

Machine learning and physical (Earth system) modelling - course 2

julien.brajard@nersc.no

October 2021

NERSC

slides+notebook:https://github.com/nansencenter/nersc_ml_course

Table of contents i

1. Model selection/validation
2. Case of auto-correlated data
3. L1/L2 regularization
4. Steps of a machine learning process

Model selection/validation

Polynomial regression

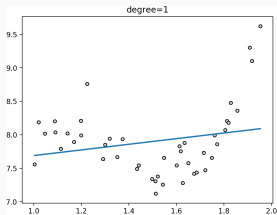
$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_d x^d = \sum_{i=0}^d \theta_i X^i$$

Choice of the model

Polynomial regression

$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_d x^d = \sum_{i=0}^d \theta_i X^i$$

degree = 1 (linear)

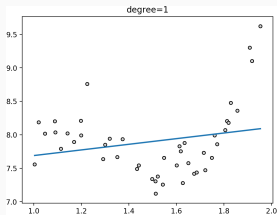


Choice of the model

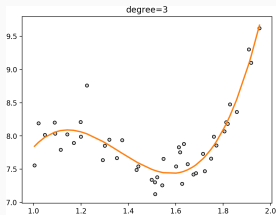
Polynomial regression

$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_d x^d = \sum_{i=0}^d \theta_i X^i$$

degree = 1 (linear)



degree = 3

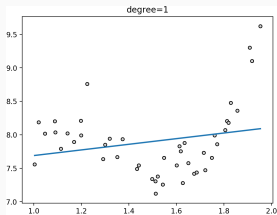


Choice of the model

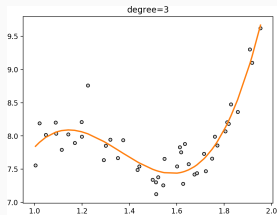
Polynomial regression

$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_d x^d = \sum_{i=0}^d \theta_i X^i$$

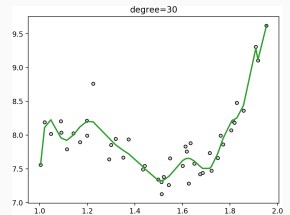
degree = 1 (linear)



degree = 3



degree = 30



What is the best model?

Train/Validation split

The idea

Evaluate a score on a independent dataset

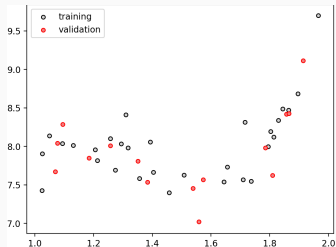
Train/Validation split

The idea

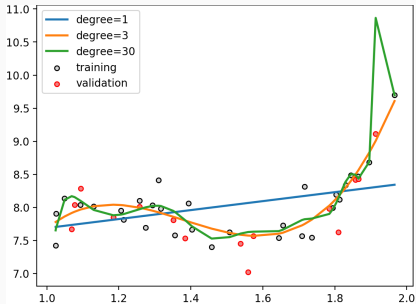
Evaluate a score on a independent dataset

In our example we can randomly divide (X, y) in two datasets:

- The training dataset X_{train}, y_{train} used to fit the model.
- The validation dataset X_{val}, y_{val} used to compute the score (e.g., correlation, mean-squared error)



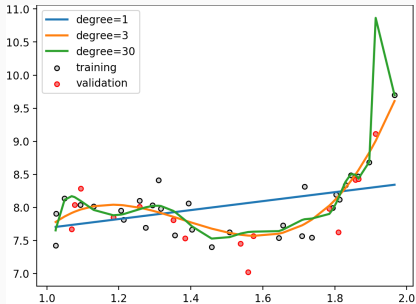
Choice of the model



Score: Mean Square Error (MSE)

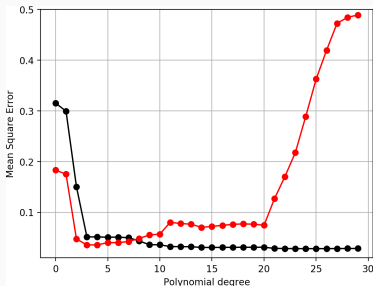
Deg.	Train Score	Val. Score
1	0.17	0.23
3	0.045	0.062
30	0.035	0.27

Choice of the model



Score: Mean Square Error (MSE)

Deg.	Train Score	Val. Score
1	0.17	0.23
3	0.045	0.062
30	0.035	0.27

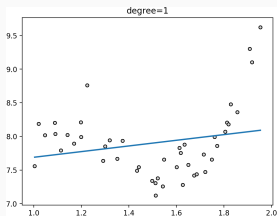


Choice of the model

Polynomial regression

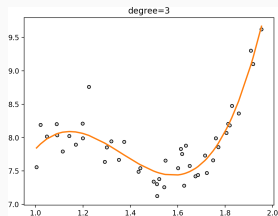
$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \dots + \theta_d x^d = \sum_{i=0}^d \theta_i X^i$$

degree = 1 (linear)



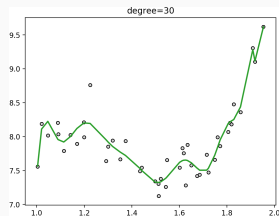
underfitting

degree = 3



good fit

degree = 30



overfitting

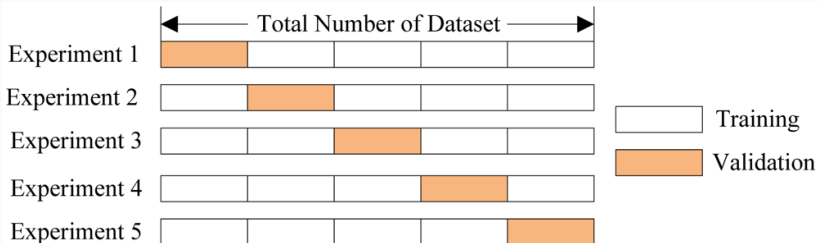
Drawbacks

- drastically reduce the number of samples which can be used for learning the model
- Results can depend on a particular random choice for the pair of (train, validation) sets.

