

Machine Learning Statistical Learning: Introduction and Cross Validation

Ana Karina Fermin

winter 2019-2020

Outline

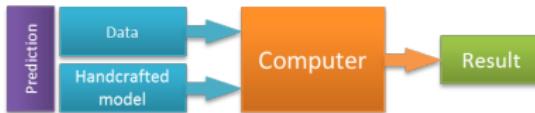
1 Introduction

2 Supervised Learning

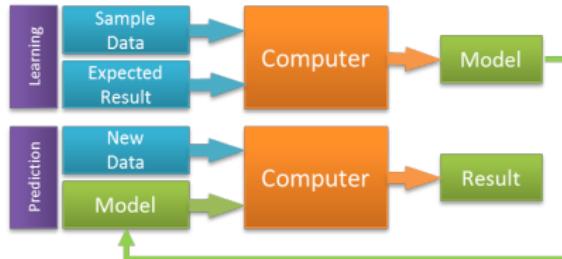
3 Error Estimation

4 References

Traditional modeling:



Machine Learning:



A definition by Tom Mitchell
(<http://www.cs.cmu.edu/~tom/>)

A computer program is said to learn from **experience E** with respect to some **class of tasks T** and **performance measure P**, if its performance at tasks in T , as measured by P, improves with experience E.



Credit Score

- **Task:** Prediction (default or no default)
- **Data:** Client profile, Client credit history...
- **Performance measure:** error rate.

News Clustering

Introduction

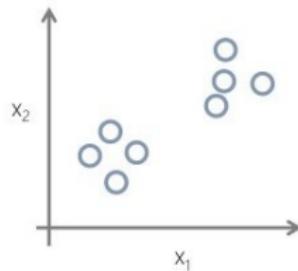
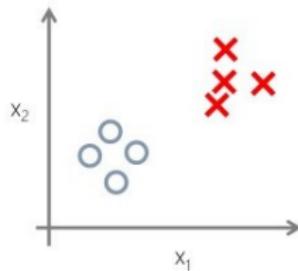
The screenshot shows the Google News interface. On the left, a sidebar lists 'SECTIONS' including Top Stories, World, U.S., Business, Technology, Entertainment, Sports, Science, and Health, with 'Top Stories' currently selected. The main area displays 'Top Stories' with two main news items. The first story is about Sarah Huckabee Sanders, featuring a photo of her and several related articles from Fox News, CNN, and the Washington Post. The second story is about Donald Trump's Time Magazine cover, also with related articles from the Washington Post. A sidebar on the right lists 'In the News' topics such as Executive I, Serena Williams, Syria, Google, John McEnroe, and Bashar al-Assad.

A news clustering algorithm:

- **Task:** group articles corresponding to the same news
- **Performance:** quality of the clusters
- **Experience:** set of articles

Supervised and Unsupervised

Introduction



Supervised Learning (Imitation)

- **Goal:** Learn a function f predicting a variable Y from an individual \underline{X} .
- **Data:** Learning set with labeled examples (\underline{X}_i, Y_i)
- **Assumption:** Future data behaves as past data!
- **Predicting is not explaining!**

Unsupervised Learning (Structure Discovery)

- **Goal:** Discover a structure within a set of individuals (\underline{X}_i) .
- **Data:** Learning set with unlabeled examples (\underline{X}_i)
- Unsupervised learning is not a well-posed setting....

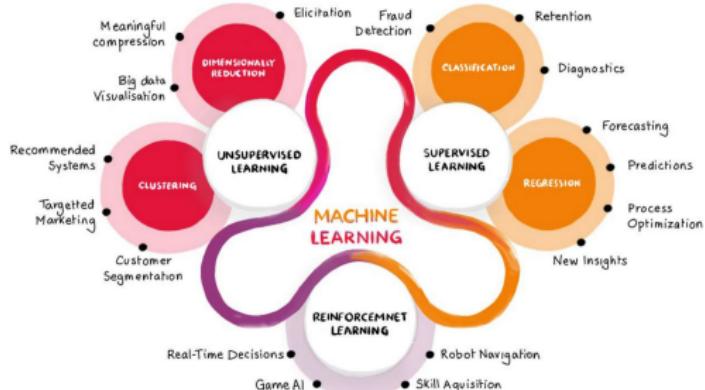


A robot endowed with a set of sensors playing football:

- **Task:** play football
- **Performance:** score evolution
- **Experience:**
 - current environment and outcome,
 - past games

Three Kinds of Learning

Introduction



Unsupervised Learning

- Task:** Clustering/DR
- Performance:** Quality
- Experience:** Raw dataset (No Ground Truth)

