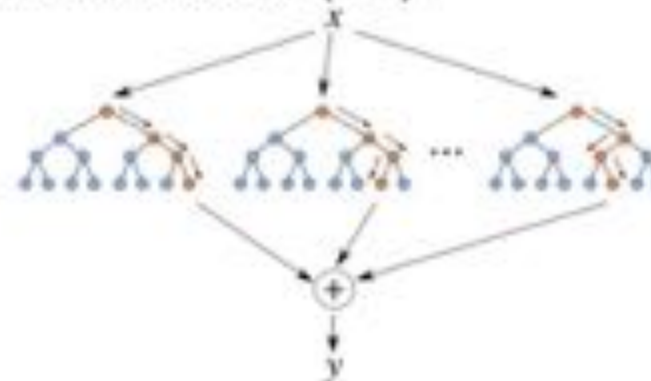
- Not too much success in parametric classifiers, as some assumptions break
- Currently, non-parametric classifiers and committees of experts excel
- k -NN shows a good compromise between accuracy and computational cost
- Support vector machines (SVM) typically outperform the rest

The easiest way to achieve interpretability is to use only a subset of algorithms that create interpretable models.

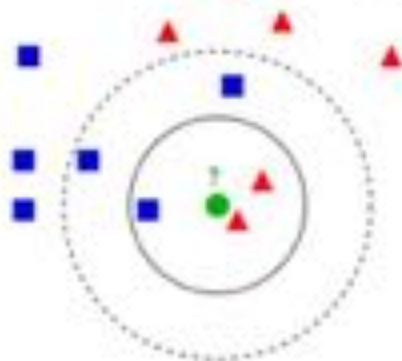
- Linear discriminant analysis (LDA)



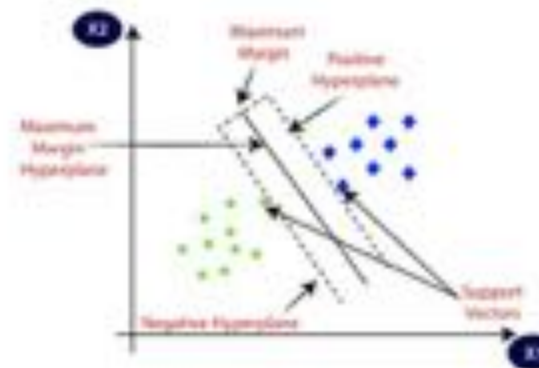
- Random Forest (RF)



- k Nearest Neighbor (k -NN)



- Support Vector Machines (SVM)

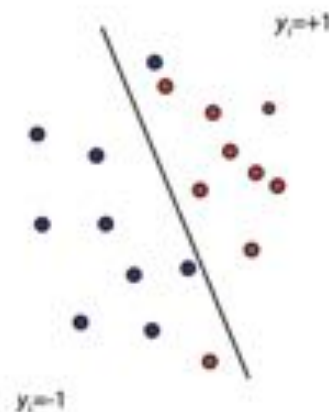




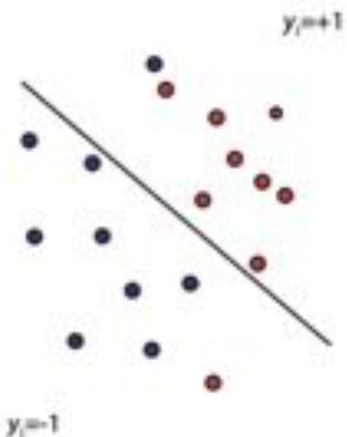
The linear case in classification problems



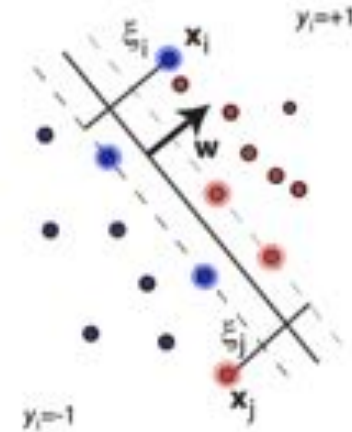
Input data



Another solution



One solution



Optimal case

Linear Discriminant Analysis

```

from sklearn.datasets import make_classification
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
# define dataset
X, y = make_classification(n_samples=1000, n_features=10, n_informative
=10, n_redundant=0, random_state=1)
# define model
model = LinearDiscriminantAnalysis()
# fit model
model.fit(X, y)
# define new data
row = [0.12777556, -3.64400522, -2.23268854, -1.82114386,
1.75466361, 0.1243966, 1.03397657, 2.35822076,
1.01001752, 0.56768485]
# make a prediction
yhat = model.predict([row])
# summarize prediction
print('Predicted_Class: %d' % yhat)

