CONSORZIO INTERUNIVERSITARIO PER LA FISICA SPAZIALE

UNIVERSITA' DEGLI STUDI DELL'AQUILA

IL MIO VOLER CERCAR OLTRE LA META

INTERNATIONAL SCHOOL OF SPACE SCIENCE

L'Aquila - ITALY

Web School on: Dynamical Systems and Machine Learning Approaches to Sun-Earth Relations

Machine Learning Methods in Space Weather

Cristina Campi

campi@dima.unige.it

Dipartimento di Matematica, Università di Genova

Università di Genova

The image part with relationship ID rId2 was not found in the file.

Machine Learning algorithms

Applications to space weather forecasts

What is a solar flare?

“A solar flare is a flash of brightness on the surface of the sun. During a flare a huge amount of energy (up to a billion hydrogen bombs) is released, heating the solar plasma, accelerating particles and propagating into space.”

(http://flarecast.eu)

Why we are interested in predicting them?

A flare may cause:

- radio blackouts after 8 minutes;
- a solar radiation storm after 20 minutes;
- a geomagnetic storm after 2-4 days.

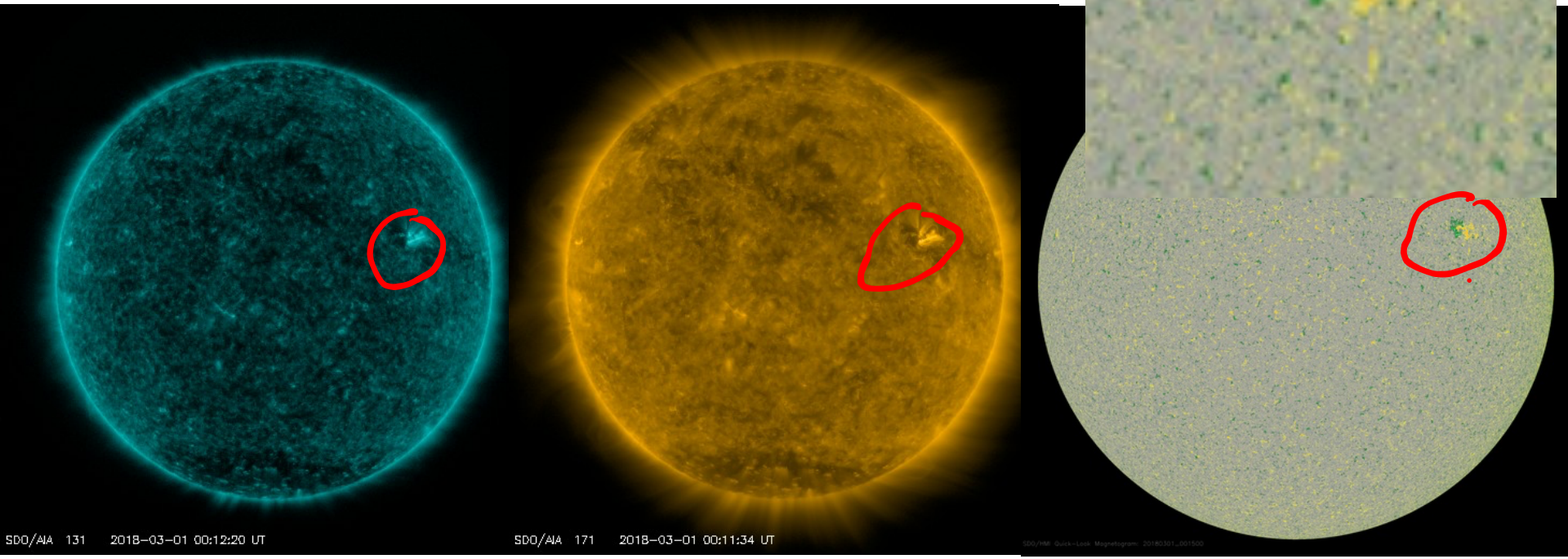
Solar flares can affect satellite operations, aviation and communication technologies, transportation networks, pipelines, and power grids on the ground.

Reliable space weather predictions are vital for taking mitigation measures in time.

A hint of solar physics

Flares eject cloud of accelerated particles in the outer space due to the reconnection of magnetic lines of force on the Sun surface.

Sun regions where magnetic fields are stronger -on average- are known as active regions.



Data are measured by Helioseismic and Magnetic Imager (HMI) aboard the Solar Dynamics Observatory (SDO)

https://sdo.gsfc.nasa.gov

The data – an example

- The data are a collection of vectors;
- Each vector represents a point-in-time: a specific configuration - with temporal info (time-sample in the time domain) and space info (active region id on the sun);
- Each vector entry represents a feature (or property) extracted using specific algorithms;
- Data from the past: we know if a are occurred w.r.t. each point-in-time we consider: we can label the vectors (0 no flare, 1 flare);
- Using these labelled data we want to predict, as soon as we have a new point-in-time, if a flare will occur.

“extracted using specific algorithms”

This step is not necessary for machine learning.

You can use the whole image with some machine learning algorithms.

Why can be beneficial to extract features and then process them?

Supervised machine learning

Support Vector Machine, decision trees, neural networks, linear and logistic regression...

- We divide the data in a training set and a test set;
- We train the algorithms on the training set using the labels;
- Once we have a trained model, we apply it to the test set and check its performances (again, thank to the labels).

How do we check the performances?