More Robust: cross validation

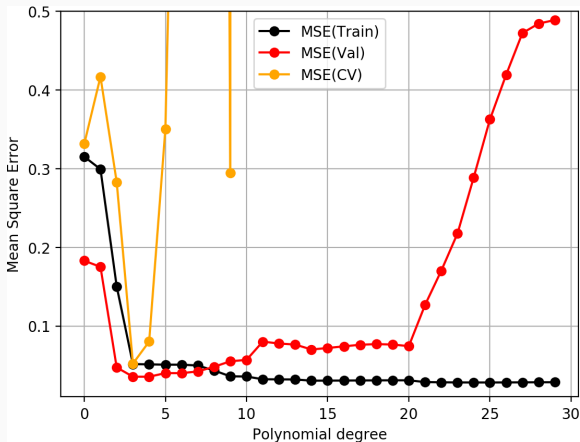
The idea

- Dividing the data in n folds,
- Learning n model (each time with a different training set),
- Compute the mean score over n validation set.



Cross-Validation

Fold	MSE
1	0.052
2	0.043
3	0.137
4	0.025
5	0.048
6	0.144
7	0.011
8	0.025
9	0.010
10	0.028
Mean	0.05



1. When applying machine learning techniques there are **hyperparameters** to be determined (e.g., degree of the polynomial in polynomial regression).

Wrapping up

1. When applying machine learning techniques there are **hyperparameters** to be determined (e.g., degree of the polynomial in polynomial regression).
2. These **hyperparameters** can be determined by splitting the data into training/validation or by cross-validation.

Wrapping up

1. When applying machine learning techniques there are **hyperparameters** to be determined (e.g., degree of the polynomial in polynomial regression).
2. These **hyperparameters** can be determined by splitting the data into training/validation or by cross-validation.
3. But then... the validation set was used to determine the best machine learning process

Wrapping up

1. When applying machine learning techniques there are **hyperparameters** to be determined (e.g., degree of the polynomial in polynomial regression).
2. These **hyperparameters** can be determined by splitting the data into training/validation or by cross-validation.
3. But then... the validation set was used to determine the best machine learning process
4. To evaluate independantly the performance of our model, we should compute the score on a **third independant dataset: The test dataset.**

Case of auto-correlated data

Random split

- A standard way to select the validation is to split randomly the dataset at a given proportion.

If 15% of the point is in the validation set, each sample \mathbf{x}_i has a probability of 15% to be in the validation set

Random split

- A standard way to select the validation is to split randomly the dataset at a given proportion.

If 15% of the point is in the validation set, each sample x_i has a probability of 15% to be in the validation set

- It can be done using the sklearn python library

```
from sklearn.model_selection import train_test_split  
  
X_train, X_test, y_train, y_test =  
    train_test_split(X, y, test_size=0.15)
```

Random split

- A standard way to select the validation is to split randomly the dataset at a given proportion.

If 15% of the point is in the validation set, each sample x_i has a probability of 15% to be in the validation set

