Predictive Modeling in Insurance,

in the context of (possibly) Big Data

A. Charpentier (UQAM & Université de Rennes 1)

Statistical & Actuarial Sciences Joint Seminar

& Center of studies in Asset Management (CESAM)

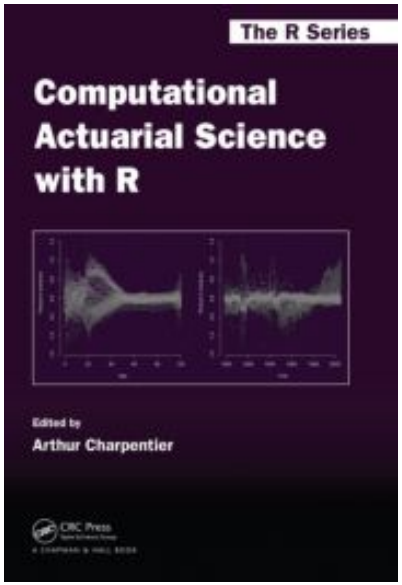
http://freakonometrics.hypotheses.org

Predictive Modeling in Insurance,

in the context of (possibly) Big Data

A. Charpentier (UQAM & Université de Rennes 1)

Professor of Actuarial Sciences, Mathematics Department, UQàM
(previously Economics Department, Univ. Rennes 1 & ENSAE Paristech
actuary in Hong Kong, IT & Stats FFSA)
ACTINFO-Covéa Chair, Actuarial Value of Information
Data Science for Actuaries program, Institute of Actuaries
PhD in Statistics (KU Leuven), Fellow Institute of Actuaries
MSc in Financial Mathematics (Paris Dauphine) & ENSAE
Editor of the freakonometrics.hypotheses.org's blog
Editor of Computational Actuarial Science, CRC



Actuarial Science, an 'American Perspective'

FUNDAMENTAL CONCEPTS OF ACTUARIAL SCIENCE

CHARLES L. TROWBRIDGE, F.S.A., M.A.A.A., E.A.

Revised Edition

ACTUARIAL EDUCATION AND RESEARCH FUND

CONTENTS

Source: Trowbridge (1989) *Fundamental Concepts of Actuarial Science.*

Actuarial Science, a 'European Perspective'

Contents

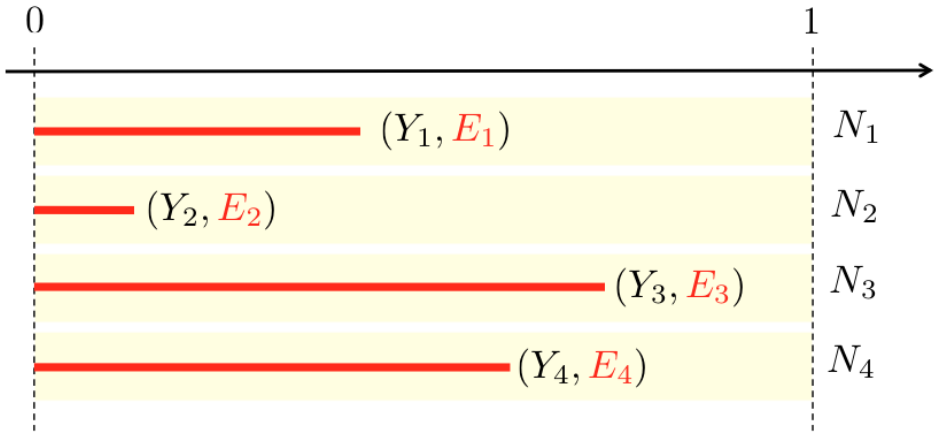
There are 10^{11} stars in the galaxy. That used to be a huge number. But it's only a hundred billion. It's less than the national deficit! We used to call them astronomical numbers. Now we should call them economical numbers —
Richard Feynman (1918–1988)

Source: Dhaene *et al.* (2004) *Modern Actuarial Risk Theory.*

Exemples of Actuarial Problems: Ratemaking and Pricing

$$\mathbb{E}[S|\boldsymbol{X}] = \mathbb{E}\left[\sum_{i=1}^{N} Z_i \middle| \boldsymbol{X}\right] = \underbrace{\mathbb{E}[N|\boldsymbol{X}]}_{\text{annual frequency}} \cdot \underbrace{\mathbb{E}[Z_i|\boldsymbol{X}]}_{\text{individual cost}}$$

- **censoring / incomplete datasets** (exposure + delay to report claims)



We observe Y and E, but the variable of interest is N.

$$Y_i \sim \mathcal{P}(E_i \cdot \lambda_i) \text{ with } \lambda_i = \exp[\beta_0 + \boldsymbol{x}_i^{\mathsf{T}}\boldsymbol{\beta} + Z_i].$$

Exemples of Actuarial Problems: Pricing and Classification

Econometric models on classes

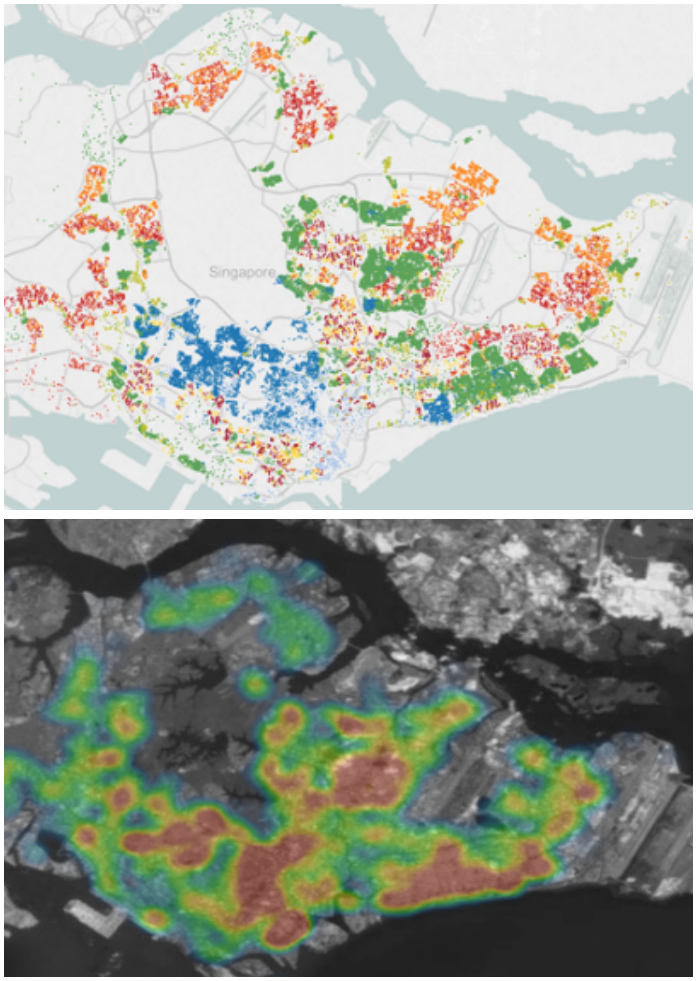
$$Y_i \sim \mathcal{B}(p_i) \text{ with } p_i = \frac{\exp[\beta_0 + \boldsymbol{x}_i^\mathsf{T}\boldsymbol{\beta}]}{1+\exp[\beta_0 + \boldsymbol{x}_i^\mathsf{T}\boldsymbol{\beta}]}$$

or on counts

$$Y_i \sim \mathcal{P}(\lambda_i) \text{ with } \lambda_i = \exp[\beta_0 + \boldsymbol{x}_i^\mathsf{T}\boldsymbol{\beta}]$$

- **(too) large datasets** $\boldsymbol{X}$ can be large (or complex)

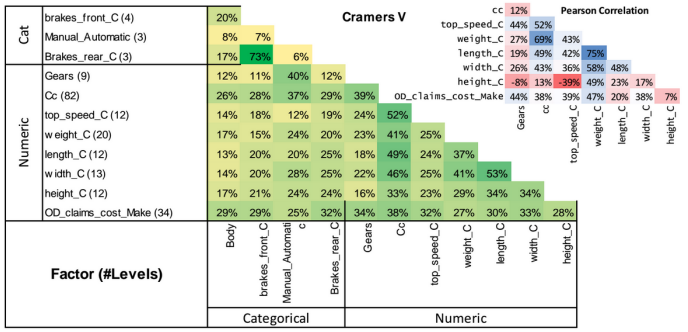
factors with a lot of modalities, spatial data, text information, etc.



Exemples of Actuarial Problems: Pricing and Classification

How to avoid overfit? How to group modalities? How to choose between (very) correlated features?

- **model selection** issues

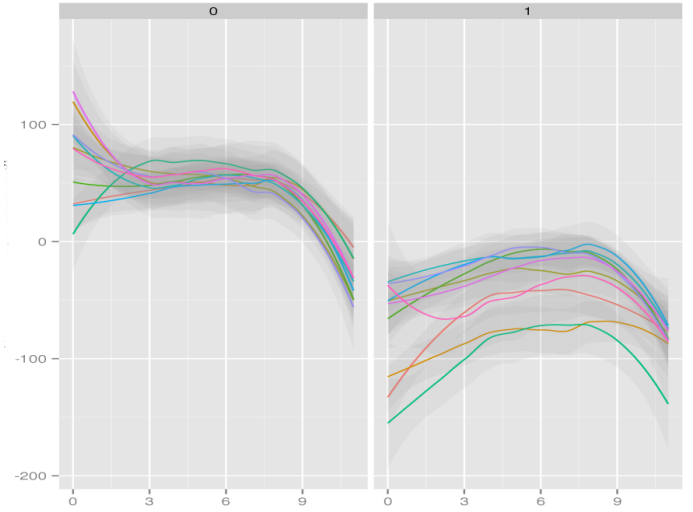


Historically Bailey (1963) 'margin method' $\widehat{\boldsymbol{n}} = \boldsymbol{r}\boldsymbol{c}^{\mathsf{T}}$, with row ($\boldsymbol{r}$) and column ($\boldsymbol{c}$) effects, and constraints

$$\sum_i n_{i,j} = \sum_i r_i \cdot c_j \text{ and } \sum_j n_{i,j} = \sum_j r_i \cdot c_j$$

Related to Poisson regression,

$$N \sim \mathcal{P}(\exp[\beta_0 + \boldsymbol{r}^{\mathsf{T}}\boldsymbol{\beta}_R + \boldsymbol{c}^{\mathsf{T}}\boldsymbol{\beta}_C])$$



Exemples of Actuarial Problems: Claims Reserving and Predictive Models

- **predictive modeling** issues

In all those cases, the goal is to get a predictive model, $\widehat{y} = \widehat{m}(\boldsymbol{x})$ given some features $\boldsymbol{x}$.
Recall that the main interest in insurance is either

- a probability $m(\boldsymbol{x}) = \mathbb{P}[Y = 1|\boldsymbol{X} = \boldsymbol{x}]$
- an expected value $m(\boldsymbol{x}) = \mathbb{E}[Y|\boldsymbol{X} = \boldsymbol{x}]$

but sometimes, we need the (conditiondal) distribution of $\widehat{Y}$.

