Apprentissage automatique Avancé

Lesson 2 : sparse models, and model selection

Sébastien Verel

Laboratoire d'Informatique, Signal et Image de la Côte d'opale (LISIC)
Université du Littoral Côte d'Opale, Calais, France
`http://www-lisic.univ-littoral.fr/~verel/`

October, 2023

General outline

- Partie n°1 [6h] : Bonnes pratiques en IA
- Partie n°2 [3h] : Méthodes Ensemblistes
- Partie n°3 [3h] : Autoencodeurs
- Partie n°4 [3h] : Réseaux convolutionnels
- Partie n°5 [6h] : Réseaux antagonistes génératifs
- Partie n°6 [6h] : Traitement Naturel du Langage
- Partie n°7 [7h] : Apprentissage par renforcement

Outline of the day

- Overfiting
- Regularization methods
- Model selection

Polynomial regression

Definition of the model

With polynomial linear regression :

$$h_\beta(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 + \beta_5 X_1 X_2 + ...$$

Mean square error function

$$J_{x,y}(\beta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\beta(x_i) - y_i)^2$$

Linear regression is simple, interpretable, easy to write,
and can be generalized with a sum of any set of functions $\{\varphi_i\}$:

$$h_\beta(X) = \beta_0 + \beta_1 \varphi_1(X) + \beta_2 \varphi_2(X) + \beta_3 \varphi_3(X) + \ldots + \beta_p \varphi_p(X)$$

but...

Practice alone (3)

Use the data set `cars` from `data01/cars.csv`

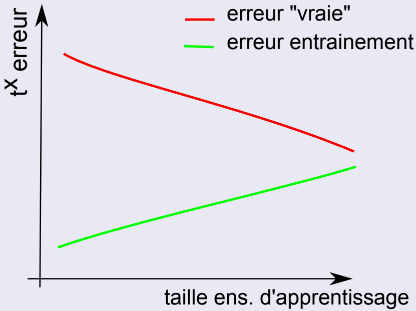
1. Compute a regression of degree 2.
2. What is the best polynomial regression ?

See `PolynomialFeatures` if needed

Errors of models

Relation between errors

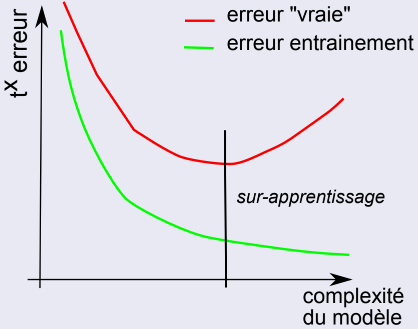
- Learning error : error rate on the training set
- "True" error : error rate on the all possible examples



Overfitting

Error *vs.* Model complexity

Too much learning ⇒ **overfit** of the model on the training set
⇒ Loss of generalization capacity ≈ Learning "by heart"



Complexity metrics :

polynomial degree, number of independent terms, etc.

Bias-variance relation

Suppose a function f to learn such that $y_i = f(x_i) + \epsilon_i$, where ϵ_i is the noise a null mean, and variance σ^2.

$\{(x_i, y_i)\}$ a training set, h a learnt model, and $\bar{h}$ the "average" model learn on all possible sets.

The variance of a model can be decomposed (see proof) by :

$$E[(h(x_i) - y_i)^2] = E[(h(x_i) - f(x_i))^2] + E[(f(x_i) - y_i)^2]$$

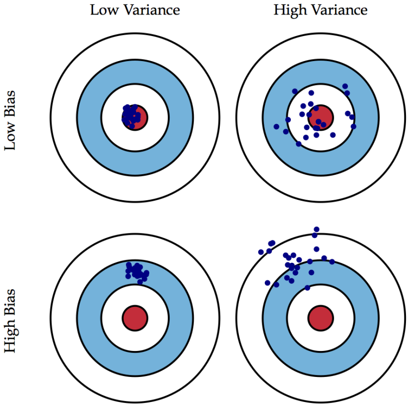
Relation biais/variance

$E[(h(x_i) - y_i)^2] =$

$$E[(\bar{h}(x_i) - f(x_i))^2] + E[(h(x_i) - \bar{h}(x_i))^2] + E[(f(x_i) - y_i)^2]$$

$$\text{Variance of model} = \text{Bias}^2 + \text{Variance} + \text{Variance of Noice}$$