- It can be done using the sklearn python library

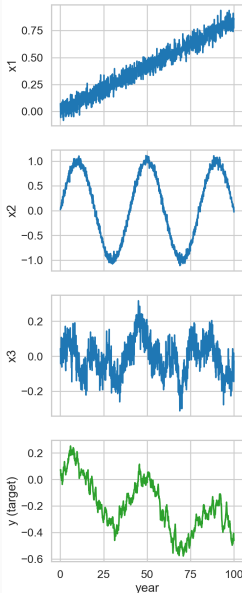
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test =
    train_test_split(X, y, test_size=0.15)
```

- **WARNING!** It can lead to problems with auto-correlated data (e.g. pixels of an image, time series).

More exactly: it leads to problem if the residual between the target and the model prediction (a.k.a. model error) is auto-correlated.

Illustration

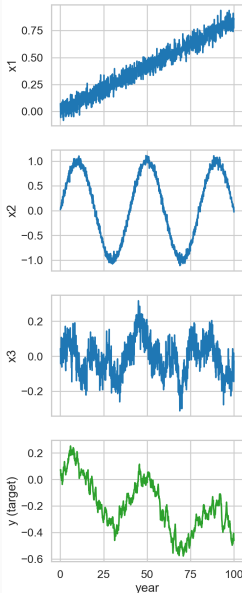


Objective: predict the target y from the three features x_1, x_2 and x_3 .

Note: this is an artificial problem that have been created for the illustration. The true model is known:

$$y = -\frac{1}{2}x_1 + \frac{1}{5}x_2 + \frac{1}{5}x_3$$

Illustration



Objective: predict the target y from the three features x_1, x_2 and x_3 .

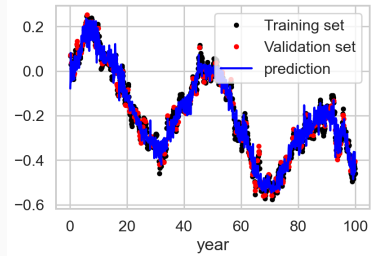
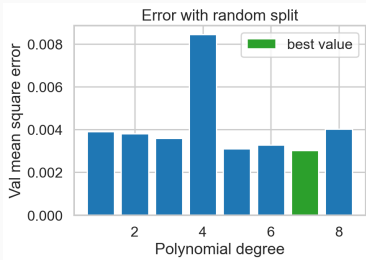
Note: this is an artificial problem that have been created for the illustration. The true model is known:

$$y = -\frac{1}{2}x_1 + \frac{1}{5}x_2 + \frac{1}{5}x_3$$

Model and metric

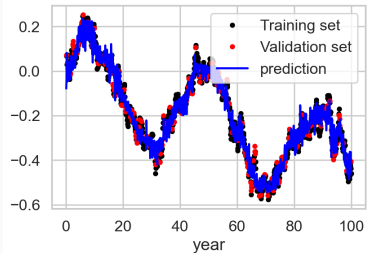
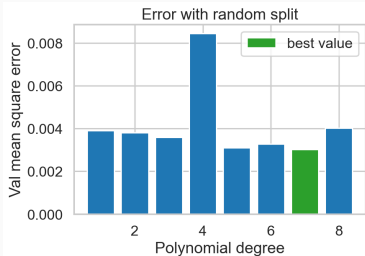
We will use a polynomial regression and chose the polynomial degree of the best model (in term of mean square error) from a validation set

Result with random split

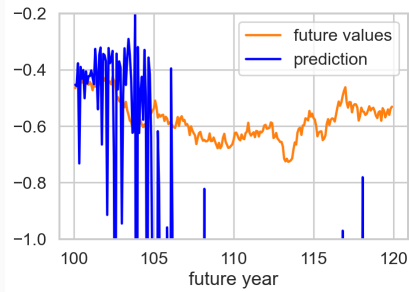


Can the model determine ($d = 7$) generalize to new data? (year > 100)

Result with random split

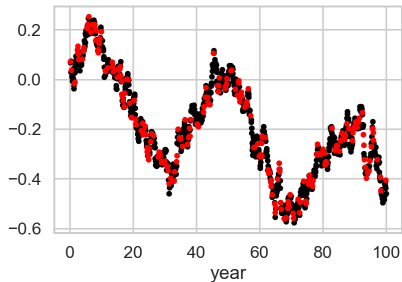


Can the model determine ($d = 7$) generalize to new data? (year > 100)

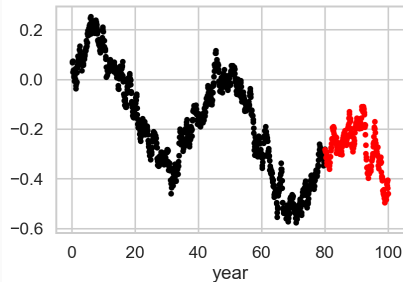


One approach: block split

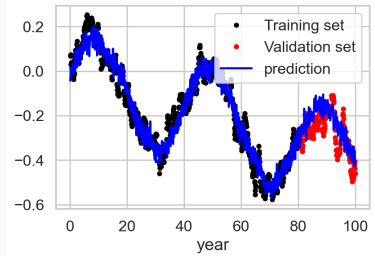
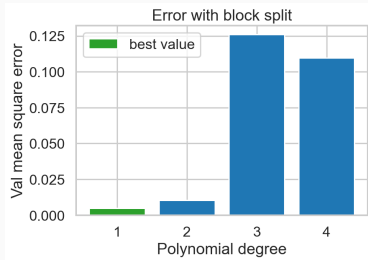
Random split (15%)



Block split (15%)

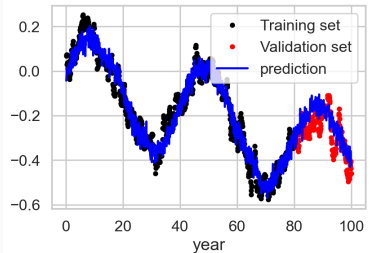
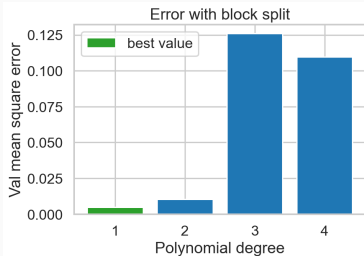


Result with block split

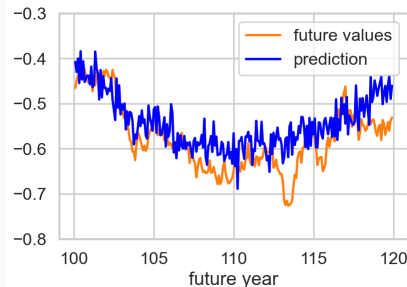


Can the model determine ($d = 1$) generalize to new data? (year > 100)

Result with block split

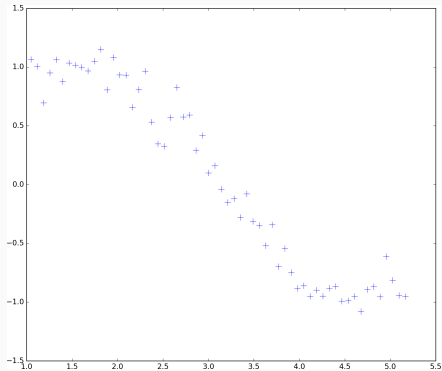


Can the model determine ($d = 1$) generalize to new data? (year > 100)



L1/L2 regularization

Example of a non-linear relationship



An idea

We could take a polynomial hypothesis model:

$$h_{\theta}(x) = \theta_0 x^0 + \theta_1 x^1 + \dots + \theta_p x^p$$

Example

$\{(x_1, y_1), \dots, (x_n, y_n)\}$ is the training dataset.

For a given polynomial degree p , parameters θ are determined minimizing the least-mean square cost function:

$$J(\theta) = \frac{1}{n} \sum (y_i - h_{\theta}(x_i))^2$$

with $h_{\theta}(x) = \theta_0 x^0 + \theta_1 x^1 + \dots + \theta_p x^p$

- It can be determined using a gradient descent method

Example

$\{(x_1, y_1), \dots, (x_n, y_n)\}$ is the training dataset.

For a given polynomial degree p , parameters θ are determined minimizing the least-mean square cost function:

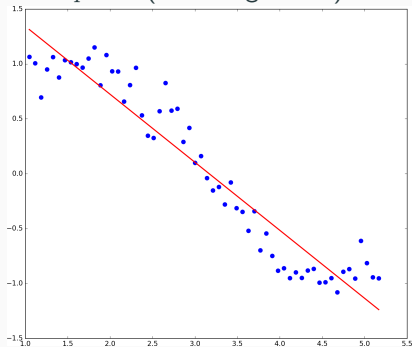
$$J(\theta) = \frac{1}{n} \sum (y_i - h_{\theta}(x_i))^2$$

with $h_{\theta}(x) = \theta_0 x^0 + \theta_1 x^1 + \dots + \theta_p x^p$

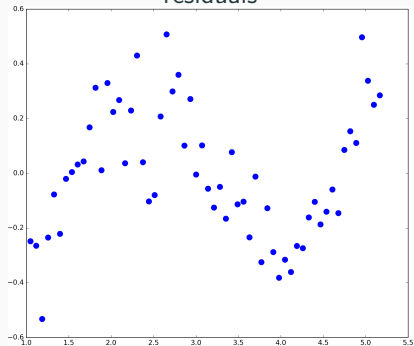
- It can be determined using a gradient descent method
- If the degree of the polynomial $p = 1$, it is a simple linear regression

A first result

$p = 1$ (linear regression)



residuals

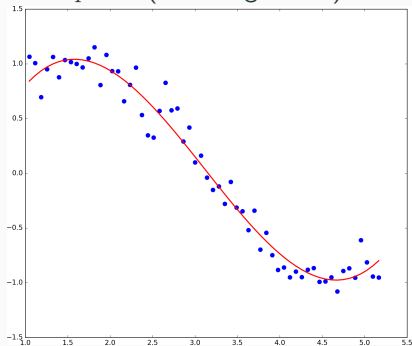


Prediction error

$$\text{err} = \frac{1}{n} \sum \text{res}^2 = 5.46e - 2$$

Increasing the polynomial degree ?

$p = 3$ (cubic regression)



Prediction error

$$\text{err} = \frac{1}{n} \sum \text{res}^2 = 1.80e - 2$$

Is it different from the linear regression ?

Let's consider :

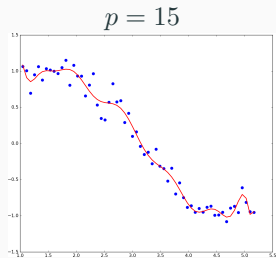
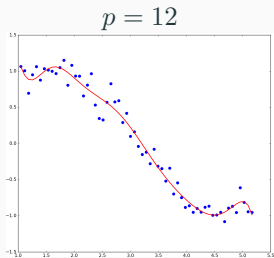
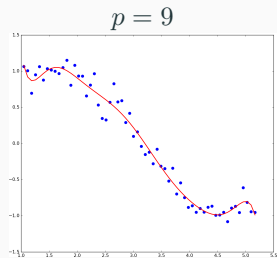
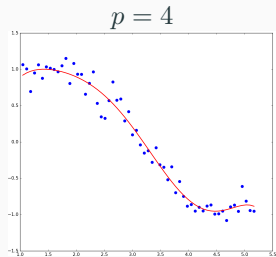
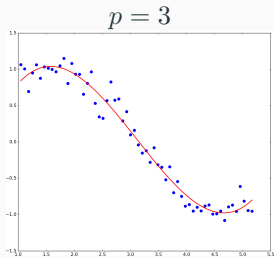
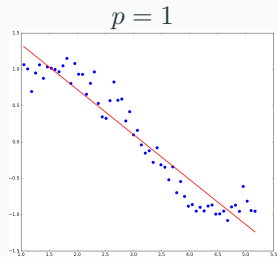
$$\mathbf{x} = \begin{pmatrix} 1 \\ x^1 \\ x^2 \\ \vdots \\ x^p \end{pmatrix}$$

then

$$h_{\boldsymbol{\theta}}(x) = \theta_0 + \theta_1 x^1 + \dots + \theta_p x^p = \boldsymbol{\theta}^T \mathbf{x}$$

By extending a scalar predictor to a vector, polynomial regression is equivalent to linear regression.

Increasing the degree ?



Overfitting

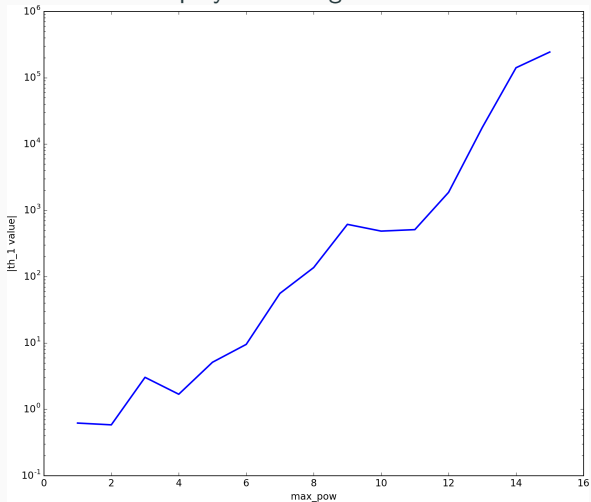
When there is too many parameters to fit, the model can reproduce a random noise, it is called **overfitting**

When there is too many parameters to fit, the model can reproduce a random noise, it is called **overfitting**

	rmse	th_0	th_1	th_2	th_3	th_4	th_5	th_6	th_7	th_8