Observed Increments, in Currency Units

AY/DY	0	1	2	3	4	5	6	7	8
1999	1217	1041	653	3	186	6	150	174	237
2000	1233	1374	134	235	24	167	166	280	249
2001	1264	1229	322	147	167	141	152	236	253
2002	1301	1538	270	214	158	185	180	265	284
2003	1527	1732	296	198	264	158	205	302	323
2004	1655	1683	261	160	189	155	201	297	318
2005	1753	1414	246	174	180	148	192	283	303
2006	1765	1237	328	170	176	144	187	276	296
2007	1900	1825	407	211	218	179	233	343	367

Observed Cumulative Data, in Currency Units

AY/DY	0	1	2	3	4	5	6	7	8
1999	1217	2258	2911	2915	3101	3107	3257	3431	3668
2000	1233	2607	2741	2977	3000	3167	3333	3613	3863
2001	1264	2493	2815	2962	3129	3270	3422	3658	3910
2002	1301	2839	3109	3322	3481	3666	3846	4111	4395
2003	1527	3259	3555	3753	4017	4175	4380	4681	5005
2004	1655	3338	3599	3759	3948	4103	4304	4601	4918
2005	1753	3166	3413	3587	3767	3915	4107	4390	4693
2006	1765	3001	3329	3499	3675	3819	4007	4283	4578
2007	1900	3725	4133	4344	4562	4741	4973	5316	5683

Development Factors

1.9603	1.1093	1.0511	1.0502	1.0393	1.0491	1.0689	1.0690	1

Source: Swedish data—Data from: Swedish Financial Supervisory Authority.
Note: The figures in gray are estimated using an ordinary development approach. AY: Accident year; DY: Development year. Currency in 1000 units.

History of Actuarial Models (in one slide)

Bailey (1963) or Taylor (1977) considered deterministic models, $n_{i,j} = r_i \cdot c_j$ or $n_{i,j} = r_i \cdot d_{i+j}$. Some additional constraints are given to get an identifiable model.

Then some stochastic version of those models were introduced, see Hachemeister (1975) or de Vylder (1985), e.g.

$$N_{i,j} \sim \mathcal{P}(\exp[\mathcal{P}(\exp[\beta_0 + \boldsymbol{R}^\mathsf{T}\boldsymbol{\beta}_R + \boldsymbol{C}^\mathsf{T}\boldsymbol{\beta}_C])$$

or

$$\log N_{i,j} \sim \mathcal{N}(\beta_0 + \boldsymbol{R}^\mathsf{T}\boldsymbol{\beta}_R + \boldsymbol{C}^\mathsf{T}\boldsymbol{\beta}_C, \sigma^2)$$

All those techniques are econometric-based techniques. Why not consider some statistical learning techniques?

Statistical Learning and Philosophical Issues

From *Machine Learning and Econometrics*, by Hal Varian :

"**Machine learning** use data to predict some variable as a function of other covariables,

- may, or may not, care about insight, importance, patterns
- may, or may not, care about inference (how y changes as some $\boldsymbol{x}$ change)

Econometrics use statistical methodes for prediction, inference and causal modeling of economic relationships

- hope for some sort of insight (inference is a goal)
- in particular, causal inference is goal for decision making."

$\rightarrow$ machine learning, 'new tricks for econometrics'

Statistical Learning and Philosophical Issues

Remark machine learning can also learn from econometrics, especially with non i.i.d. data (time series and panel data)

Remark machine learning can help to get better predictive models, given good datasets. No use on several data science issues (e.g. selection bias).

Machine Learning and 'Statistics'

Machine learning and statistics seem to be very similar, they share the same goals—they both focus on data modeling—but their methods are affected by their cultural differences.

"The goal for a statistician is to predict an interaction between variables with some degree of certainty (we are never 100% certain about anything). Machine learners, on the other hand, want to build algorithms that predict, classify, and cluster with the most accuracy, see Why a Mathematician, Statistician & Machine Learner Solve the Same Problem Differently

Machine learning methods are about algorithms, more than about asymptotic statistical properties.

Machine Learning and 'Statistics'

See also nonparametric inference: "Note that the non-parametric model is not none-parametric: parameters are determined by the training data, not the model. [...] non-parametric covers techniques that do not assume that the structure of a model is fixed. Typically, the model grows in size to accommodate the complexity of the data." see wikipedia

Validation is not based on mathematical properties, but on properties out of sample: we must use a training sample to train (estimate) model, and a testing sample to compare algorithms.

Goldilock Principle: the Mean-Variance Tradeoff

In statistics and in machine learning, there will be parameters and meta-parameters (or tunning parameters. The first ones are estimated, the second ones should be chosen.

See Hill estimator in extreme value theory. X has a Pareto distribution above some threshold u if

$$\mathbb{P}[X > x | X > u] = \left(\frac{u}{x}\right)^{\frac{1}{\xi}} \text{ for } x > u.$$

Given a sample $\boldsymbol{x}$, consider the Pareto-QQ plot, i.e. the scatterplot

$$\left\{ -\log\left(1 - \frac{i}{n+1}\right), \log x_{i:n} \right\}_{i=n-k,\cdots,n}$$

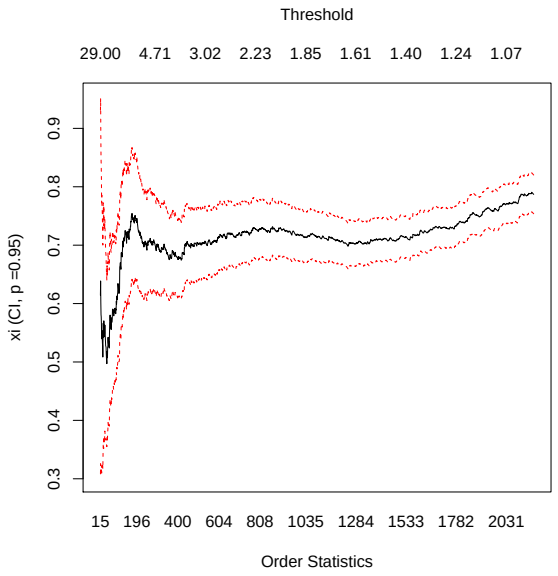
for points exceeding $X_{n-k:n}$.

Goldilock Principle: the Mean-Variance Tradeoff

The slope is ξ, i.e.

$$\log X_{n-i+1:n} \approx \log X_{n-k:n} + \xi\left(-\log\frac{i}{n+1} - \log\frac{n+1}{k+1}\right)$$

Hence, consider estimator $\widehat{\xi}_k = \dfrac{1}{k}\displaystyle\sum_{i=0}^{k-1} \log x_{n-i:n} - \log x_{n-k:n}$.



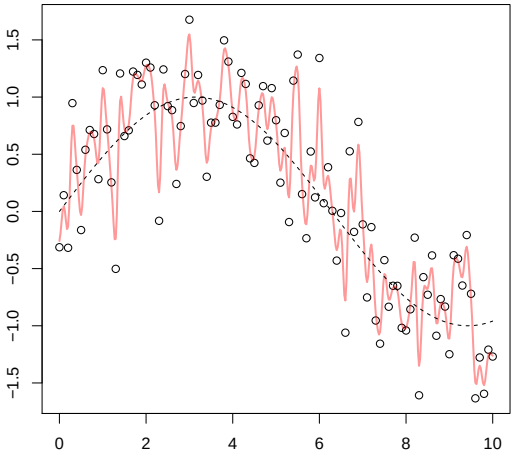
Standard mean-variance tradeoff,

- k large: bias too large, variance too small
- k small: variance too large, bias too small

Goldilock Principle: the Mean-Variance Tradeoff

Same holds in kernel regression, with bandwidth h (length of neighborhood)

$$\widehat{m}_h(x) = \frac{\sum_{i=1}^{n} K_h(x - x_i) \cdot y_i}{\sum_{i=1}^{n} K_h(x - x_i)}$$



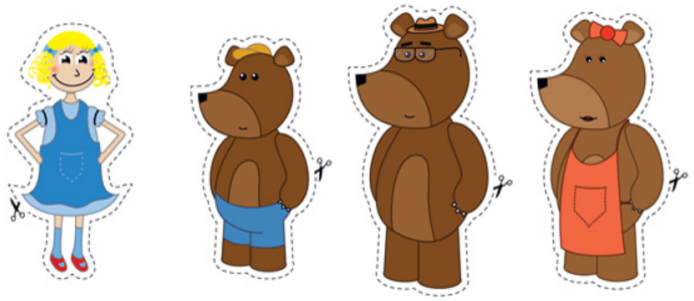
for some kernel $K(\cdot)$.

Standard mean-variance tradeoff,

- h large: bias too large, variance too small
- h small: variance too large, bias too small

Goldilock Principle: the Mean-Variance Tradeoff

More generally, we estimate $\widehat{\boldsymbol{\theta}}_h$ or $\widehat{m}_h(\cdot)$
Use the mean squared error for $\widehat{\boldsymbol{\theta}}_h$



$$\mathbb{E}\left[\left(\boldsymbol{\theta}-\widehat{\boldsymbol{\theta}}_h\right)^2\right]$$

or mean integrated squared error $\widehat{m}_h(\cdot)$,

$$\mathbb{E}\left[\int\left(m(\boldsymbol{x})-\widehat{m}_h(\boldsymbol{x})\right)^2 d\boldsymbol{x}\right]$$

In statistics, derive an asymptotic expression for these quantities, and find $h^\star$ that minimizes those.

Goldilock Principle: the Mean-Variance Tradeoff

In classical statistics, the MISE can be approximated by

$$\frac{h^4}{4}\left(\int x^2K(\boldsymbol{x})d\boldsymbol{x}\right)^2\int\left(m''(\boldsymbol{x})+2m'(\boldsymbol{x})\frac{f'(\boldsymbol{x})}{f(\boldsymbol{x})}\right)d\boldsymbol{x}+\frac{1}{nh}\sigma^2\int K^2(x)dx\int\frac{d\boldsymbol{x}}{f(\boldsymbol{x})}$$

where f is the density of $\boldsymbol{x}$'s. Thus the optimal h is

$$h^\star=n^{-\frac{1}{5}}\left(\frac{\sigma^2\int K^2(x)dx\int\frac{d\boldsymbol{x}}{f(\boldsymbol{x})}}{\left(\int x^2K(\boldsymbol{x})d\boldsymbol{x}\right)^2\int\left(\int m''(\boldsymbol{x})+2m'(\boldsymbol{x})\frac{f'(\boldsymbol{x})}{f(\boldsymbol{x})}\right)^2d\boldsymbol{x}}\right)^{\frac{1}{5}}$$

(hard to get a simple rule of thumb... up to a constant, $h^\star\sim n^{-\frac{1}{5}}$)

In statistics learning, use bootstrap, or cross-validation to get an optimal h...

Randomization is too important to be left to chance!

Consider some sample $\boldsymbol{x} = (x_1, \cdots, x_n)$ and some statistics $\widehat{\theta}$. Set $\widehat{\theta}_n = \widehat{\theta}(\boldsymbol{x})$

Jackknife used to reduce bias: set $\widehat{\theta}_{(-i)} = \widehat{\theta}(\boldsymbol{x}_{(-i)})$, and $\tilde{\theta} = \dfrac{1}{n}\sum_{i=1}^{n} \widehat{\theta}_{(-i)}$

$$\text{If } \mathbb{E}(\widehat{\theta}_n) = \theta + O(n^{-1}) \text{ then } \mathbb{E}(\tilde{\theta}_n) = \theta + O(n^{-2}).$$

See also leave-one-out cross validation, for $\widehat{m}(\cdot)$

$$\text{mse} = \frac{1}{n}\sum_{i=1}^{n}[y_i - \widehat{m}_{(-i)}(x_i)]^2$$

Boostrap estimate is based on bootstrap samples: set $\widehat{\theta}_{(b)} = \widehat{\theta}(\boldsymbol{x}_{(b)})$, and $\tilde{\theta} = \dfrac{1}{n}\sum_{i=1}^{n} \widehat{\theta}_{(b)}$, where $\boldsymbol{x}_{(b)}$ is a vector of size n, where values are drawn from $\{x_1, \cdots, x_n\}$, with replacement. And then use the law of large numbers...