Overfitting : bias-variance tradeoff



- Error from the **bias** : difference between predictions and true values
- Erreur from the **variance** : variability of the prediction for one given data x

Source Scott Fortmann-Roe :
`http://scott.fortmann-roe.com/docs/BiasVariance.html`

Goodness of fit : Coefficient of determination

Definition

$$R^2 = 1 - \frac{\text{Variance of the residus}}{\text{Variance of the data}}$$

with residus : $r_i = h_\beta(x_i) - y_i$

The R^2 is the part of variance of f explained by the model h

- $R^2 \leq 1$, but R^2 can be negative
- $R^2 = 0$ when the model h is equal to mean value, *i.e.* when $h(x) = E[y_i]$.
- $R^2 < 0$ when the model h is worst than $h(x) = E[y_i]$
- $R^2 = 1$ when the model h is perfect, $h(x) = f(x)$ on the set to estimate R^2
- In general, but depending on the context, a relevant R^2 is above 0.8.

Evaluation of a model quality

Technique

Partitioning the set into :

- **Training** set ($\approx$ 70%)
- **Test** set, independent one ($\approx$ 30%)

Error rate, or R^2, can be estimated without bias on the test set.

Drawback

- An initial large set is required
- Dilemma :
  - The larger the test set, the better the estimation is
  - The larger the training set, the better the model is

Resampling method

Allow to estimate the generalization error

K-folds cross-validation

```
Partition randomly the set into $K$ blocks
For each blok $k$,
    Design a model using the $k - 1$ other training blocks
    Compute the test error $e_k$ on the block $k$
Compute the mean of errors $e_k$
```

Other techniques :

- Leave-one-out ($K = n$)
- Bootstrap, bagging, etc.

Exercice

See example in scikit-learn ”Underfitting vs. Overfitting” :

Polynomial regression of $f(x) = \cos(\frac{3\pi}{2}x)$ with polynomial of degree 1, 4, and 10

```
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression

polynomial_features = PolynomialFeatures(degree=degrees[i],
                  include_bias=False)
linear_regression = LinearRegression()
pipeline = Pipeline([
   ("polynomial_features", polynomial_features),
   ("linear_regression", linear_regression) ])

pipeline.fit(X[:, np.newaxis], y)
```

Why there is over fitting ?

In the context of glm with $\{\varphi_i\}$ having "good" properties.
Suppose that the true function f is the sum of c terms :

$$f(X) = \beta_0 + \beta_1\varphi_1(X) + \beta_2\varphi_2(X) + \ldots + \beta_c\varphi_c(X)$$

And we would like to find a model $h_{\hat{\beta}}$ with p terms :

$$h_{\hat{\beta}}(X) = \hat{\beta}_0 + \hat{\beta}_1\varphi_1(X) + \hat{\beta}_2\varphi_2(X) + \hat{\beta}_3\varphi_3(X) + \ldots + \hat{\beta}_p\varphi_p(X)$$

Indeed, we would like to solve the system :

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

with X is matrix of dimension $n \times p$: $X = [\ldots]$ ...

if $\text{rank}(X) \approx n \leq p+1$, there is an exact solution $\hat{\beta}$ (interpolation), but, maybe this solution can have more terms p than c...

If $n > p+1$, no garanti to have an exact solution of the system.

Toward a method to fight overfitting

$$f(X) = \beta_0 + \beta_1\varphi_1(X) + \beta_2\varphi_2(X) + \ldots + \beta_c\varphi_c(X)$$

$$h_{\hat{\beta}}(X) = \hat{\beta}_0 + \hat{\beta}_1\varphi_1(X) + \hat{\beta}_2\varphi_2(X) + \hat{\beta}_3\varphi_3(X) + \ldots + \hat{\beta}_p\varphi_p(X)$$

$$\{x_1, x_2, \ldots, x_n\}$$

Intuitively, the sample size should be around the $p+1$,
and p should be around c...