max_pow_1	+5.47e-02	+1.96e+00	-6.20e-01	NaN	NaN	NaN	NaN	NaN	NaN	NaN
max_pow_2	+5.46e-02	+1.91e+00	-5.83e-01	-5.96e-03	NaN	NaN	NaN	NaN	NaN	NaN
max_pow_3	+1.84e-02	-1.08e+00	+3.03e+00	-1.29e+00	+1.37e-01	NaN	NaN	NaN	NaN	NaN
max_pow_4	+1.80e-02	-2.66e-01	+1.69e+00	-5.32e-01	-3.57e-02	+1.39e-02	NaN	NaN	NaN	NaN
max_pow_5	+1.70e-02	+2.99e+00	-5.12e+00	+4.72e+00	-1.93e+00	+3.35e-01	-2.07e-02	NaN	NaN	NaN
max_pow_6	+1.65e-02	-2.80e+00	+9.52e+00	-9.71e+00	+5.23e+00	-1.55e+00	+2.33e-01	-1.36e-02	NaN	NaN
max_pow_7	+1.55e-02	+1.93e+01	-5.60e+01	+6.90e+01	-4.46e+01	+1.65e+01	-3.53e+00	+4.05e-01	-1.92e-02	NaN
max_pow_8	+1.53e-02	+4.32e+01	-1.37e+02	+1.84e+02	-1.33e+02	+5.77e+01	-1.53e+01	+2.42e+00	-2.10e-01	+7.68e-03
max_pow_9	+1.46e-02	+1.68e+02	-6.15e+02	+9.63e+02	-8.46e+02	+4.61e+02	-1.62e+02	+3.68e+01	-5.22e+00	+4.22e-01
max_pow_10	+1.46e-02	+1.38e+02	-4.86e+02	+7.26e+02	-5.96e+02	+2.93e+02	-8.75e+01	+1.45e+01	-8.06e-01	-1.38e-01
max_pow_11	+1.45e-02	-7.49e+01	+5.12e+02	-1.33e+03	+1.87e+03	-1.61e+03	+9.14e+02	-3.50e+02	+9.14e+01	-1.61e+01
max_pow_12	+1.45e-02	-3.39e+02	+1.87e+03	-4.42e+03	+6.01e+03	-5.25e+03	+3.12e+03	-1.30e+03	+3.84e+02	-8.03e+01
max_pow_13	+1.43e-02	+3.20e+03	-1.78e+04	+4.46e+04	-6.66e+04	+6.61e+04	-4.61e+04	+2.32e+04	-8.55e+03	+2.30e+03
max_pow_14	+1.31e-02	+2.38e+04	-1.41e+05	+3.79e+05	-6.10e+05	+6.57e+05	-5.03e+05	+2.82e+05	-1.17e+05	+3.66e+04
max_pow_15	+1.17e-02	-3.62e+04	+2.44e+05	-7.46e+05	+1.38e+06	-1.71e+06	+1.53e+06	-1.00e+06	+4.98e+05	-1.88e+05

High parameters values

Value of the parameters $|\theta_1|$ with respect with the degree of the polynomial regression



The idea

The idea of regularization is to perform a regression minimizing a cost function that includes a term to penalize "big" values for the parameters:

$$J(\boldsymbol{\theta}) = \frac{1}{n} \sum (y_i - h_{\boldsymbol{\theta}}(x_i))^2 + \alpha P(\boldsymbol{\theta})$$

Regularization

The idea

The idea of regularization is to perform a regression minimizing a cost function that includes a term to penalize "big" values for the parameters:

$$J(\boldsymbol{\theta}) = \frac{1}{n} \sum (y_i - h_{\boldsymbol{\theta}}(x_i))^2 + \alpha P(\boldsymbol{\theta})$$

We consider two penalty terms:

- Ridge Regularization (L2): $P(\boldsymbol{\theta}) = \sum_{i=0}^p \theta_i^2$

The idea