```

Machine Learning Mastering



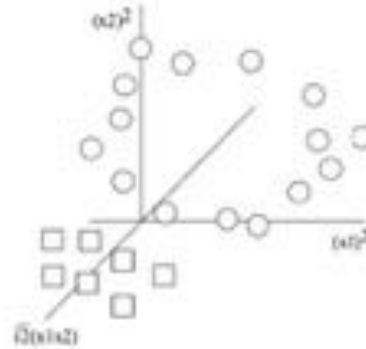
The non-linear case in classification problems

Original space \mathcal{X}

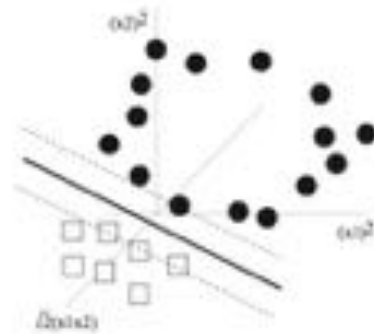


→ Project →

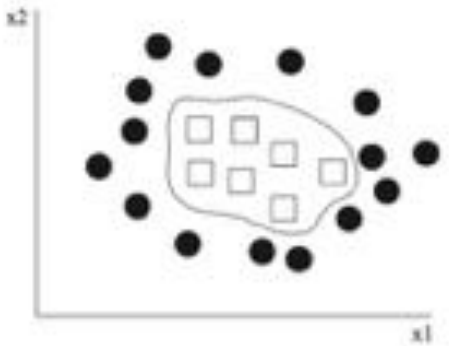
Feature space \mathcal{H}



Use linear model



← Back in \mathcal{X} ←



Support Vector Machines (SVM): “non-parametric kernel method that fits an optimal linear hyperplane separating the classes in a higher dimensional representation (feature) space”



Not enough labelled data

- The image statistics can be misinterpreted because there are not enough representative pixels per class.
- We can include information from unlabelled samples in what is known as semi-supervised learning.
- For instance using **Bagged kernels**

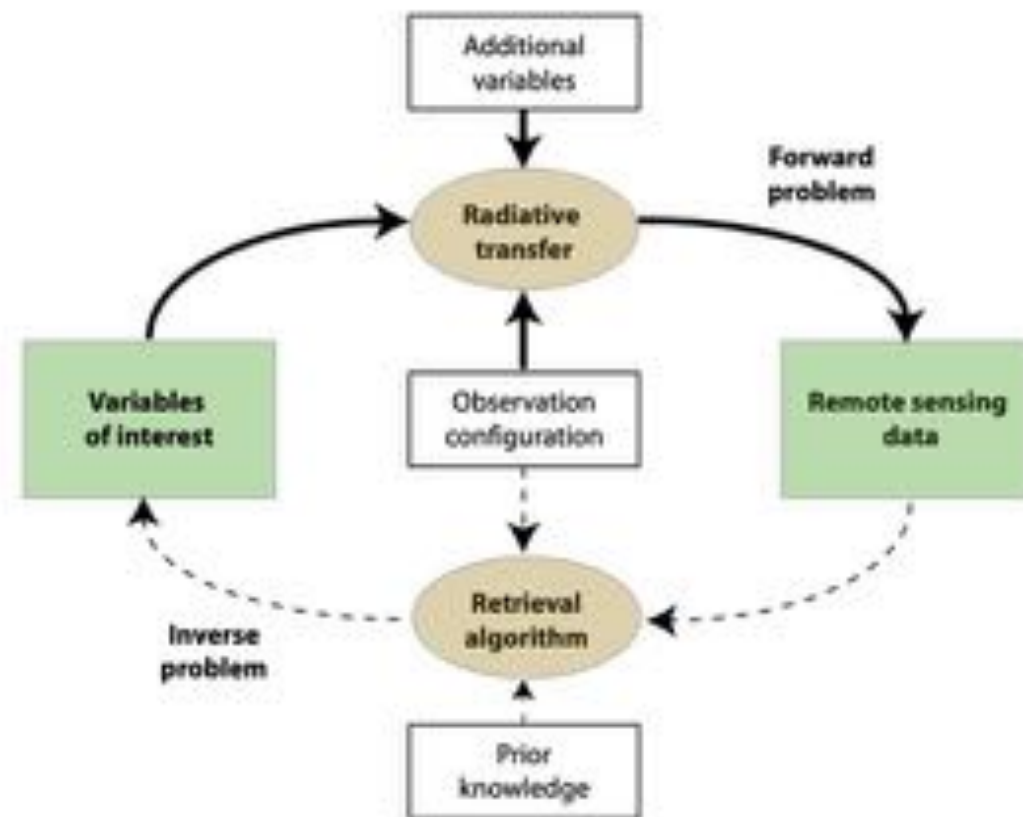




Regression problems

The objective here is to transform measurements into biophysical parameter estimates with EO data.