See Efron (1979).

Statistical Learning and Philosophical Issues

From $(y_i, \boldsymbol{x}_i)$, there are different stories behind, see Freedman (2005)

- the causal story : $x_{j,i}$ is usually considered as independent of the other covariates $x_{k,i}$. For all possible $\boldsymbol{x}$, that value is mapped to $m(\boldsymbol{x})$ and a noise is atatched, ε. The goal is to recover $m(\cdot)$, and the residuals are just the difference between the response value and $m(\boldsymbol{x})$.
- the conditional distribution story : for a linear model, we usually say that Y given $\boldsymbol{X} = \boldsymbol{x}$ is a $\mathcal{N}(m(\boldsymbol{x}), \sigma^2)$ distribution. $m(\boldsymbol{x})$ is then the conditional mean. Here $m(\cdot)$ is assumed to really exist, but no causal assumption is made, only a conditional one.
- the explanatory data story : there is no model, just data. We simply want to summarize information contained in $\boldsymbol{x}$'s to get an accurate summary, close to the response (i.e. $\min\{\ell(\boldsymbol{y}_i, m(\boldsymbol{x}_i))\}$) for some loss function ℓ.

Machine Learning vs. Statistical Modeling

In machine learning, given some dataset $(\boldsymbol{x}_i, y_i)$, solve

$$\widehat{m}(\cdot) = \underset{m(\cdot)\in\mathcal{F}}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} \ell(y_i, m(\boldsymbol{x}_i)) \right\}$$

for some loss functions $\ell(\cdot,\cdot)$.

In statistical modeling, given some probability space $(\Omega, \mathcal{A}, \mathbb{P})$, assume that y_i are realization of i.i.d. variables Y_i (given $\boldsymbol{X}_i = \boldsymbol{x}_i$) with distribution F_i. Then solve

$$\widehat{m}(\cdot) = \underset{m(\cdot)\in\mathcal{F}}{\operatorname{argmax}} \{\log \mathcal{L}(m(\boldsymbol{x}); \boldsymbol{y})\} = \underset{m(\cdot)\in\mathcal{F}}{\operatorname{argmax}} \left\{ \sum_{i=1}^{n} \log f(y_i; m(\boldsymbol{x}_i)) \right\}$$

where $\log \mathcal{L}$ denotes the log-likelihood.

Loss Functions

Fitting criteria are based on loss functions (also called cost functions). For a quantitative response, a popular one is the quadratic loss, $\ell(y, m(\boldsymbol{x})) = [y - m(\boldsymbol{x})]^2$.

Recall that

$$\begin{cases} \mathbb{E}(Y) = \underset{m\in\mathbb{R}}{\text{argmin}}\{\|Y - m\|_{\ell_2}\} = \underset{m\in\mathbb{R}}{\text{argmin}}\{\mathbb{E}\left([Y - m]^2\right)\} \\ \text{Var}(Y) = \underset{m\in\mathbb{R}}{\min}\{\mathbb{E}\left([Y - m]^2\right)\} = \mathbb{E}\left([Y - \mathbb{E}(Y)]^2\right) \end{cases}$$

The empirical version is

$$\begin{cases} \overline{y} = \underset{m\in\mathbb{R}}{\text{argmin}}\{\sum_{i=1}^{n} \frac{1}{n}[y_i - m]^2\} \\ s^2 = \underset{m\in\mathbb{R}}{\min}\{\sum_{i=1}^{n} \frac{1}{n}[y_i - m]^2\} = \sum_{i=1}^{n} \frac{1}{n}[y_i - \overline{y}]^2 \end{cases}$$

Model Evaluation

In linear models, the R^2 is defined as the proportion of the variance of the the response y that can be obtained using the predictors.

But maximizing the R^2 usually yields overfit (or unjustified optimism in Berk (2008)).

In linear models, consider the adjusted R^2,

$$\overline{R}^2 = 1 - [1 - R^2]\frac{n-1}{n-p-1}$$

where p is the number of parameters, or more generally trace($\boldsymbol{S}$) when some smoothing matrix is considered

$$\widehat{y} = \widehat{m}(\boldsymbol{x}) = \sum_{i=1}^{n} \boldsymbol{S}_{\boldsymbol{x},i} y_i = \boldsymbol{S}_{\boldsymbol{x}}^{\mathsf{T}} \boldsymbol{y}$$

where $\boldsymbol{S}_{\boldsymbol{x}}$ is some vector of weights (called smoother vector), related to a $n \times n$ smoother matrix, $\widehat{\boldsymbol{y}} = \boldsymbol{S}\boldsymbol{y}$ where prediction is done at points $\boldsymbol{x}_i$'s.

Model Evaluation

Alternatives are based on the Akaike Information Criterion (AIC) and the Bayesian Information Criterion (BIC), based on a penalty imposed on some criteria (the logarithm of the variance of the residuals),

$$AIC = \log\left(\frac{1}{n}\sum_{i=1}^{n}[y_i - \widehat{y}_i]^2\right) + \frac{2p}{n}$$

$$BIC = \log\left(\frac{1}{n}\sum_{i=1}^{n}[y_i - \widehat{y}_i]^2\right) + \frac{\log(n)p}{n}$$

In a more general context, replace p by $\text{trace}(\boldsymbol{S})$.

Model Evaluation

One can also consider the expected prediction error (with a probabilistic model)

$$\mathbb{E}[\ell(Y, \widehat{m}(\boldsymbol{X})]$$

We cannot claim (using the law of large number) that

$$\frac{1}{n}\sum_{i=1}^{n} \ell(y_i, \widehat{m}(\boldsymbol{x}_i)) \overset{a.s.}{\nrightarrow} \mathbb{E}[\ell(Y, m(\boldsymbol{X})]$$

since $\widehat{m}$ depends on $(y_i, \boldsymbol{x}_i)$'s.

Natural option : use two (random) samples, a training one and a validation one.

Alternative options, use cross-validation, leave-one-out or k-fold.

Underfit / Overfit and Variance - Mean Tradeoff

Goal in predictive modeling: reduce uncertainty in our predictions.

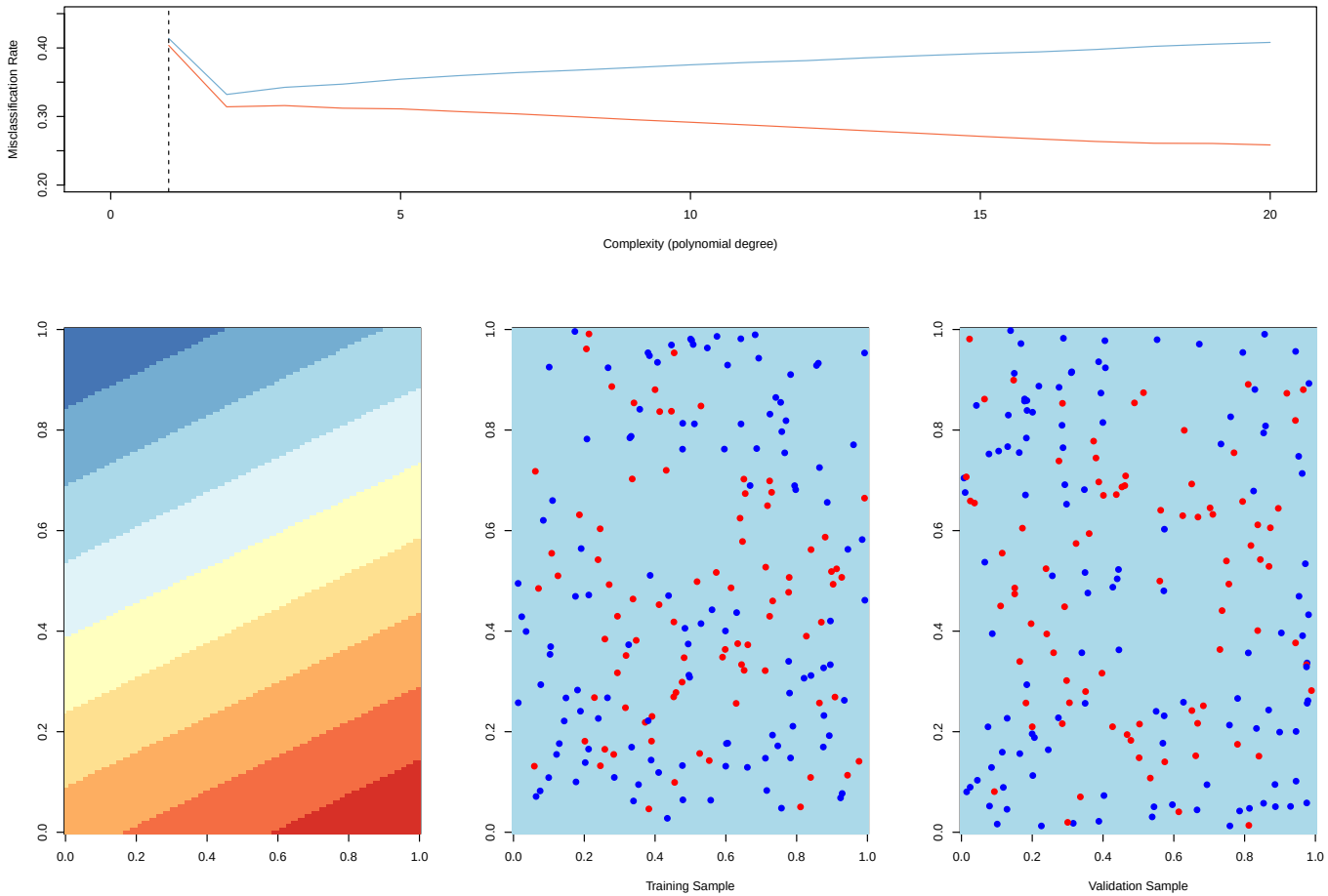
Need more data to get a better knowledge.