The idea of regularization is to perform a regression minimizing a cost function that includes a term to penalize "big" values for the parameters:

$$J(\boldsymbol{\theta}) = \frac{1}{n} \sum (y_i - h_{\boldsymbol{\theta}}(x_i))^2 + \alpha P(\boldsymbol{\theta})$$

We consider two penalty terms:

- Ridge Regularization (L2): $P(\boldsymbol{\theta}) = \sum_{i=0}^p \theta_i^2$
- Lasso Regularization (L1): $P(\boldsymbol{\theta}) = \sum_{i=0}^p |\theta_i|$

The idea

The idea of regularization is to perform a regression minimizing a cost function that includes a term to penalize "big" values for the parameters:

$$J(\boldsymbol{\theta}) = \frac{1}{n} \sum (y_i - h_{\boldsymbol{\theta}}(x_i))^2 + \alpha P(\boldsymbol{\theta})$$

We consider two penalty terms:

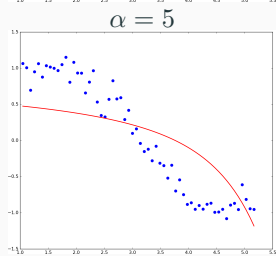
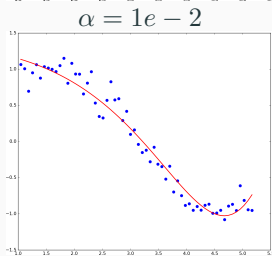
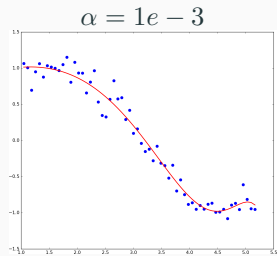
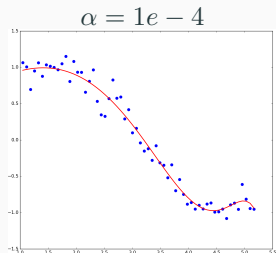
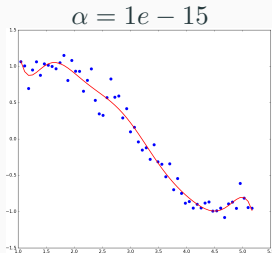
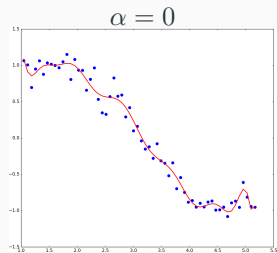
- Ridge Regularization (L2): $P(\boldsymbol{\theta}) = \sum_{i=0}^p \theta_i^2$
- Lasso Regularization (L1): $P(\boldsymbol{\theta}) = \sum_{i=0}^p |\theta_i|$
- Elastic Net combines both regularization

Ridge regression (L2)

Ridge regression is a linear regression with a Ridge regularization:

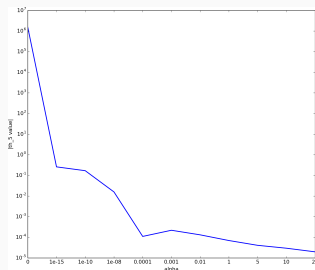
$$J(\theta) = \frac{1}{n} \sum (y_i - h_{\theta}(x_i))^2 + \alpha \sum_{i=0}^p \theta_i^2$$

Results for $p = 15$ and varying α



Values of coefficients

alpha_0	+1.17e-02	-3.62e+04	+2.44e+05	-7.46e+05	+1.38e+06	-1.71e+06	+1.53e+06	-1.00e+06	+4.98e+05	-1.88e+05
alpha_1e-15	+1.46e-02	+9.43e+01	-2.98e+02	+3.79e+02	-2.37e+02	+6.72e+01	-2.60e-01	-4.35e+00	+5.62e-01	+1.42e-01
alpha_1e-10	+1.54e-02	+1.12e+01	-2.90e+01	+3.11e+01	-1.52e+01	+2.89e+00	+1.69e-01	-9.10e-02	-1.08e-02	+1.98e-03
alpha_1e-08	+1.58e-02	+1.34e+00	-1.53e+00	+1.75e+00	-6.80e-01	+3.88e-02	+1.58e-02	+1.59e-04	-3.60e-04	-5.37e-05
alpha_0.0001	+1.60e-02	+5.61e-01	+5.47e-01	-1.28e-01	-2.57e-02	-2.82e-03	-1.10e-04	+4.06e-05	+1.52e-05	+3.65e-06
alpha_0.001	+1.67e-02	+8.18e-01	+3.05e-01	-8.67e-02	-2.05e-02	-2.84e-03	-2.19e-04	+1.81e-05	+1.24e-05	+3.43e-06
alpha_0.01	+2.39e-02	+1.30e+00	-8.84e-02	-5.15e-02	-1.01e-02	-1.41e-03	-1.32e-04	+7.23e-07	+4.14e-06	+1.30e-06