- Forward modelling: simulate a database of pairs of reflectance spectra + parameters with RTMs
- Inverse modelling: numerical/statistical invert models from RS data to estimate the parameters by designing algorithms that, starting from radiance, can give estimates of the variables of interest





- Three main families of methods:

1) **Statistical inversion models:** parametric and non-parametric

- Parametric models rely on physical knowledge of the problem and build explicit parameterized expressions that relate a few spectral channels with the bio-geo-physical parameter(s) of interest.
- Non-parametric models are adjusted to predict a variable of interest using a training dataset of input-output data pairs.

2) **Physical inversion models:** try to reverse RTMs

- After generating input-output (parameter-radiance) datasets, the problem reduces to, given new spectra, searching for similar spectra in the dataset and assigning the most plausible ('closest') parameter.

3) **Hybrid models** try to combine the previous approaches.

Two main approaches:

- Parametric regression inversion models: assume an explicit model for retrieval, e.g. discrete band approaches like indices, band ratios...
- Non-parametric regression: do not assume explicit feature relations

Linear non-parametric models

Stepwise multiple linear regression (SMLR)

Partial least squares regression (PLSR)

Ridge regression (RR)

Least Absolute Shrinkage and Selection Operator (LASSO)

Non-linear non-parametric models

Decision trees, bagging and random forests

Neural networks

Kernel methods: SVR, RVM, KRR, GPR

Bayesian networks

Weaknesses

- Makes only poorly use of the available information within the spectral observation; at most a spectral subset is used. Therefore, they tend to be more noise-sensitive as compared to full-spectrum methods
- Parametric regression puts boundary conditions at level of chosen bands, formulations and regression function.
- Statistical function accounts for one variable at the time.
- A limited portability to different measurement conditions or sensor characteristics.
- No uncertainty estimates are provided. Hence the quality of the output maps remain unknown.

Strengths

- Simple and comprehensive regression models; little knowledge of user required.
- Computationally inexpensive.
- Fast in processing.



Weaknesses

- Training can be computational expensive.
- Can create over-complex models that do not generalize well (overfitting).
- Expert knowledge required, e.g. for tuning. However, toolboxes exist that automate some steps.
- Some regressors behave rather unstable when applied to data that deviate from statistically different from those used for training.
- Most of them act as a black box.

Strengths

- Can make use of all bands, full spectrum.
- Build advanced, adaptive (nonlinear) models.
- Some methods cope well with redundancy and noisy data.
- Once trained, fast processing images.
- Some of them (e.g. NN, decision trees) can be trained with high numbers of samples.
- Some methods provide insight in model development (e.g. GPR: relevant bands; decision trees: model structure).
- Some methods provide uncertainty intervals (e.g. GPR, KRR).



How to measure the goodness of a model?

Given two variables y_i and \hat{y}_i , $i = 1, \dots, N$

- Error (residuals): $e_i = y_i - \hat{y}_i$
- Bias: mean error (ME): $ME = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)$
- Accuracy: $RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2}$
- Goodness-of-fit: Pearson's correlation coefficient

In Python:

```
from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error
print("r2=", r2_score(ytest, ypred))
print("MAE =", mean_absolute_error(ytest, ypred))
print("RMSE =", mean_squared_error(ytest, ypred))
```

What is Gaussian Process (GP):

- Gaussian Processes are a generalization of the Gaussian probability distribution.
- It is a probability density over functions, non-linear and non-parametric.
- GPR is still a form of supervised learning.
- A Gaussian process is a Gaussian random function, and is fully specified by a mean function $m(x)$ and covariance function $k(x, x)$. This covariance function is called the latent function or the “nuisance” function.
- They are a type of kernel model and they are capable of predicting highly calibrated class membership probabilities
- The hyperparameters of the GPR algorithm on a given dataset can be tuned.