Unfortunately, reducing the error of the prediction on a dataset does not generally give a good generalization performance

$\longrightarrow$ need a training and a validation dataset

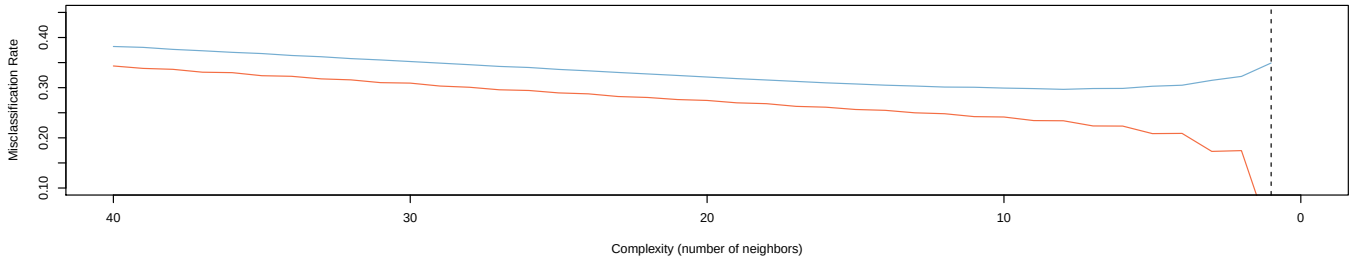
Overfit, Training vs. Validation and Complexity (Vapnik Dimension)

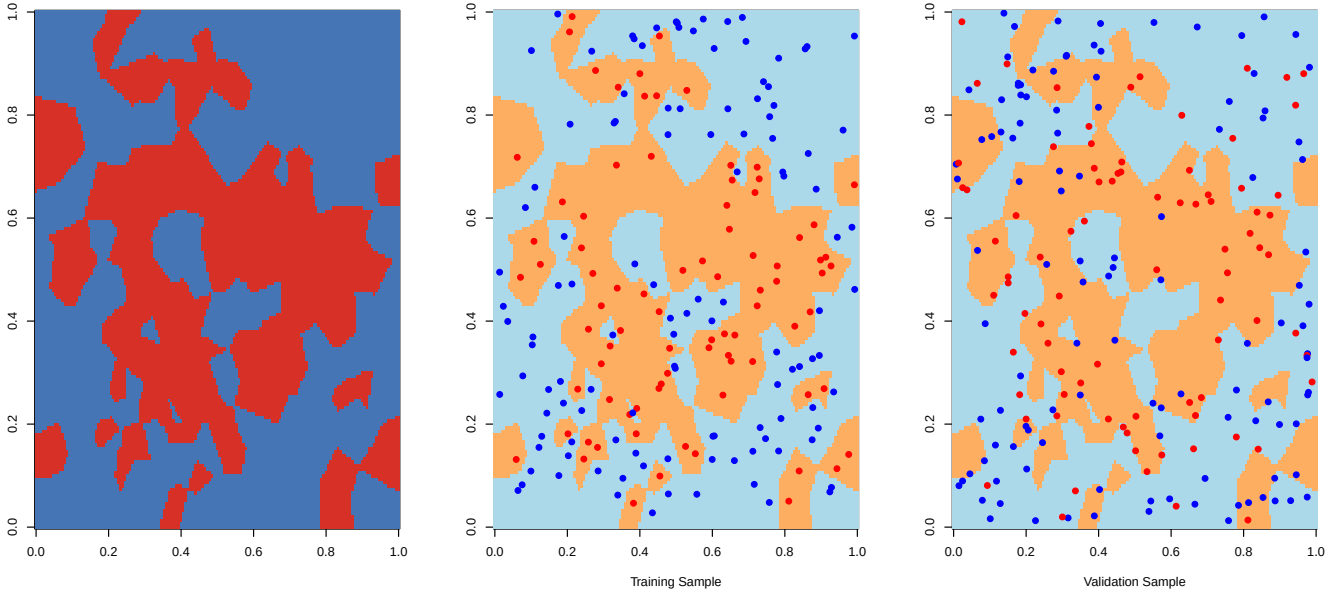
complexity $\longleftrightarrow$ polynomial degree



Overfit, Training vs. Validation and Complexity (Vapnik Dimension)

complexity $\longleftrightarrow$ number of neighbors (k)

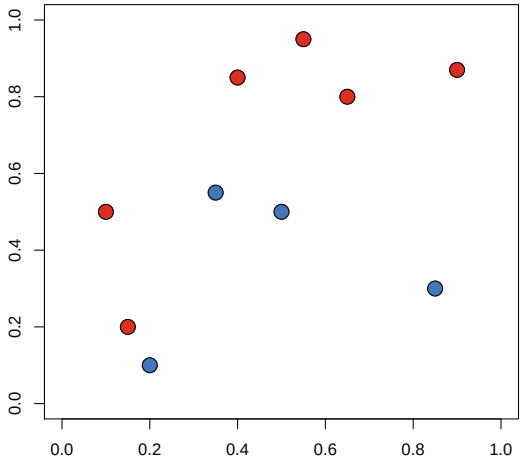




Logistic Regression

Assume that $\mathbb{P}(Y_i = 1) = \pi_i$,

$$\text{logit}(\pi_i) = \boldsymbol{x}_i^\mathsf{T}\boldsymbol{\beta}, \text{ where } \text{logit}(\pi_i) = \log\left(\frac{\pi_i}{1-\pi_i}\right),$$



or

$$\pi_i = \text{logit}^{-1}(\boldsymbol{x}_i^\mathsf{T}\boldsymbol{\beta}) = \frac{\exp[\boldsymbol{x}_i^\mathsf{T}\boldsymbol{\beta}]}{1+\exp[\boldsymbol{x}_i^\mathsf{T}\boldsymbol{\beta}]}.$$

The log-likelihood is

$$\log\mathcal{L}(\boldsymbol{\beta}) = \sum_{i=1}^{n} y_i\log(\pi_i)+(1-y_i)\log(1-\pi_i) = \sum_{i=1}^{n} y_i\log(\pi_i(\boldsymbol{\beta}))+(1-y_i)\log(1-\pi_i(\boldsymbol{\beta}))$$

and the first order conditions are solved numerically

$$\frac{\partial\log\mathcal{L}(\boldsymbol{\beta})}{\partial\beta_k} = \sum_{i=1}^{n} X_{k,i}[y_i-\pi_i(\boldsymbol{\beta})] = 0.$$

Predictive Classifier

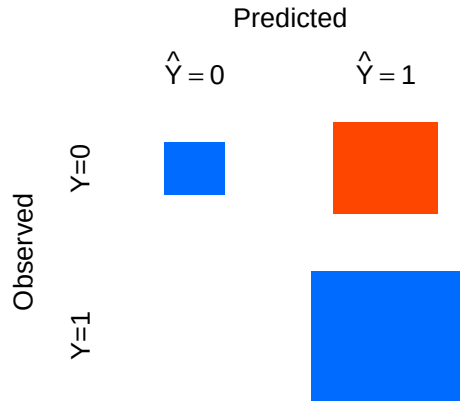
To go from a score

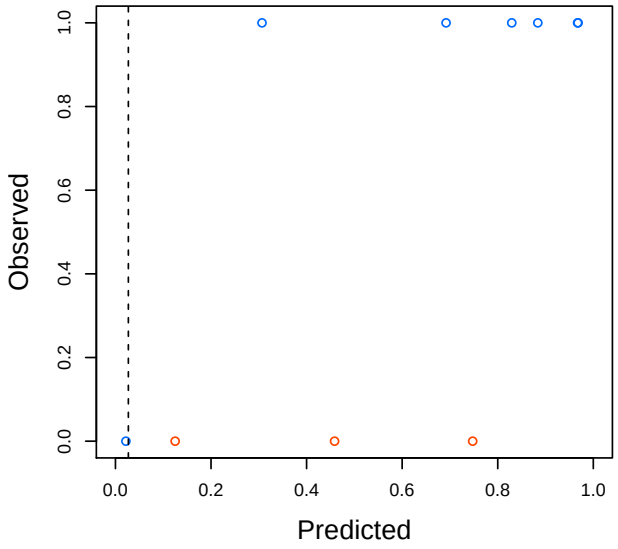
$$\widehat{s}(\boldsymbol{x}) = \frac{\exp[\boldsymbol{x}^{\mathsf{T}}\widehat{\boldsymbol{\beta}}]}{1+\exp[\boldsymbol{x}^{\mathsf{T}}\widehat{\boldsymbol{\beta}}]}$$

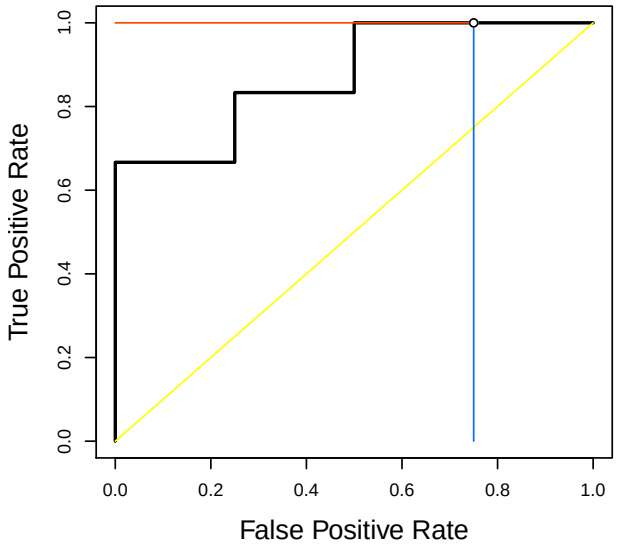
to a class:

if $\widehat{s}(\boldsymbol{x}) > s$, then $\widehat{Y}(\boldsymbol{x}) = 1$ (or •) and $\widehat{s}(\boldsymbol{x}) \leq s$, then $\widehat{Y}(\boldsymbol{x}) = 0$ (or •).

Plot $TP(s) = \mathbb{P}[\widehat{Y} = 1|Y = 1]$ against $FP(s) = \mathbb{P}[\widehat{Y} = 1|Y = 0]$







Why a Logistic and not a Probit Regression?

Bliss (1934) suggested a model such that

$$\mathbb{P}(Y=1|\boldsymbol{X}=\boldsymbol{x})=H(\boldsymbol{x}^{\mathsf{T}}\boldsymbol{\beta}) \text{ where } H(\cdot)=\Phi(\cdot)$$

the c.d.f. of the $\mathcal{N}(0,1)$ distribution. This is the probit model. This yields a latent model, $y_i=\mathbf{1}(y_i^\star>0)$ where

$$y_i^\star=\boldsymbol{x}_i^{\mathsf{T}}\boldsymbol{\beta}+\varepsilon_i \text{ is a nonobservable score.}$$

In the logistic regression, we model the odds ratio,

$$\frac{\mathbb{P}(Y=1|\boldsymbol{X}=\boldsymbol{x})}{\mathbb{P}(Y\neq 1|\boldsymbol{X}=\boldsymbol{x})}=\exp[\boldsymbol{x}^{\mathsf{T}}\boldsymbol{\beta}]$$

$$\mathbb{P}(Y=1|\boldsymbol{X}=\boldsymbol{x})=H(\boldsymbol{x}^{\mathsf{T}}\boldsymbol{\beta}) \text{ where } H(\cdot)=\frac{\exp[\cdot]}{1+\exp[\cdot]}$$

which is the c.d.f. of the logistic variable, see Verhulst (1845)

Table 3.2 Transformation of percentages to probits

%	0	1	2	3	4	5	6	7	8	9
0	—	2.67	2.95	3.12	3.25	3.36	3.45	3.52	3.59	3.66
10	3.72	3.77	3.82	3.87	3.92	3.96	4.01	4.05	4.08	4.12
20	4.16	4.19	4.23	4.26	4.29	4.33	4.36	4.39	4.42	4.45
30	4.48	4.50	4.53	4.56	4.59	4.61	4.64	4.67	4.69	4.72
40	4.75	4.77	4.80	4.82	4.85	4.87	4.90	4.92	4.95	4.97
50	5.00	5.03	5.05	5.08	5.10	5.13	5.15	5.18	5.20	5.23
60	5.25	5.28	5.31	5.33	5.36	5.39	5.41	5.44	5.47	5.50
70	5.52	5.55	5.58	5.61	5.64	5.67	5.71	5.74	5.77	5.81
80	5.84	5.88	5.92	5.95	5.99	6.04	6.08	6.13	6.18	6.23
90	6.28	6.34	6.41	6.48	6.55	6.64	6.75	6.88	7.05	7.33
—	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
99	7.33	7.37	7.41	7.46	7.51	7.58	7.65	7.75	7.88	8.09

Soit p la population : représentons par dp l'accroissement infiniment petit qu'elle reçoit pendant un temps infiniment court dt. Si la population croissait en progression géométrique, nous aurions l'équation $\frac{dp}{dt}=mp$. Mais comme la vitesse d'accroissement de la population est retardée par l'augmentation même du nombre des habitans, nous devrons retrancher de mp une fonction inconnue de p; de manière que la formule à intégrer deviendra

$$\frac{dp}{dt}=mp-\varphi(p).$$

L'hypothèse la plus simple que l'on puisse faire sur la forme de la fonction φ, est de supposer $\varphi(p)=np^2$. On trouve alors pour intégrale de l'équation ci-dessus

$$t=\frac{1}{m}[\log.p-\log.(m-np)]+\text{constante},$$

et il suffira de trois observations pour déterminer les deux coefficiens constans m et n et la constante arbitraire.