Idea

Translate this hypothesis based on intuition to the model design :
Find coeff. such that values are close to the original function,
In other words, when the number of predictors is large,
the value of coefficients should be constraint to zero

Regularization methods, sparse approach

Goal

Reduce the complexity of the model (cf. Ockham razor),
i.e the "size" of the coefficients

Reduce the variance of the estimated coefficients

Method

Modify the function to minimize,
add a penalty according to model complexity :

$$J_{x,y}(\beta) = L_\beta(x,y) + \lambda\ \Omega(\beta)$$

- $L_\beta(x,y)$: **training error** (MSE, etc.),
  distance between data and prediction of the model
- $\Omega(\beta)$: **regularization cost** (model complexity),
  distance between β_j and 0
- $\lambda \in \mathbb{R}$: tradeoff parameter between error, and complexity

[or, minimize $J_{x,y}(\beta) = L_\beta(x,y)$ such that $\Omega(\beta) \leq t$]

Regularization function

$$J_{x,y}(\beta) = L_\beta(x,y) + \lambda\, \Omega(\beta)$$

A L_k norm can be used : $\Omega(\beta) = \|\beta\|_k^k$

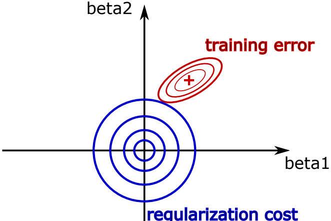
- L_2 Norm : Ridge regression [Hoerl *et al.*, 1970] [4]

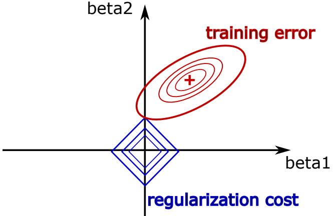
$$\Omega(\beta) = \|\beta\|_2^2 = \sum_{j=1}^{p} \beta_i^2$$

- L_1 Norm : Lasso regression [Tibshirani, 1996] [7]

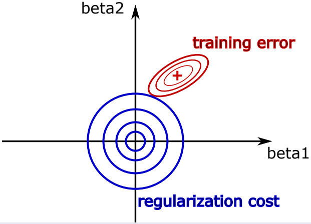
$$\Omega(\beta) = \|\beta\|_1 = \sum_{j=1}^{p} |\beta_j|$$

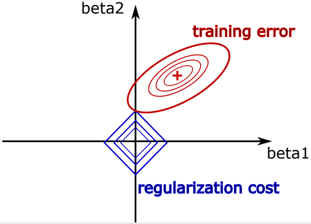
The shape of lines of equality are different, and the model :





Other regularization function





- Ridge regression : all coefficients are pushed to be small
- Lasso regression : the number of non-zero coeff. is minimized
- ElasticNet regression : combinaison of the ridge and lasso

$$\Omega(\beta) = \sum_{j=1}^{p} ((1-\alpha)\beta_j^2 + \alpha|\beta_j|)$$

The additional parameter $\alpha \in [0,1]$ tunes the tradeoff between ridge, and lasso parts

Don't forget to normalize the predictors !

Time complexity to compute the sparse regression

$$J_{x,y}(\beta) = L_\beta(x, y) + \lambda\,\Omega(\beta)$$

- L_2 Norm : Ridge regression

$$\Omega(\beta) = \|\beta\|_2^2 = \sum_{j=1}^{p} \beta_i^2$$

The gradient can be computed. Same complexity as before with simple MSE : $\mathcal{O}(p^3)$ (suppose that p larger than n)

- L_1 Norm : Lasso regression

$$\Omega(\beta) = \|\beta\|_1 = \sum_{j=1}^{p} |\beta_j|$$

The gradient is not defined for the absolute value on zero.
A new strategy is required to minimize Lasso criterium...