Deep Learning, DL (it is all about scale)

- Deep Learning is a subfield of ML concerned with algorithms inspired by the structure and function of the brain called artificial neural networks.
- Multi-output regression involves predicting two or more numerical variables.
- Deep learning neural networks are an example of an algorithm that natively supports multi-output regression problems.
- Neural network models for multi-output regression tasks can be easily defined and evaluated using the Keras deep learning library.
- What is deep learning?

- Biophysical parameter estimation is perhaps the most important (and challenging) problem in remote sensing
- Traditional methods were focused on simplistic approaches using only a few spectral bands
- New regression-based approaches alleviate the problems by exploiting the wealth of spectral and auxiliary information
- The common approaches consider:
 - Empirical models (e.g. indices) are easy, fast but too general
 - Physical radiative transfer models are flexible but slow and require specific information (e.g. aerosols, geometry) which is not always available
 - Non-parametric regression may offer a robust alternative that can be easily implemented in operative processing chains



- A major disadvantage of using machine learning is that insights about the data and the task the machine solves is hidden in increasingly complex models.
- The higher the interpretability of a machine learning model, the easier it is for someone to comprehend why certain decisions or predictions have been made. Miller (2017), "Interpretability is the degree to which a human can understand the cause of a decision". Another one is: "Interpretability is the degree to which a human can consistently predict the model's result".
- New era started with Explainable AI (XAI)

Useful and very interesting links

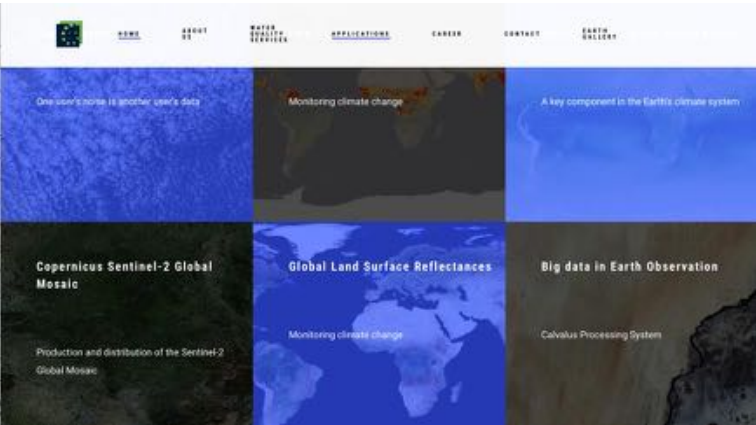










Image and Signal Processing Group, Universitat de València, <https://isp.uv.es/code/>

Brockmann Consult <https://www.brockmann-consult.de/>

-  **Regression, system identification and function approximation**
-  **Classification, change and anomaly detection**
-  **Feature extraction, dimensionality reduction and manifold learning**
-  **Image and video processing**
-  **Remote sensing applications**
-  **Causal inference**
-  **Cross-Sensor Adversarial Domain Adaptation of Landsat-8 and Proba-V images for Cloud Detection**
-  **Multitemporal Cloud Masking in the Google Earth Engine: validation results**

Bringing Machine Learning to Earth Observation Training

EUMETSAT Liege Colloquium, Liege, 05.2023

A massive open online course showcasing AI/ML in a cloud environment

INTRODUCTION: Why a MOOC?

- EUMETSAT EDW¹, Mercator Ocean International, and the European Environment Agency provide environmental data as entrusted entities to the Copernicus programme.
- Working with this data is challenging with users ever more in need of the resources offered through cloud services, and analysis techniques offered by Artificial Intelligence and machine learning approaches.
- A Massive Open Online Course (MOOC) was designed to support users in exploring the use of Copernicus data for thematic applications using AI/ML approaches.

MIXED MEDIA APPROACH TO HANDS-ON LEARNING WITH HOSTED JUPYTER NOTEBOOKS

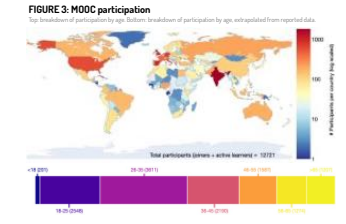
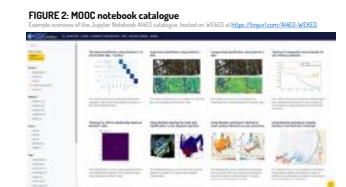
- The MOOC provided video interviews with experts, tests, image galleries and links to further information.
- Each week also included practical examples in the form of Jupyter Notebooks with recorded video run-throughs showing AI/ML workflows.
- Notebooks were based on tested learning design principles with clear explanation and appropriate obfuscation of reusable functions.
- Participants interacted with experts via course fora, weekly round up videos, and through WEKED helpdesk.