En résolvant la dernière équation par rapport à p, il vient

$$p=\frac{mp'e^{mt}}{np'e^{mt}+m-np'} \quad \ldots\ldots \quad (1)$$

en désignant par p' la population qui répond à $t=o$, et par e la base des logarithmes népériens. Si l'on fait $t=\infty$, on voit que la valeur de p correspondante est $P=\frac{m}{n}$. Telle est donc *la limite supérieure de la population*.

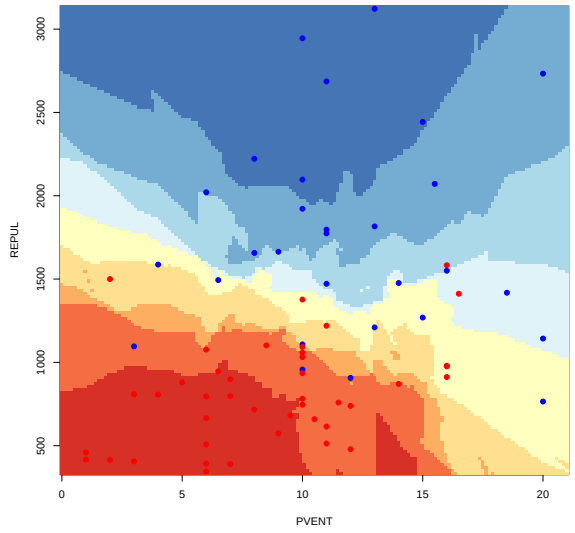
k-Nearest Neighbors (a.k.a. k-NN)

In pattern recognition, the k-Nearest Neighbors algorithm (or k-NN for short) is a non-parametric method used for classification and regression. (Source: wikipedia).

$$\mathbb{E}[Y|\boldsymbol{X}=\boldsymbol{x}] \sim \frac{1}{k} \sum_{d(\boldsymbol{x_i},\boldsymbol{x}) \text{ small}} y_i$$

For k-Nearest Neighbors, the class is usually the majority vote of the k closest neighbors of $\boldsymbol{x}$.

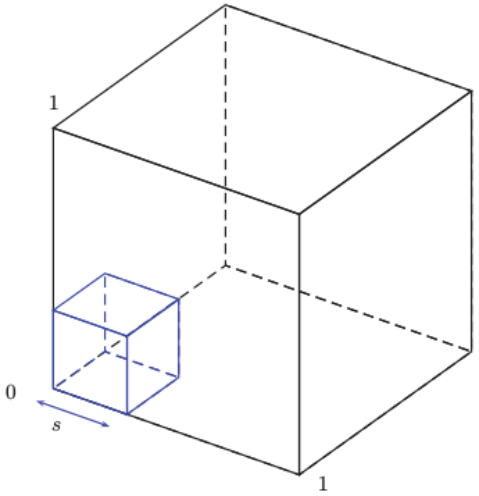
Distance $d(\cdot,\cdot)$ should not be sensitive to units: normalize by standard deviation

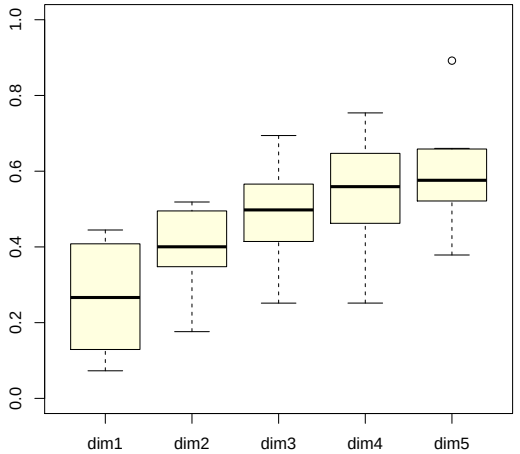


k-Nearest Neighbors and Curse of Dimensionality

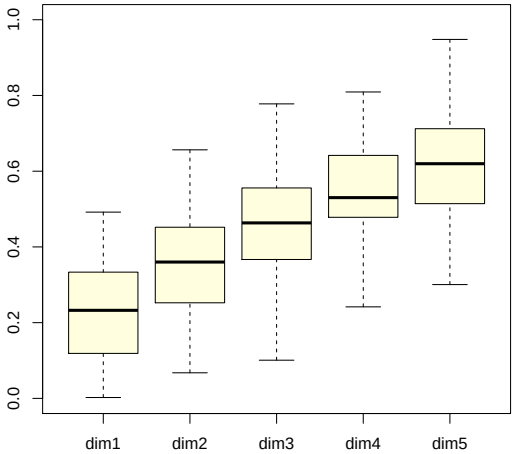
The higher the dimension, the larger the distance to the closest neigbbor

$$\min_{i\in\{1,\cdots,n\}}\{d(\boldsymbol{a},\boldsymbol{x}_i)\},\boldsymbol{x}_i\in\mathbb{R}^d.$$





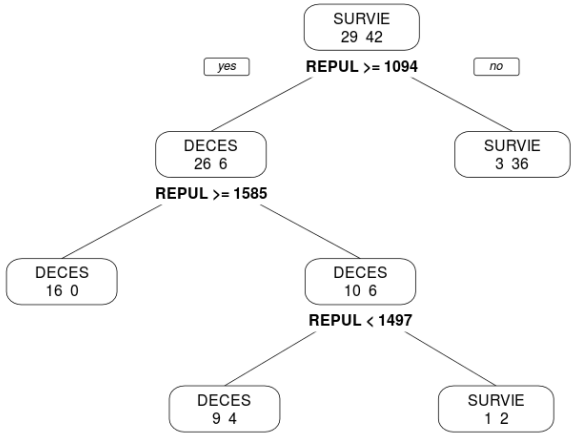
$n=10$

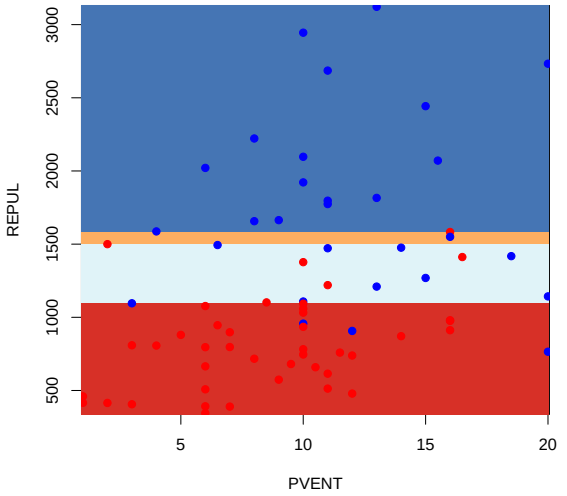


$n=100$

Classification (and Regression) Trees, CART

one of the predictive modelling approaches used in statistics, data mining and machine learning [...] In tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels. (Source: wikipedia).





Bagging

Bootstrapped Aggregation (Bagging) , is a machine learning ensemble meta-algorithm designed to improve the stability and accuracy of machine learning algorithms used in statistical classification (Source: wikipedia).

It is an ensemble method that creates multiple models of the same type from different sub-samples of the same dataset [boostrap]. The predictions from each separate model are combined together to provide a superior result [aggregation].

→ can be used on any kind of model, but interesting for trees, see Breiman (1996)

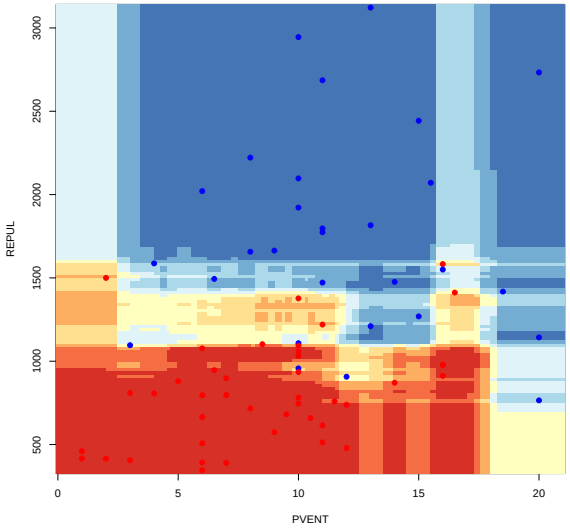
Boostrap can be used to define the concept of margin,

$$\text{margin}_i = \frac{1}{B}\sum_{b=1}^{B} \mathbf{1}(\widehat{y}_i = y_i) - \frac{1}{B}\sum_{b=1}^{B} \mathbf{1}(\widehat{y}_i \neq y_i)$$

Remark Probability that ith raw is not selection $(1-n^{-1})^n \to e^{-1} \sim 36.8\%$, cf training / validation samples (2/3-1/3)

Random Forests

Strictly speaking, when boostrapping among observations, and aggregating, we use a bagging algorithm.
In the random forest algorithm, we combine Breiman's bagging idea and the random selection of features, introduced independently by Ho (1995) and Amit & Geman (1997).



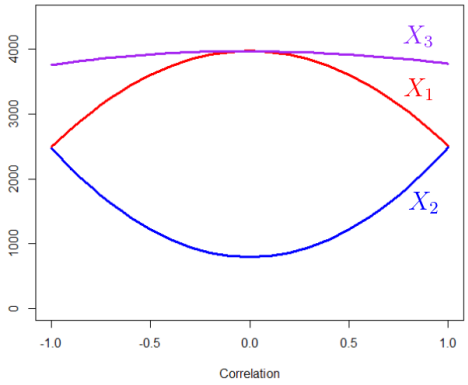
Bagging, Forests, and Variable Importance

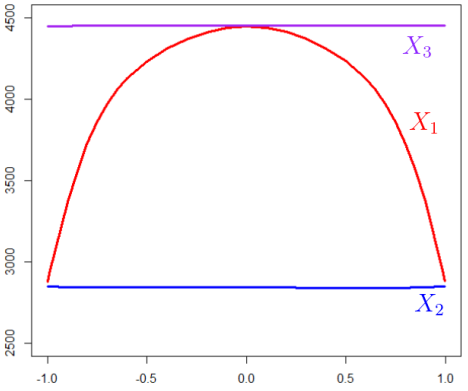
Given some random forest with M trees, set

$$VI(X_k) = \frac{1}{M} \sum_m \sum_t \frac{N_t}{N} \Delta i(t)$$

where the first sum is over all trees, and the second one is over all nodes where the split is done based on variable X_k. But difficult to interprete with correlated features. Considere model $y = \beta_0 + \beta_1 x_1 + \beta_3 x_3 + \varepsilon$, and we consider a model based on $\boldsymbol{x} = (x_1, x_2, x_3)$ where x_1 and x_2 are correlated.