Confusion matrix

		ACTUAL CLASS	
		0	1
PREDICTED CLASS	0	TRUE NEGATIVES (TN)	FALSE NEGATIVES (FN)
	1	FALSE POSITIVES (FP)	TRUE POSITIVES (TP)

Forecast validation

Bloomfield D. S. et al, The Astrophysical Journal Letters, 2012

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

Not so good in the case of “rare” events

1000 points-in-time, 1 of them represents a devastating flare, the other 999 are no-flare event.

We classify correctly all the TNs and misclassify the “big one”:

TN = 999, TP = 0, TP + TN + FP + FN = 1000

Accuracy = 0.999

Other common scores

Heidke skill score (HSS):

$$\mathrm{HSS} = \frac{2[(\mathrm{TP}\ \mathrm{TN}) - (\mathrm{FP}\ \mathrm{FN})]}{(\mathrm{TP} + \mathrm{FN})(\mathrm{FN} + \mathrm{TN}) + (\mathrm{TP} + \mathrm{FP})(\mathrm{FP} + \mathrm{TN})}$$

True skill statistic (TSS) or Youden's index:

$$\mathrm{TSS} = \frac{\mathrm{TP}}{\mathrm{TP+FN}} - \frac{\mathrm{FP}}{\mathrm{FP+TN}} = \frac{\mathrm{TP}}{\mathrm{TP+FN}} + \frac{\mathrm{TN}}{\mathrm{TP+FN}} - 1$$

Critical Success Index (CSI):

$$\mathrm{CSI} = \frac{\mathrm{TP}}{\mathrm{TP+FN+FP}}$$

Lasso

Tibshirani R., Journal of the Royal Statistical Society: Series B, 199

Training phase

data: $X \in \mathbb{R}^{N \times F}$ N samples with F features

label: y $\in \mathbb{R}^{N}$

Lasso method:

$$\hat{\beta} = argmin_{\beta}(\|y - X\beta\|_2^2 + \lambda\|\beta\|_1) \text{ with } \lambda \in \mathbb{R}$$

$$\hat{\beta} \in \mathbb{R}^{F}$$

$$X\hat{\beta} = \hat{y}$$

Lasso

Tibshirani R., Journal of the Royal Statistical Society: Series B, 199

Testing phase

data: $W \in \mathbb{R}^{M \times F}$ M samples with F features

label: $v \in \mathbb{R}^{N}$

Lasso method:

$$W\hat{\beta} = \hat{v}$$

Lasso

Tibshirani R., Journal of the Royal Statistical Society: Series B, 199

We have now a trained model (algorithm + data)

$$x^{new} \in \mathbb{R}^{1 \times F} \rightarrow x^{new}\,\hat{\beta} = y^{new}$$

Some questions

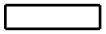
- Why do we need a training and a test phase?
- What about λ? Optimized via cross-validation (Stone M., Journal of the Royal Statistical Society: Series B, 1974)
- No guarantee that $y^{new} \in \{0,1\}$. We need a threshold.
- How can we solve $\hat{\beta} = argmin_{\beta}(\|y - X\beta\|_2^2 + \lambda\|\beta\|_1)$?

Feature selection

Guyon I. et al, Machine Learning, 2002

If we have many features/variables/descriptors, a feature selection step could be useful:

- To reduce the dimensionality of the data.
- To get what features are really useful for the construction of the model.
- To interpret the physical meaning of the (important) features.



Feature selection

Guyon I. et al, Machine Learning, 2002

X original training set, $\mathcal{P}$ = [1, ..., P] list of all the features, $\mathcal{R}$ = [] list of ranked features.

While $\mathcal{P} \neq \emptyset$

- Xp = X[:, $\mathcal{P}$]
- Estimate $\hat{\beta}$
- Compute the weight vector $w = \sum_{i=1}^{N} \widehat{\beta}_i y_i \text{Xp}[i, :]$
- Compute the ranking criteria vector $c = (w)^2$
- Find $\tilde{p} = argmin_p\ (c)$
- Add $\tilde{p}$ to $\mathcal{R}$
- Remove $\tilde{p}$ from $\mathcal{P}$

Unsupervised machine learning

Clustering, blind signal source separation

We do not use labels to train the model

We are learning without a ‘teacher’

C-Means

MacQueen J. B., 1967

data: $X \in \mathbb{R}^{N \times F}$ N samples with F features

(fixed) number of clusters: C

centroids representing the clusters:

$$\mathcal{C} = \{\gamma_i \in \mathbb{R}^F s.t. i = 1, \dots, \mathrm{C}\}$$

membership matrix $\mathcal{U} \in \mathbb{R}^{\mathrm{C} \times N}$: element u_{ij} represents the (binary) membership of the *j*-th sample to the *i*-th cluster

C-Means

MacQueen J. B., 1967

$$J(\mathcal{C}, \mathcal{U}, X) = \sum_{i=1}^{\mathrm{C}} \sum_{j=1}^{N} u_{ij}\, d(x_j, \gamma_i)^2$$

where $d(x_i, \gamma_j)^2$ is the distance between the sample x_j and the centroid γ_i

$$u_{ij} = \begin{cases} 1 & \text{if } d(x_j, \gamma_i) \leq d(x_j, \gamma_k)\ \forall\ k = 1, \dots \mathrm{C} \\ 0 & \text{otherwise} \end{cases}$$

$$\gamma_i = \frac{\sum_{j=1}^{N} u_{ij}\, x_j}{\sum_{j=1}^{N} u_{ij}}$$

Python, R, Matlab...

Programming languages have nowadays special tools for machine learning

scikit-learn is open-source machine learning Python library

https://scikit-learn.org/stable/