OUTCOMES

- Since opening, the course has attracted over 10,000 registrations from across age groups, countries, and sectors of society.
- Material is reused in training by organisers and participants.

LESSONS LEARNED:

- Example workflows increase utility of the course.
- Sufficient cloud resources must be available for concurrent users.
- Balance must be found between levels of content (introductory vs advanced/real-life workflows).
- Showing training of models is hard (need a lot of resource and hosted data in memory), demonstrating application is easier.
- Facilitating contribution of AI/ML workflows to open source repositories could expand what was done in the MOOC, as the research around AI/ML grows.



In more detail...

TOPIC 4: MONITORING THE OCEANS:

- The Marine specific section of the MOOC includes:
 - Topic 4a: Monitoring the Ocean** - an overview video of what we can learn about the oceans from space in situ measurements, and modeling.
 - Topic 4b: Tracking Ships** - a video and Jupyter Notebook on how AI and ML can identify, classify and track movement of ships from space.
 - Topic 4c: Marine Safety** - a video on how AI/ML techniques are helping identify marine debris.
 - Topic 4d: Monitoring Marine Life** - a video on how AI is helping to monitor fish, people, and coral.
 - Topic 4e: Expanding Our Vision of the Sea Surface** - a video on the use of remote sensing for measurement of different variables.
 - Topic 4f: Using ML to Differentiate Between Sediment and Chlorophyll** - a video and Jupyter Notebook on the use of water quality parameters.
 - Topic 4g: Using ML to combine water quality data from different satellites** - a video and Jupyter Notebook on combining data from two different satellites using ML methods.
 - Topic 4h: Marine Analysis for Wilder Users** - a video on the Marine Analyst platform.

INFRASTRUCTURE USE

The live run of the MOOC (lasting 6 weeks) drove increased usage of the WEKED JupyterHub, shown by consistently increased numbers of concurrent users. Further users cloned the notebooks from GITHUB to deploy on their own computer systems.

MORE INFORMATION:

Future Learn - MOOC <https://www.futurelearn.com/courses/ai-ml-for-earth-monitoring>

WEKED <https://www.wiked.eu/>

Jupyter <https://jupyter.org/>

MOOC notebooks <https://github.com/eumetsat/copernicus-training>

Hayley Evers-King¹, Neil Fletcher¹, Ravi Kapur², Ben Loveday³, Julia Wagemann⁴, Joana Miguens¹, Federico Fierli¹, Chris Stewart⁵, Fabrice Messal⁶, Mark Higgins¹

¹EUMETSAT, Darmstadt, Germany
²Imperial Space, London, UK
³meteo.fr, Darmstadt, Germany
⁴NEED Srl, Ferrara, Italy
⁵European Centre for Medium-range Weather Forecasting (ECMWF), Bonn, Germany
⁶Mercator Ocean International, Toulouse, France

Take a picture for a copy of this poster

Or contact us at: ops@eumetsat.int, copernicus.training@eumetsat.int

EUMETSAT | PROGRAMME OF THE EUROPEAN UNION | copernicus | IMPLEMENTED BY



Machine Learning Regression for Ocean Parameters retrieval

Regression approaches for ocean parameter retrievals.

This repository contains the code used in [Ruescas et al. \(2018\)](#). Please consult the paper for more technical information on the methodology.

Contents

The folder contains three notebooks:

1. [Load_and_visualize_data](#) reads the dataset and performs the training of the models.
2. [Statistics_analysis](#) analyses the results of the models using metrics and plots to understand the accuracy and uncertainties.
3. [WQ_predict_image](#) applies the models to OLCI L2 images, focused on the C2RCC products.

In addition, you will find the following helper functions in the "ml" folder:

1. [data_load_53.py](#) loads the data set for training and testing.
2. [models.py](#); the core of the model.
3. [ml_regression.py](#) train and save the proposed models with the different combination of bands.
4. [c2_processing.py](#) loads a Sentinel-3 image and apply a trained model to the data.
5. [Py4R_plots.py](#) with function for plotting.

OLCI images are stored in the Support_data folder. This data is retrived in the third notebook.

+ Code

+ Markdown

<< Index

Load and visualise data >>

[View on GitHub](#) | [EUMETSAT Training](#) | [Contact helpdesk for support](#) | [Contact our training team to collaborate on and reuse this material](#)



Thank you!
Questions are welcome.