Compare AIC vs. Variable Importance as a function of r



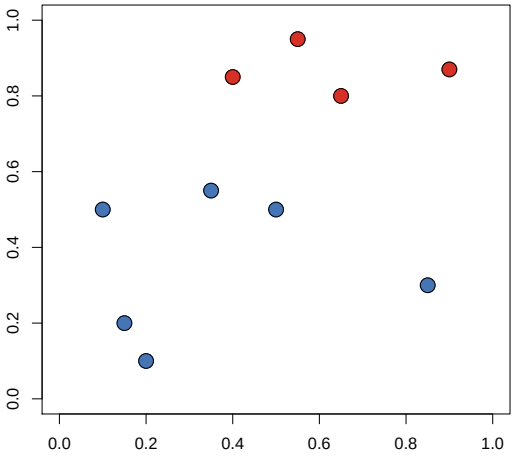


Support Vector Machine

SVMs were developed in the 90's based on previous work, from Vapnik & Lerner (1963), see Vailant (1984)

Assume that points are linearly separable, i.e. there is $\boldsymbol{\omega}$ and b such that

$$Y = \begin{cases} +1 \text{ if } \boldsymbol{\omega}^\mathsf{T}\boldsymbol{x} + b > 0 \\ -1 \text{ if } \boldsymbol{\omega}^\mathsf{T}\boldsymbol{x} + b < 0 \end{cases}$$



Problem: infinite number of solutions, need a good one, that separate the data, (somehow) far from the data.

The distance from $\boldsymbol{x}_0$ to the hyperplane $\boldsymbol{\omega}^\mathsf{T}\boldsymbol{x} + b$ is

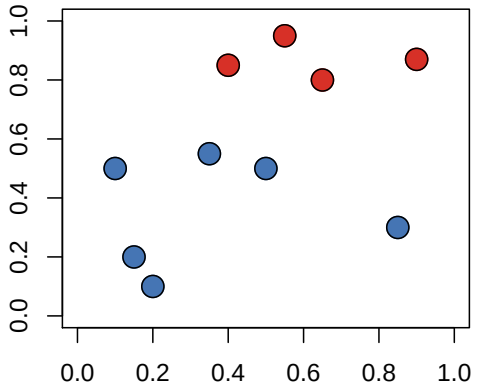
$$d(\boldsymbol{x}_0, H_{\boldsymbol{\omega},b}) = \frac{\boldsymbol{\omega}^\mathsf{T}\boldsymbol{x}_0 + b}{\|\boldsymbol{\omega}\|}$$

Support Vector Machine

Define support vectors as observations such that

$$|\boldsymbol{\omega}^{\mathsf{T}}\boldsymbol{x}_i + b| = 1$$

The margin is the distance between hyperplanes defined by support vectors.



The distance from support vectors to $H_{\boldsymbol{\omega},b}$ is $\|\boldsymbol{\omega}\|^{-1}$, and the margin is then $2\|\boldsymbol{\omega}\|^{-1}$.

$\longrightarrow$ the algorithm is to minimize the inverse of the margins s.t. $H_{\boldsymbol{\omega},b}$ separates ± 1 points, i.e.

$$\min\left\{\frac{1}{2}\boldsymbol{\omega}^{\mathsf{T}}\boldsymbol{\omega}\right\} \text{ s.t. } y_i(\boldsymbol{\omega}^{\mathsf{T}}\boldsymbol{x}_i + b) \geq 1, \ \forall i.$$

Support Vector Machine

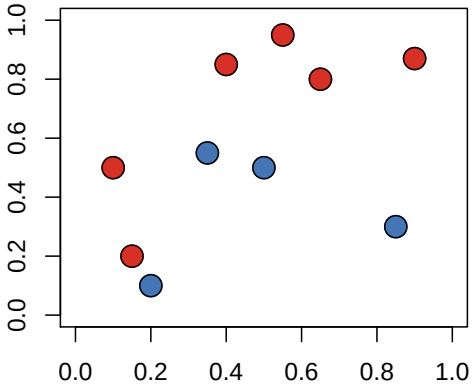
Now, what about the non-separable case?

Here, we cannot have $y_i(\boldsymbol{\omega}^\mathsf{T}\boldsymbol{x}_i + b) \geq 1 \ \forall i$.

$\longrightarrow$ introduce slack variables,

$$\begin{cases} \boldsymbol{\omega}^\mathsf{T}\boldsymbol{x}_i + b \geq +1 - \xi_i \text{ when } y_i = +1 \\ \boldsymbol{\omega}^\mathsf{T}\boldsymbol{x}_i + b \leq -1 + \xi_i \text{ when } y_i = -1 \end{cases}$$

where $\xi_i \geq 0 \ \forall i$. There is a classification error when $\xi_i > 1$.



The idea is then to solve

$$\min\left\{\frac{1}{2}\boldsymbol{\omega}^\mathsf{T}\boldsymbol{\omega} + C\mathbf{1}^\mathsf{T}\mathbf{1}_{\boldsymbol{\xi}>1}\right\}, \text{ instead of } \min\left\{\frac{1}{2}\boldsymbol{\omega}^\mathsf{T}\boldsymbol{\omega}\right\}$$

Support Vector Machines, with a Linear Kernel

So far, $d(\boldsymbol{x}_0, H_{\boldsymbol{\omega},b}) = \min_{\boldsymbol{x}\in H_{\boldsymbol{\omega},b}} \{\|\boldsymbol{x}_0 - \boldsymbol{x}\|_{\ell_2}\}$

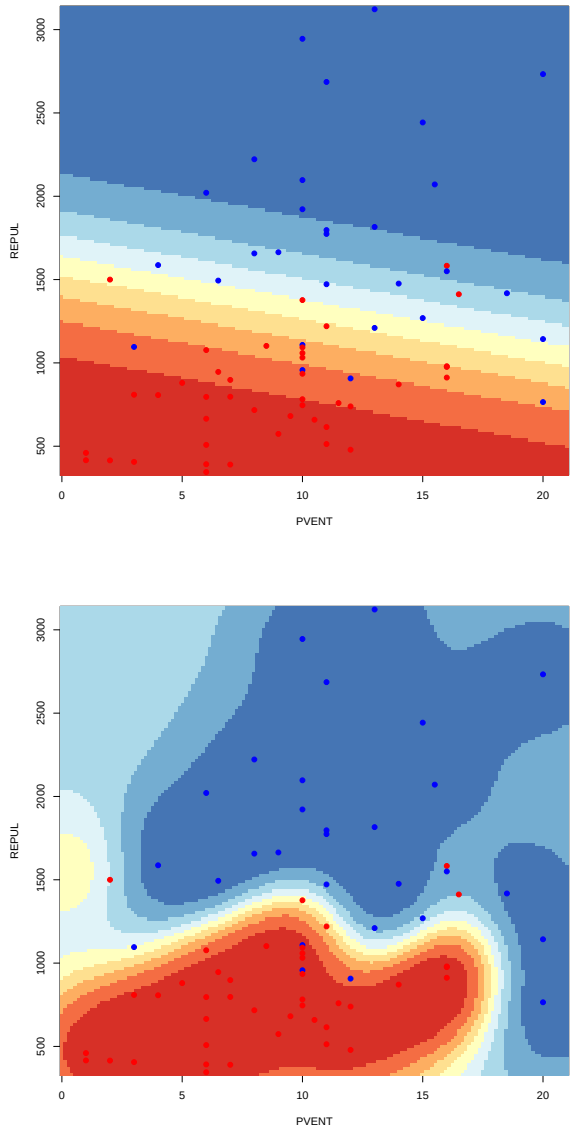
where $\|\cdot\|_{\ell_2}$ is the Euclidean (ℓ_2) norm,

$$\begin{aligned}\|\boldsymbol{x}_0 - \boldsymbol{x}\|_{\ell_2} &= \sqrt{(\boldsymbol{x}_0 - \boldsymbol{x})\cdot(\boldsymbol{x}_0 - \boldsymbol{x})} \\ &= \sqrt{\boldsymbol{x}_0\cdot\boldsymbol{x}_0 - 2\boldsymbol{x}_0\cdot\boldsymbol{x} + \boldsymbol{x}\cdot\boldsymbol{x}}\end{aligned}$$

More generally, $d(\boldsymbol{x}_0, H_{\boldsymbol{\omega},b}) = \min_{\boldsymbol{x}\in H_{\boldsymbol{\omega},b}} \{\|\boldsymbol{x}_0 - \boldsymbol{x}\|_{k}\}$

where $\|\cdot\|_{k}$ is some kernel-based norm,

$$\|\boldsymbol{x}_0 - \boldsymbol{x}\|_{k} = \sqrt{k(\boldsymbol{x}_0,\boldsymbol{x}_0) - 2k(\boldsymbol{x}_0,\boldsymbol{x}) + k(\boldsymbol{x}\cdot\boldsymbol{x})}$$



Regression?

In statistics, regression analysis is a statistical process for estimating the relationships among variables [...] In a narrower sense, regression may refer specifically to the estimation of continuous response variables, as opposed to the discrete response variables used in classification. (Source: wikipedia).

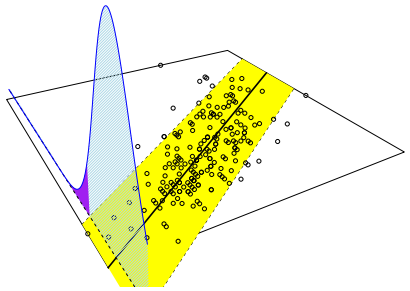
Here regression is opposed to classification (as in the CART algorithm). y is either a continuous variable $y \in \mathbb{R}$ or a counting variable $y \in \mathbb{N}$.

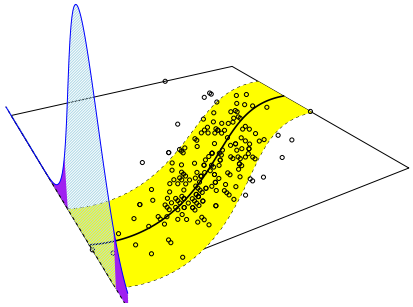
In many cases in econometric and actuarial literature we *simply* want a good fit for the conditional expectation, $\mathbb{E}[Y|\boldsymbol{X} = \boldsymbol{x}]$.