Coordinate descent

Principle

Optimize iteratively variable per variable the objective using gradient descent.
Here, the objective is $J_{x,y}(\beta) = L_\beta(x,y) + \lambda\ \Omega(\beta)$

```
Set each coefficient to zero : $\beta_j = 0$
repeat
    for $j = 1 \dots p$ do
        Compute gradient of var. $j$ : $g(\beta_j) = \partial L_\beta(x,y) + \lambda\ \partial\Omega(\beta)$
        Gradient step to update var. $j$ : $\beta_j = \beta_j - \alpha\ g(\beta_j)$
    end for
until stopping criterium is false
```

Comment : $\partial\Omega(\beta)$ is computed using sub-gradient :
$= -1$ if $\beta_j < -\lambda$, $= 0$ if $-\lambda \leq \beta_j \leq \lambda$, and $= 1$ if $\lambda < \beta_j$

As a consequence, λ tunes the importance of the regularization part.

Iterative selection methods

Position of the problem

Selection of non-zero terms from the p terms :

$$h_{\hat{\beta}}(X) = \hat{\beta}_0 + \hat{\beta}_1\varphi_1(X) + \hat{\beta}_2\varphi_2(X) + \hat{\beta}_3\varphi_3(X) + \ldots + \hat{\beta}_p\varphi_p(X)$$

$\Rightarrow$ optimization problem with the "shape" (discrete),
and the weights (continuous) of terms as decision variables

How many different different models (selection) from p terms ?

Greedy heuristics

- Backward selection : From the full model with p predictors, iteratively remove the worst interesting predictor
- Forward selection : From the empty model with no predictor, iteratively add the most interesting predictor

A lot of methods have been proposed to compute the Lasso regression.
Please read carefully to documentation of the used method.

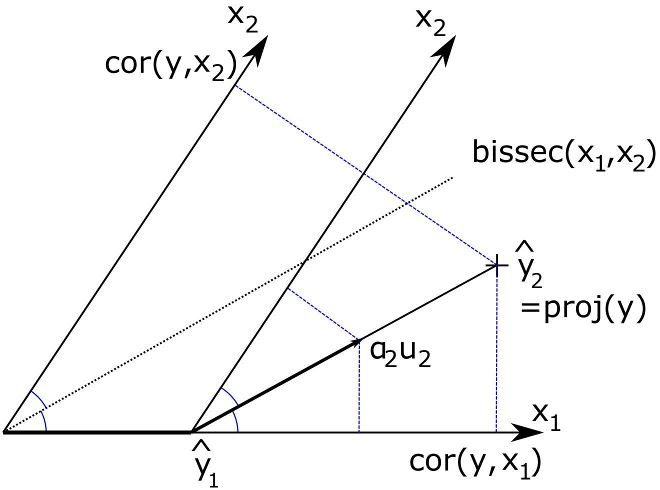
LARS regression [Efron *et al.* 2004] [3]

Algorithm

```
Set each coefficient to zero : $\beta_j = 0$
repeat
    Find the most correlated predictors $x_{j_1}$ with $y$

    Increase $\beta_{j_1}$ (following the correlation sign) until another predictor
    $x_{j_2}$ has much correlation with $r = y - h_\beta(x)$

    Increase $(\beta_{j_1}, \beta_{j_2})$ (following the join direction) until another
    predictor $x_{j_3}$ has much correlation with $r = y - h_\beta(x)$

    Increase $(\beta_{j_1}, \beta_{j_2}, \beta_{j_3})$ $\dots$
until all predictors are used in the model
```

LARS regression [Efron *et al.* 2004] [3]

In two dimension :



Comments on LARS

Pros

- Time complexity similar to OLS : $\mathcal{O}(p^3)$
- Take care about correlated variables
- Efficient when $p >> n$
- Lasso path (see after)

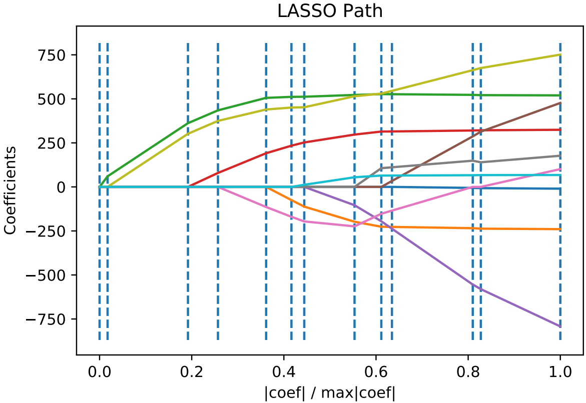
Cons

Less efficient when :