alpha_1	+9.41e-02	+9.69e-01	-1.39e-01	-1.93e-02	-3.00e-03	-4.66e-04	-6.97e-05	-9.90e-06	-1.29e-06	-1.43e-07
alpha_5	+2.31e-01	+5.48e-01	-5.89e-02	-8.52e-03	-1.42e-03	-2.41e-04	-4.08e-05	-6.87e-06	-1.15e-06	-1.91e-07
alpha_10	+3.00e-01	+4.00e-01	-3.72e-02	-5.53e-03	-9.50e-04	-1.67e-04	-2.96e-05	-5.23e-06	-9.25e-07	-1.63e-07
alpha_20	+3.79e-01	+2.77e-01	-2.25e-02	-3.40e-03	-5.99e-04	-1.08e-04	-1.97e-05	-3.60e-06	-6.58e-07	-1.20e-07



Value of θ_5

Determination of the parameters in the ridge regression

Considering the cost function J :

$$J(\boldsymbol{\theta}) = J_{lms}(\boldsymbol{\theta}) + \alpha \sum_{i=0}^p \theta_i^2$$

In a gradient algorithm, update of the parameters:

$$\theta_i^{k+1} = \theta_i^k - \nu \cdot \left(\frac{\partial J_{lms}}{\partial \theta_i} - 2\alpha \theta_i^k \right)$$

So the update rule is:

$$\theta_i^{k+1} = \theta_i^k (1 - 2\nu\alpha) - \Delta_{lms}$$

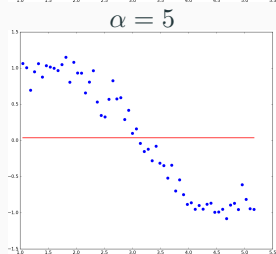
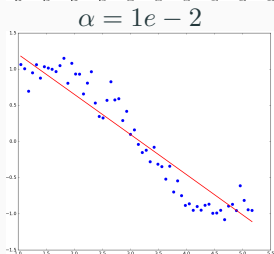
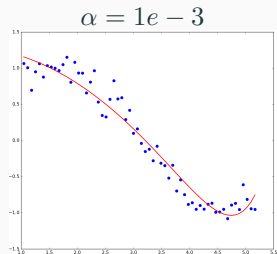
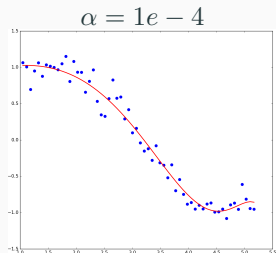
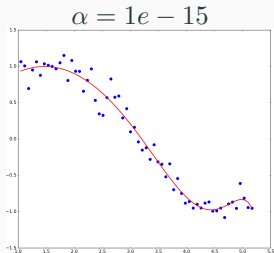
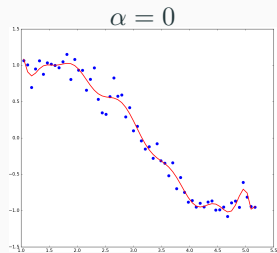
where Δ_{lms} is the update in case of non-regularized regression

Lasso regression (L1)

Lasso regression is a linear regression with a Lasso regularization:

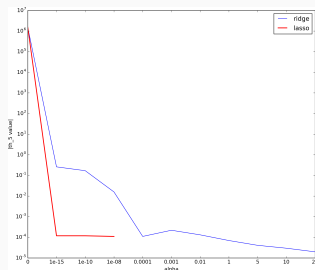
$$J(\boldsymbol{\theta}) = \frac{1}{n} \sum (y_i - h_{\boldsymbol{\theta}}(x_i))^2 + \alpha \sum_{i=0}^p |\theta_i|$$

Results for $p = 15$ and varying α



Values of coefficients

alpha_0	+1.17e-02	-3.62e+04	+2.44e+05	-7.46e+05	+1.38e+06	-1.71e+06	+1.53e+06	-1.00e+06	+4.98e+05	-1.88e+05
alpha_1e-15	+1.59e-02	+2.22e-01	+1.06e+00	-3.69e-01	+8.85e-04	+1.63e-03	-1.19e-04	-6.44e-05	-6.28e-06	+1.45e-06
alpha_1e-10	+1.59e-02	+2.22e-01	+1.06e+00	-3.69e-01	+8.84e-04	+1.63e-03	-1.18e-04	-6.44e-05	-6.28e-06	+1.45e-06
alpha_1e-08	+1.59e-02	+2.22e-01	+1.06e+00	-3.69e-01	+7.69e-04	+1.62e-03	-1.10e-04	-6.45e-05	-6.32e-06	+1.43e-06
alpha_0.0001	+1.72e-02	+9.03e-01	+1.71e-01	-0.00e+00	-4.78e-02	-0.00e+00	-0.00e+00	+0.00e+00	+0.00e+00	+9.47e-06
alpha_0.001	+2.80e-02	+1.29e+00	-0.00e+00	-1.26e-01	-0.00e+00	-0.00e+00	-0.00e+00	+0.00e+00	+0.00e+00	+0.00e+00
alpha_0.01	+6.07e-02	+1.76e+00	-5.52e-01	-5.62e-04	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00