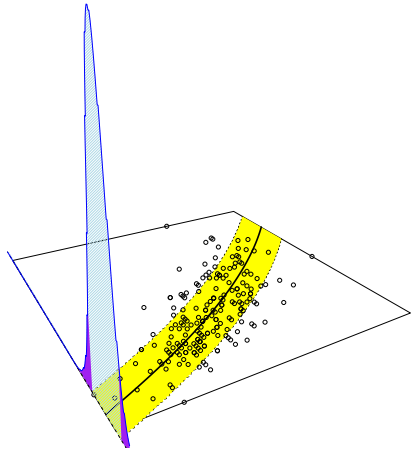
Linear, Non-Linear and Generalized Linear

$$(Y|\boldsymbol{X}=\boldsymbol{x}) \sim \mathcal{N}(\theta_{\boldsymbol{x}}, \sigma^2) \quad (Y|\boldsymbol{X}=\boldsymbol{x}) \sim \mathcal{N}(\theta_{\boldsymbol{x}}, \sigma^2) \quad (Y|\boldsymbol{X}=\boldsymbol{x}) \sim \mathcal{L}(\theta_{\boldsymbol{x}}, \varphi)$$

$$\mathbb{E}[Y|\boldsymbol{X}=\boldsymbol{x}] = \boldsymbol{x}^{\mathsf{T}}\boldsymbol{\beta} \qquad \mathbb{E}[Y|\boldsymbol{X}=\boldsymbol{x}] = h(\boldsymbol{x}) \qquad \mathbb{E}[Y|\boldsymbol{X}=\boldsymbol{x}] = h(\boldsymbol{x})$$







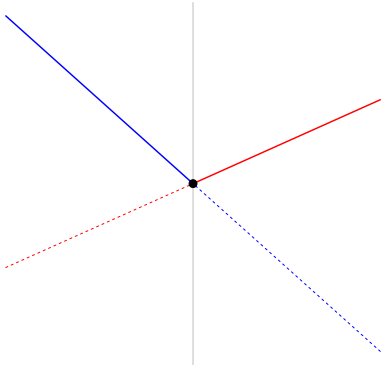
Regression Smoothers, *natura non facit saltus*

In statistical learning procedures, a key role is played by basis functions. We will see that it is common to assume that

$$m(\boldsymbol{x}) = \sum_{j=0}^{k} \beta_j h_j(\boldsymbol{x}),$$

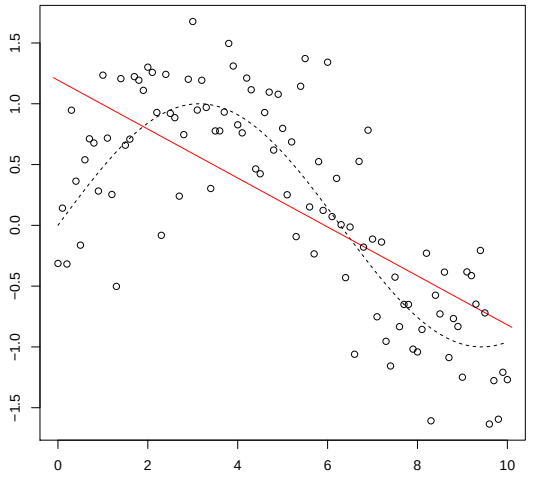
where h_0 is usually a constant function and h_j defined basis functions.

For instance, $h_m(x) = x^j$ for a polynomial expansion with a single predictor, or $h_j(x) = (x - s_j)_+$ for some knots s_j's (for linear splines, but one can consider quadratic or cubic ones).



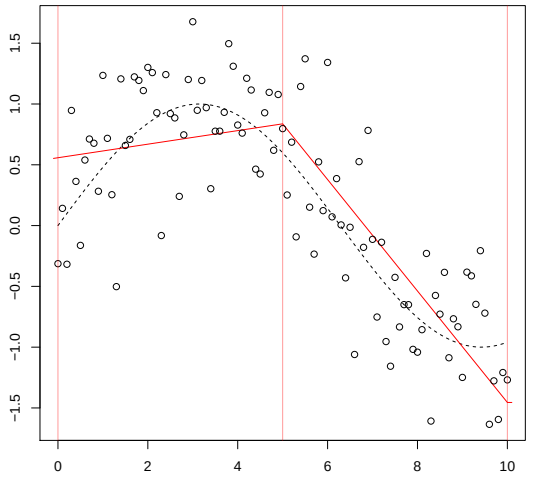
Regression Smoothers: Polynomial or Spline

Stone-Weiestrass theorem every continuous function defined on a closed interval $[a, b]$ can be uniformly approximated as closely as desired by a polynomial function



Use also spline functions, e.g. piecewise linear

$$h(x) = \beta_0 + \sum_{j=1}^{k} \beta_j (x - s_j)_+$$



Linear Model

Consider some linear model $y_i = \boldsymbol{x}_i^\mathsf{T}\boldsymbol{\beta} + \varepsilon_i$ for all $i = 1, \cdots, n$.

Assume that ε_i are i.i.d. with $\mathbb{E}(\varepsilon) = 0$ (and finite variance). Write

$$\underbrace{\begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}}_{\boldsymbol{y}, n\times 1} = \underbrace{\begin{pmatrix} 1 & x_{1,1} & \cdots & x_{1,k} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n,1} & \cdots & x_{n,k} \end{pmatrix}}_{\boldsymbol{X}, n\times(k+1)} \underbrace{\begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{pmatrix}}_{\boldsymbol{\beta}, (k+1)\times 1} + \underbrace{\begin{pmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_n \end{pmatrix}}_{\boldsymbol{\varepsilon}, n\times 1}.$$

Assuming $\boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\mathbb{I})$, the maximum likelihood estimator of $\boldsymbol{\beta}$ is

$$\widehat{\boldsymbol{\beta}} = \text{argmin}\{\|\boldsymbol{y} - \boldsymbol{X}^\mathsf{T}\boldsymbol{\beta}\|_{\ell_2}\} = (\boldsymbol{X}^\mathsf{T}\boldsymbol{X})^{-1}\boldsymbol{X}^\mathsf{T}\boldsymbol{y}$$

... under the assumtption that $\boldsymbol{X}^\mathsf{T}\boldsymbol{X}$ is a full-rank matrix.

What if $\boldsymbol{X}_i^\mathsf{T}\boldsymbol{X}$ cannot be inverted? Then $\widehat{\boldsymbol{\beta}} = [\boldsymbol{X}^\mathsf{T}\boldsymbol{X}]^{-1}\boldsymbol{X}^\mathsf{T}\boldsymbol{y}$ does not exist, but $\widehat{\boldsymbol{\beta}}_\lambda = [\boldsymbol{X}^\mathsf{T}\boldsymbol{X} + \lambda\mathbb{I}]^{-1}\boldsymbol{X}^\mathsf{T}\boldsymbol{y}$ always exist if $\lambda > 0$.

Ridge Regression

The estimator $\widehat{\boldsymbol{\beta}} = [\boldsymbol{X}^\mathsf{T}\boldsymbol{X} + \lambda\mathbb{I}]^{-1}\boldsymbol{X}^\mathsf{T}\boldsymbol{y}$ is the Ridge estimate obtained as solution of

$$\widehat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} [y_i - \beta_0 - \boldsymbol{x}_i^\mathsf{T}\boldsymbol{\beta}]^2 + \lambda \underbrace{\|\boldsymbol{\beta}\|_{\ell_2}}_{\mathbf{1}^\mathsf{T}\boldsymbol{\beta}^2} \right\}$$

for some tuning parameter λ. One can also write

$$\widehat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta};\|\boldsymbol{\beta}\|_{\ell_2}\leq s}{\operatorname{argmin}} \{\|\boldsymbol{Y} - \boldsymbol{X}^\mathsf{T}\boldsymbol{\beta}\|_{\ell_2}\}$$

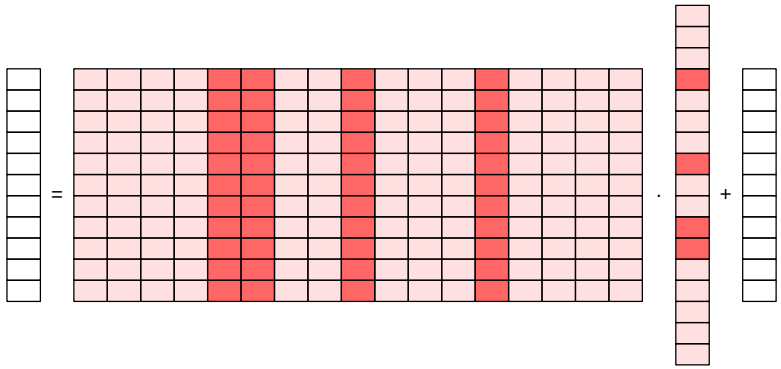
Remark Note that we solve $\widehat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \{\text{objective}(\boldsymbol{\beta})\}$ where

$$\text{objective}(\boldsymbol{\beta}) = \underbrace{\mathcal{L}(\boldsymbol{\beta})}_{\text{training loss}} + \underbrace{\mathcal{R}(\boldsymbol{\beta})}_{\text{regularization}}$$

Going further on sparcity issues

In severall applications, k can be (very) large, but a lot of features are just noise: $\beta_j = 0$ for many j's. Let s denote the number of relevent features, with $s << k$, cf Hastie, Tibshirani & Wainwright (2015),

$$s = \text{card}\{\mathcal{S}\} \text{ where } \mathcal{S} = \{j; \beta_j \neq 0\}$$



The model is now $y = \boldsymbol{X}_{\mathcal{S}}^{\mathsf{T}}\boldsymbol{\beta}_{\mathcal{S}} + \varepsilon$, where $\boldsymbol{X}_{\mathcal{S}}^{\mathsf{T}}\boldsymbol{X}_{\mathcal{S}}$ is a full rank matrix.

Going further on sparcity issues

Define $\|\boldsymbol{a}\|_{\ell_0} = \sum \mathbf{1}(|a_i| > 0)$. Ici $\dim(\boldsymbol{\beta}) = s$.

We wish we could solve

$$\widehat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta};\|\boldsymbol{\beta}\|_{\ell_0} \leq s}{\operatorname{argmin}} \{\|\boldsymbol{Y} - \boldsymbol{X}^{\mathsf{T}}\boldsymbol{\beta}\|_{\ell_2}\}$$

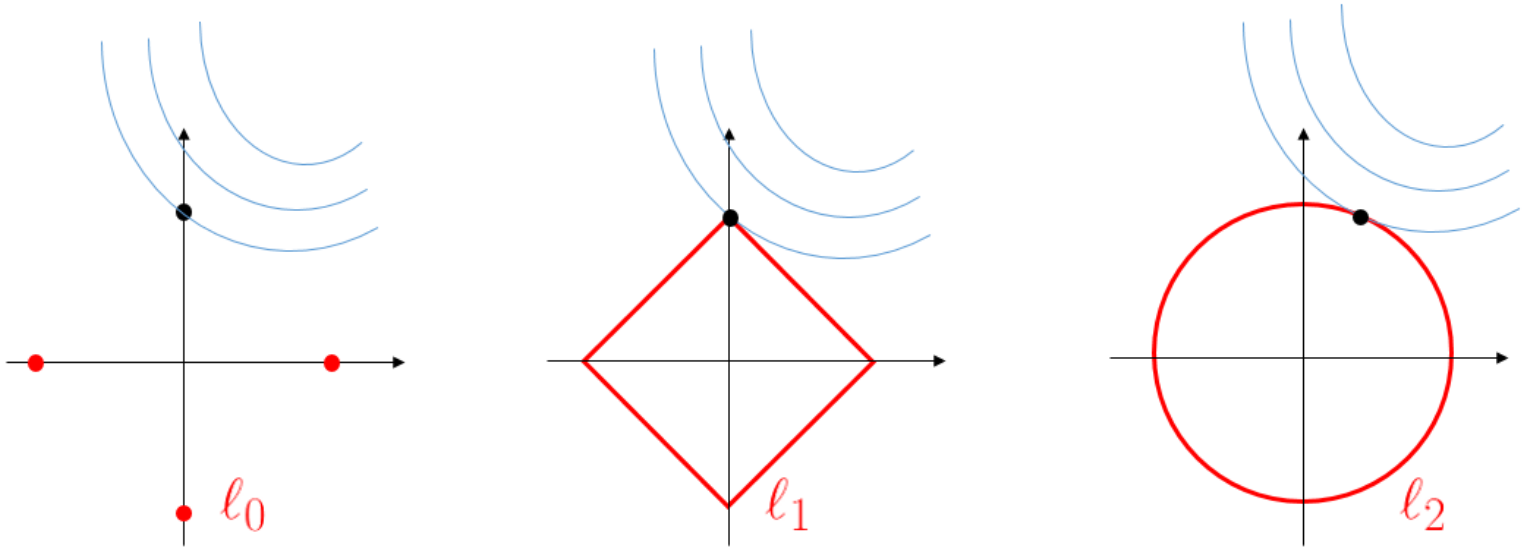
Problem: it is usually not possible to describe all possible constraints, since $\binom{s}{k}$ coefficients should be chosen here (with k (very) large).