- High dimensional data
- a lot of noice, and high dimensional multicolinear independent variables

Lasso path

see example `https://scikit-learn.org/stable/auto_examples/linear_model/plot_lasso_lars.html`



Absolute value of each coefficient as a function of $\sum_j |\beta_j| < t$

Other regularization technique

Regularization is an old, and still active subject. A lot of techniques are proposed every year depending on the context

Dropout [Srivastava *et al.*, 2014] [6]

Proposed for neural network (can be extended to other models ?)

At each gradient step (epoch in the language of NN),
a neuron and its connexions are deactivated with probability p.

During test phase, all neurons are used, but weights are multiplied by factor p

Intuitive idea

Select at random some predictors at each steps :
To train several sub-models in parallel,
To escape from local optima,

Parameter tuning

Usually ML algorithms have some parameters, called **meta-parameters**. For each parameter value, the model could be different.

For example :

- parameter λ of sparse methods,
- parameters α in ElasticNet
- ...

So, a set of models is available.

How to select a "good" model from a set ?
$\approx$ How to tune the meta-parameters ?

Selection based on a criterium

To select a model from a set (finite or not) of models H,
a metrics is used to compare models.

The selected model is the best model according to the metric :

$$h^\star = \text{argmax}_{h \in H} P(h)$$

where P is a quality metric of model

Validation set

$$X = X_{train} \cup X_{test}$$

Test set can not be used (risk of overfitting) for model selection. The train set is divided using a validation set :

$$X_{train} = X_{train'} \cup X_{validation}$$

Each possible model (meta-parameters) can be compared using the quality estimated on validation set : $P(h) = error(h, X_{validation})$

When a model is selected, the full train set can be used to compute the final model.

Cross-validation for model selection

A K-fold cross-validation on the train set can be used to compare models. X_{train} is partitioned into K folds :

$$X_{train} = X_{fold_1} \cup \ldots \cup X_{fold_K}$$

The performance of model h is :

$$P(h) = E[\text{error}(h, X_{fold_k})]$$

Then, again, the training set can be used with the selected meta-parameter/model.

Akaike information criterion (AIC)

Akaike information criterion (AIC), from Hirotugu Akaike, 1971 [1] [2], is an information criterion (remember Shanon, entropy, etc.)

Likelihood (vraisemblance)

Definition

$$\mathcal{L}(\theta \mid x) = P_\theta(X = x) = P(X = x \mid \theta)$$

"The probability of the value x of X for the parameter value θ", *i.e.* conditional probability of observing x from the knowledge of parameter θ.

Suppose that a fait coin $p_H = 0.5$.
If we observe twice the head HH :
$\mathcal{L}(p_H = 0.5 \mid HH) = P(HH \mid p_H = 0.5) = 0.25$
Compute the the likelihood for hypothesis $p_H = 0.3$, and $p_H = 0.7$.

Likelihood (vraisemblance)

Definition

$$\mathcal{L}(\theta \mid x) = P_\theta(X = x) = P(X = x \mid \theta)$$

"The probability of the value x of X for the parameter value θ", *i.e.* conditional probability of observing x from the knowledge of parameter θ.

Suppose that a fait coin $p_H = 0.5$.
If we observe twice the head HH :
$\mathcal{L}(p_H = 0.5 \mid HH) = P(HH \mid p_H = 0.5) = 0.25$
Compute the the likelihood for hypothesis $p_H = 0.3$, and $p_H = 0.7$.

Indeed, this is basis of the maximum likelihood estimation (MLE) :
$\hat{\theta} = \text{argmax}_\theta \ \mathcal{L}(\theta \mid y)$
and using Bayes' rule :
$P(\theta \mid x) = P(x \mid \theta)P(\theta)/P(x) \propto \mathcal{L}(\theta \mid x) \ . \ \text{prior}(\theta)$

Likelihood ratio

Likelihood ratio

$$\Lambda(\theta_1 : \theta_2) = \frac{\mathcal{L}(\theta_1|x)}{\mathcal{L}(\theta_2|x)}$$

likelihood between two hypothesis (parameters).