alpha_1	+6.16e-01	+3.80e-02	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00
alpha_5	+6.16e-01	+3.80e-02	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00
alpha_10	+6.16e-01	+3.80e-02	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00
alpha_20	+6.16e-01	+3.80e-02	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00	-0.00e+00



Value of θ_5

Determination of the parameters in the lasso regression

Considering the cost function J:

$$J(\boldsymbol{\theta}) = J_{lms}(\boldsymbol{\theta}) + \alpha \sum_{i=0}^p |\theta_i|$$

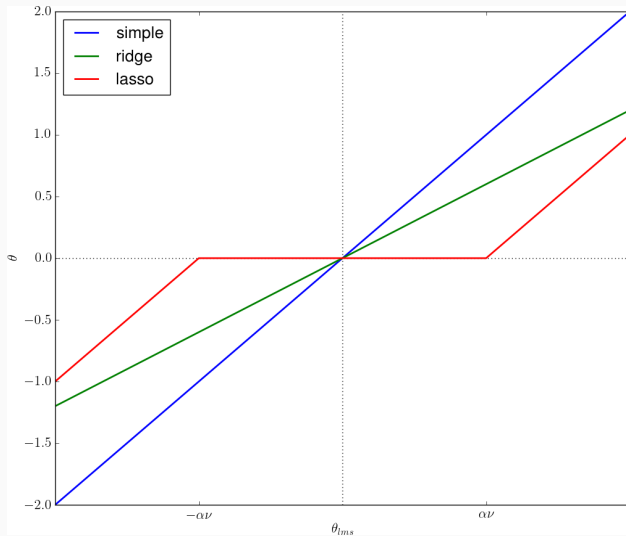
In a gradient algorithm, update of the parameters would be:

J is not differentiable. If we consider $\theta_{lms} = \theta_i^k - \nu \cdot \frac{\partial J_{lms}}{\partial \theta_i}$

The update rule is:

- $\theta_i^{k+1} = \theta_{lms} + \alpha\nu$ if $\theta_{lms} < -\alpha\nu$
- $\theta_i^k = 0$ if $-\alpha\nu < \theta_{lms} < \alpha\nu$
- $\theta_i^{k+1} = \theta_{lms} - \alpha\nu$ if $\theta_{lms} > \alpha\nu$

Summary of parameters updates



Ridge (L2)

- Prevents the overfitting
- includes all the features (dimensions) of the predictor, so it can be useless for high dimensional predictors

Comparison Ridge/Lasso

Ridge (L2)

- Prevents the overfitting
- includes all the features (dimensions) of the predictor, so it can be useless for high dimensional predictors

Lasso (L1)

- Provides sparse solutions and reduce the dimension of the predictor
- If some features in the predictor are correlated, arbitrarily select one from the others.

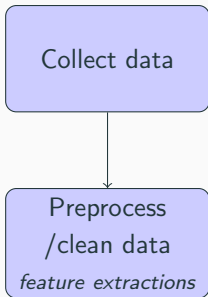
Steps of a machine learning process

Steps

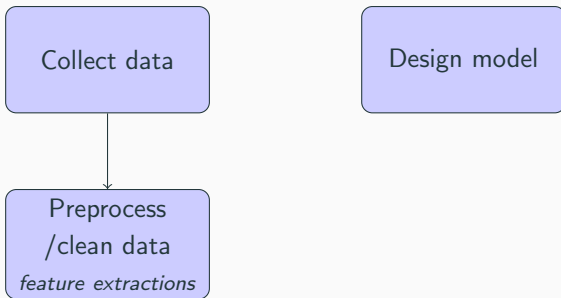


Collect data

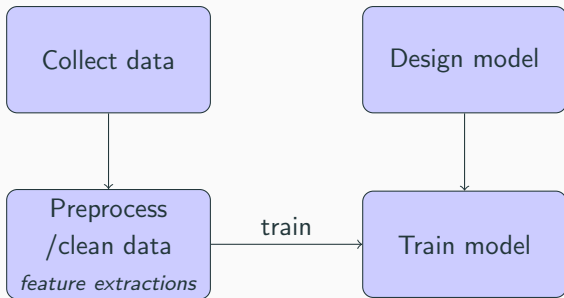
Steps



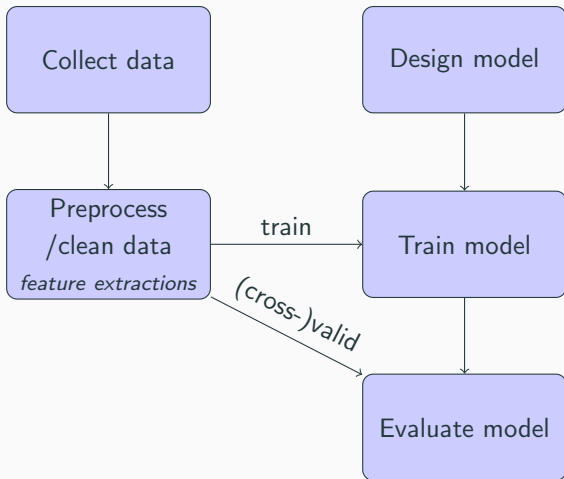
Steps



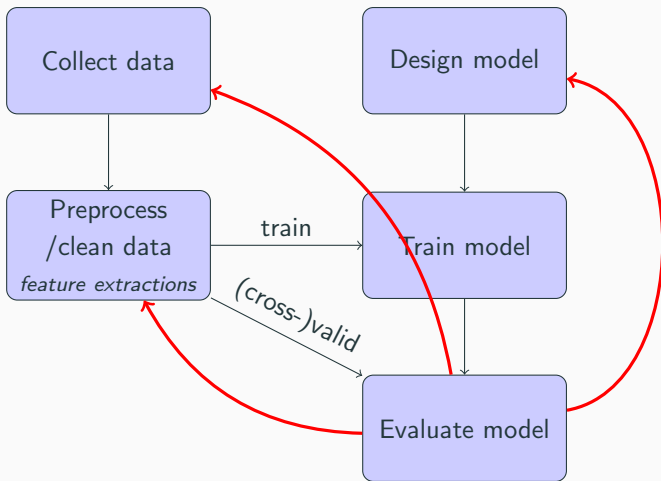
Steps



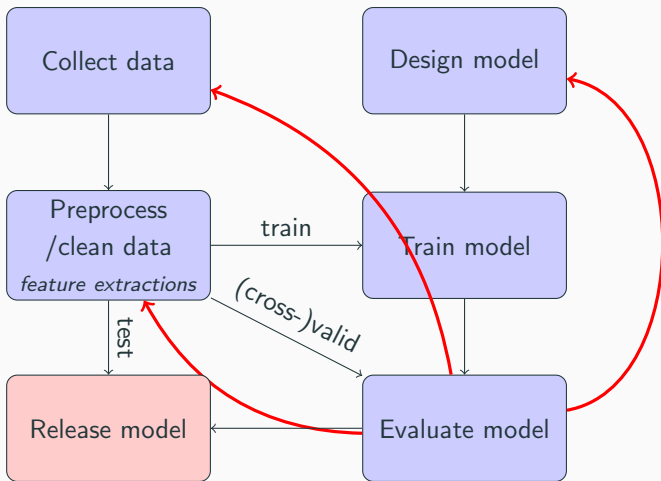
Steps



Steps



Steps



In summary

From one dataset, 3 sub-datasets have to be extracted:

- A training dataset
- A validation dataset

Can be done iteratively in a cross-validation procedure.

Some parameters of the model (e.g. polynomial order in a polynomial regression) were determined from the validation dataset.

- A test dataset (independent from the two other) to estimate the final performance of the model.