Idea: solve the dual problem

$$\widehat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta};\|\boldsymbol{Y}-\boldsymbol{X}^{\mathsf{T}}\boldsymbol{\beta}\|_{\ell_2} \leq h}{\operatorname{argmin}} \{\|\boldsymbol{\beta}\|_{\ell_0}\}$$

where we might convexify the ℓ_0 norm, $\|\cdot\|_{\ell_0}$.

Regularization ℓ_0, ℓ_1 et ℓ_2



Going further on sparcity issues

On $[-1,+1]^k$, the convex hull of $\|\boldsymbol{\beta}\|_{\ell_0}$ is $\|\boldsymbol{\beta}\|_{\ell_1}$

On $[-a,+a]^k$, the convex hull of $\|\boldsymbol{\beta}\|_{\ell_0}$ is $a^{-1}\|\boldsymbol{\beta}\|_{\ell_1}$

Hence,

$$\widehat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta};\|\boldsymbol{\beta}\|_{\ell_1}\leq\tilde{s}}{\operatorname{argmin}} \{\|\boldsymbol{Y}-\boldsymbol{X}^{\mathsf{T}}\boldsymbol{\beta}\|_{\ell_2}\}$$

is equivalent (Kuhn-Tucker theorem) to the Lagragian optimization problem

$$\widehat{\boldsymbol{\beta}} = \operatorname{argmin}\{\|\boldsymbol{Y}-\boldsymbol{X}^{\mathsf{T}}\boldsymbol{\beta}\|_{\ell_2}+\lambda\|\boldsymbol{\beta}\|_{\ell_1}\}$$

LASSO *Least Absolute Shrinkage and Selection Operator*

$$\widehat{\boldsymbol{\beta}} \in \text{argmin}\{\|\boldsymbol{Y} - \boldsymbol{X}^\mathsf{T}\boldsymbol{\beta}\|_{\ell_2} + \lambda\|\boldsymbol{\beta}\|_{\ell_1}\}$$

is a convex problem (several algorithms*), but not strictly convex (no unicity of the minimum). Nevertheless, predictions $\widehat{\boldsymbol{y}} = \boldsymbol{x}^\mathsf{T}\widehat{\boldsymbol{\beta}}$ are unique

* MM, minimize majorization, coordinate descent Hunter (2003).

Optimal LASSO Penalty

Use cross validation, e.g. K-fold,

$$\widehat{\boldsymbol{\beta}}_{(-k)}(\lambda) = \text{argmin}\left\{\sum_{i\notin\mathcal{I}_k}[y_i - \boldsymbol{x}_i^\mathsf{T}\boldsymbol{\beta}]^2 + \lambda\|\boldsymbol{\beta}\|\right\}$$

then compute the sum of the squared errors,

$$Q_k(\lambda) = \sum_{i\in\mathcal{I}_k}[y_i - \boldsymbol{x}_i^\mathsf{T}\widehat{\boldsymbol{\beta}}_{(-k)}(\lambda)]^2$$

and finally solve

$$\lambda^\star = \text{argmin}\left\{\overline{Q}(\lambda) = \frac{1}{K}\sum_k Q_k(\lambda)\right\}$$

Note that this might overfit, so Hastie, Tibshiriani & Friedman (2009) suggest the largest λ such that

$$\overline{Q}(\lambda) \leq \overline{Q}(\lambda^\star) + \text{se}[\lambda^\star] \text{ with } \text{se}[\lambda]^2 = \frac{1}{K^2}\sum_{k=1}^{K}[Q_k(\lambda) - \overline{Q}(\lambda)]^2$$

Boosting

Boosting is a machine learning ensemble meta-algorithm for reducing bias primarily and also variance in supervised learning, and a family of machine learning algorithms which convert weak learners to strong ones. (Source: wikipedia)

The heuristics is simple: we consider an iterative process where we keep modeling the errors.

Fit model for $\boldsymbol{y}$, $m_1(\cdot)$ from $\boldsymbol{y}$ and $\boldsymbol{X}$, and compute the error, $\boldsymbol{\varepsilon}_1 = \boldsymbol{y} - m_1(\boldsymbol{X})$.

Fit model for $\boldsymbol{\varepsilon}_1$, $m_2(\cdot)$ from $\boldsymbol{\varepsilon}_1$ and $\boldsymbol{X}$, and compute the error, $\boldsymbol{\varepsilon}_2 = \boldsymbol{\varepsilon}_1 - m_2(\boldsymbol{X})$, etc. Then set

$$m(\cdot) = \underbrace{m_1(\cdot)}_{\sim y} + \underbrace{m_2(\cdot)}_{\sim \varepsilon_1} + \underbrace{m_3(\cdot)}_{\sim \varepsilon_2} + \cdots + \underbrace{m_k(\cdot)}_{\sim \varepsilon_{k-1}}$$

Boosting

With (very) general notations, we want to solve

$$m^\star = \text{argmin}\{\mathbb{E}[\ell(Y, m(\boldsymbol{X}))]\}$$

for some loss function ℓ.

It is an iterative procedure: assume that at some step k we have an estimator $m_k(\boldsymbol{X})$. Why not constructing a new model that might improve our model,

$$m_{k+1}(\boldsymbol{X}) = m_k(\boldsymbol{X}) + h(\boldsymbol{X}).$$

What $h(\cdot)$ could be?

Boosting

In a perfect world, $h(\boldsymbol{X}) = \boldsymbol{y} - m_k(\boldsymbol{X})$, which can be interpreted as a residual.

Note that this residual is the gradient of $\frac{1}{2}[y - m(\boldsymbol{x})]^2$

A gradient descent is based on Taylor expansion

$$\underbrace{f(\boldsymbol{x}_k)}_{\langle f, \boldsymbol{x}_k \rangle} \sim \underbrace{f(\boldsymbol{x}_{k-1})}_{\langle f, \boldsymbol{x}_{k-1} \rangle} + \underbrace{(\boldsymbol{x}_k - \boldsymbol{x}_{k-1})}_{\alpha} \underbrace{\nabla f(\boldsymbol{x}_{k-1})}_{\langle \nabla f, \boldsymbol{x}_{k-1} \rangle}$$

But here, it is different. We claim we can write

$$\underbrace{f_k(\boldsymbol{x})}_{\langle f_k, \boldsymbol{x} \rangle} \sim \underbrace{f_{k-1}(\boldsymbol{x})}_{\langle f_{k-1}, \boldsymbol{x} \rangle} + \underbrace{(f_k - f_{k-1})}_{\beta} \underbrace{?}_{\langle f_{k-1}, \nabla \boldsymbol{x} \rangle}$$

where ? is interpreted as a 'gradient'.

Boosting

Here, f_k is a $\mathbb{R}^d \to \mathbb{R}$ function, so the gradient should be in such a (big) functional space $\to$ want to approximate that function.

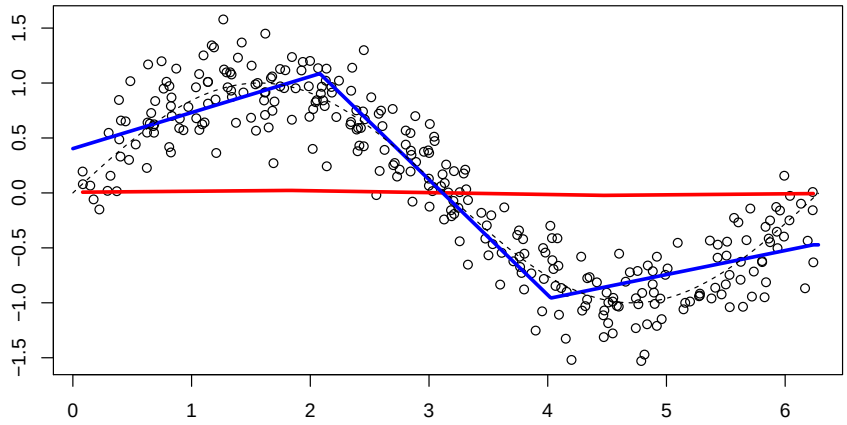
$$m_k(\boldsymbol{x}) = m_{k-1}(\boldsymbol{x}) + \underset{f \in \mathcal{F}}{\text{argmin}} \left\{ \sum_{i=1}^{n} \ell(Y_i, m_{k-1}(\boldsymbol{x}) + f(\boldsymbol{x})) \right\}$$

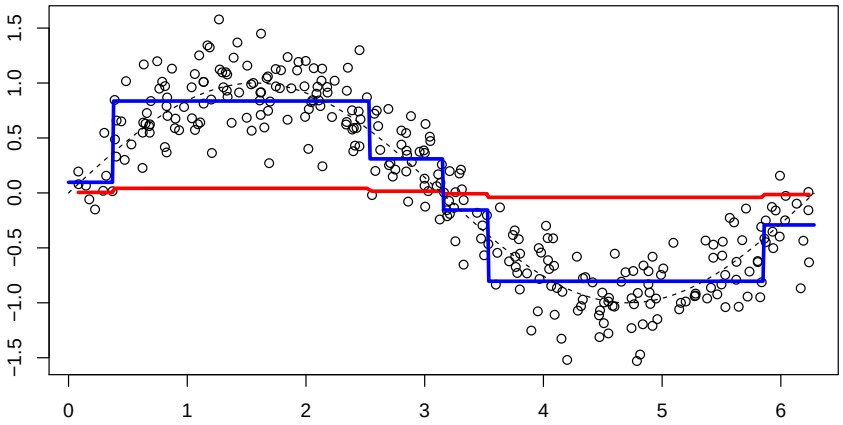
where $f \in \mathcal{F}$ means that we seek in a class of weak learner functions.

If learner are two strong, the first loop leads to some fixed point, and there is no learning procedure, see linear regression $y = \boldsymbol{x}^\mathsf{T}\boldsymbol{\beta} + \varepsilon$. Since $\varepsilon \perp \boldsymbol{x}$ we cannot learn from the residuals.

In order to make sure that we learn weakly, we can use some shrinkage parameter ν (or collection of parameters ν_j).

Boosting with Piecewise Linear Spline & Stump Functions





Take Home Message

- Similar goal: getting a predictive model, $\widehat{m}(\boldsymbol{x})$
- Different/Similar tools: minimize loss/maximize likelihood

$$\widehat{m}(\cdot) = \underset{m(\cdot)\in\mathcal{F}}{\text{argmin}} \left\{ \sum_{i=1}^{n} \ell(y_i, m(\boldsymbol{x}_i)) \right\} \text{ vs. } \widehat{m}(\cdot) = \underset{m(\cdot)\in\mathcal{F}}{\text{argmax}} \left\{ \sum_{i=1}^{n} \log f(y_i; m(\boldsymbol{x}_i)) \right\}$$

- Try to remove the noise and avoid overfit using cross-validation,

$$\ell(y_i, \widehat{m}_{(-i)}(\boldsymbol{x}_i))$$

- Use computational tricks (bootstrap) to increase robustness
- Nice tools to select interesting features (LASSO, variable importance)