Can be used to compare (stat. test) two hypothesis, and in Bayesian inference (it removes the proportional part) :

$O(\theta_1 : \theta_2 \mid x) = O(\theta_1 : \theta_2) \ . \ \Lambda(\theta_1 : \theta_2)$

Relative likelihood

$R(\theta) = \frac{\mathcal{L}(\theta|x)}{\mathcal{L}(\hat{\theta}|x)}$ where $\hat{\theta}$ is the estimate of the maximum likelihood.

Log-likelihood

$\log \mathcal{L}(\theta \mid x)$: maximizing likelihood, or log-likelihood is equivalent

more simple since $\log(ab) = \log(a) + \log(b)$, more accurate (precision of digits), and quicker to compute.

Measure the quantity of information !

Akaike information criterion (AIC)

Definition

$$\mathrm{AIC} = 2k - 2\ln \hat{\mathcal{L}}$$

where :

k : number of estimated parameters in the model
$\hat{\mathcal{L}}$: estimated likelihood of the model

The model with the smallest value is preferred

AIC estimates the relative information lost of the model from optimal model $\hat{\theta}$.
Tradeoff between accuracy of the model (likelihood), and model complexity (n. of param.)

Indeed, AIC compares two models g_1, and g_2 from an optimal (unknown) model f using relative likelihood. Be careful, the estimation is only valid asymptotically (when the number of data is large)

Bayesian information criterion (BIC) [5]

Definition

$$\mathrm{BIC} = k \ln n - 2 \ln \hat{\mathcal{L}}$$

where :

k : number of estimated parameters in the model

n : number of samples

$\hat{\mathcal{L}}$: estimated likelihood of the model

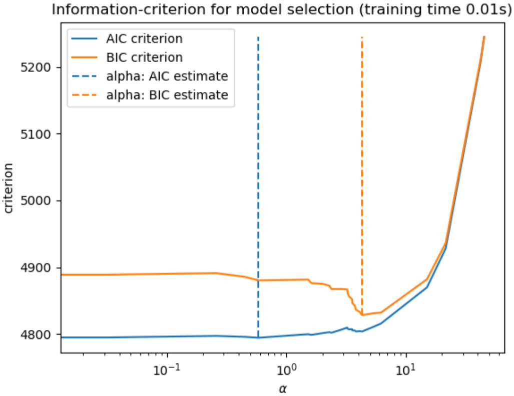
The number of parameter is $k = p + 2$ for multi-linear model.
The model with the smallest value is preferred

Tradeoff between accuracy of the model, and model complexity, but the larger penalty (2 *vs.* ln n) compare to AIC

BIC can be used to approximate the likelihood of the model knowing the data. It converges to the true value when n is large (on the contrary of AIC).

In practice with Scikit-learn

`https://scikit-learn.org/stable/auto_examples/linear_model/plot_lasso_model_selection.html`



Hirotugu Akaike.
A new look at the statistical model identification.
IEEE transactions on automatic control, 19(6) :716–723, 1974.

Kenneth P Burnham and David R Anderson.
Practical use of the information-theoretic approach.
In *Model selection and inference*, pages 75–117. Springer, 1998.

Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani.
Least angle regression.
The Annals of statistics, 32(2) :407–499, 2004.

Arthur E Hoerl and Robert W Kennard.
Ridge regression : Biased estimation for nonorthogonal problems.
Technometrics, 12(1) :55–67, 1970.

Gideon Schwarz.

Estimating the dimension of a model.
The annals of statistics, pages 461–464, 1978.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov.
Dropout : a simple way to prevent neural networks from overfitting.
The journal of machine learning research, 15(1) :1929–1958, 2014.

Robert Tibshirani.
Regression shrinkage and selection via the lasso.
Journal of the Royal Statistical Society : Series B (Methodological), 58(1) :267–288, 1996.