Supervised Learning

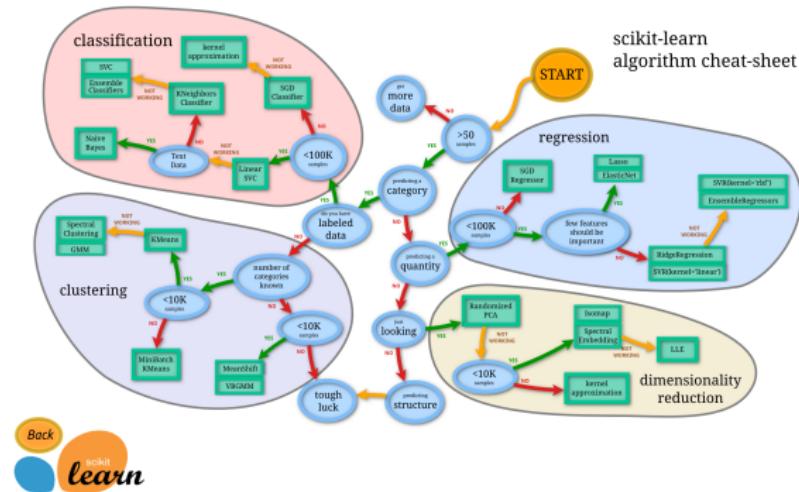
- Task:** Prediction
- Performance:** Average error
- Experience:** Predictions (Ground Truth)

Reinforcement Learning

- Task:** Action
- Performance:** Total reward
- Experience:** Reward from env. (Interact. with env.)

Machine Learning (Unsupervised/Supervised)

Introduction



ML Methods

- Huge catalog of methods,
- Need to define the performance, feature design.
- **Here, we will only see supervised learning**



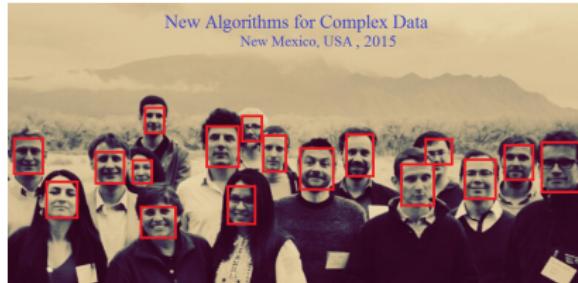
Credit Score

- **Task:** Prediction (default or no default)
- **Data:** Client profile, Client credit history...
- **Performance measure:** error rate.



Spam detection

- **Task:** Prediction (spam or no spam)
- **Data:** email collection
- **Performance measure:** error rate.



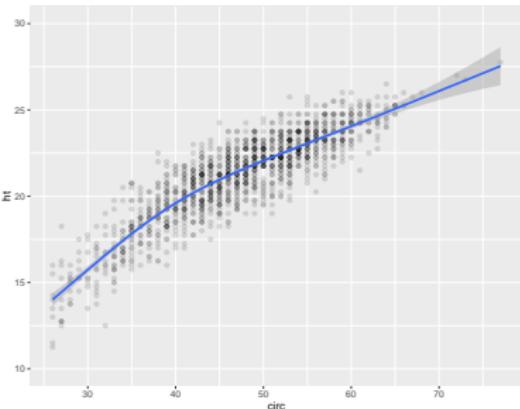
Face detection

- **Task:** Detect the position of faces in an image
- Different setting?
- Reformulation as a supervised learning problem.
- **Goal:** Detect the presence of faces at several positions and scales.
- **Data:** $X = \text{sub image} / Y = \text{presence or no of a face...}$
- **Performance measure:** error rate.
- Lots of detections in an image: post processing required...
- **Performance measure:** box precision.

0	0	0	0	0
1	1	1	1	1
2	2	2	2	2
3	3	3	3	3
4	4	4	4	4
5	5	5	5	5
6	6	6	6	6
7	7	7	7	7
8	8	8	8	8
9	9	9	9	9

Reading a ZIP code on an envelop

- **Task:** give a number from an image.
- **Data:** $X = \text{image} / Y = \text{corresponding number}$.
- **Performance measure:** error rate.

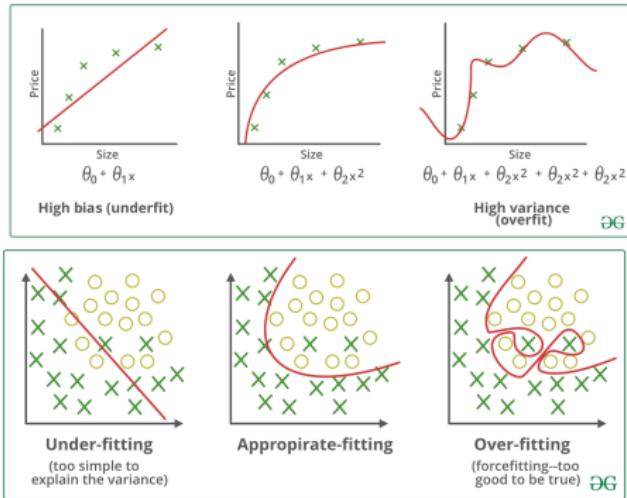


Height estimation

- Simple (and classical) dataset.
- **Task:** predict the height from circumference.
- **Data:** $X = \text{circumference}$ / $Y = \text{height}$.
- **Performance measure:** means squared error.

Under and Over Fitting

Introduction



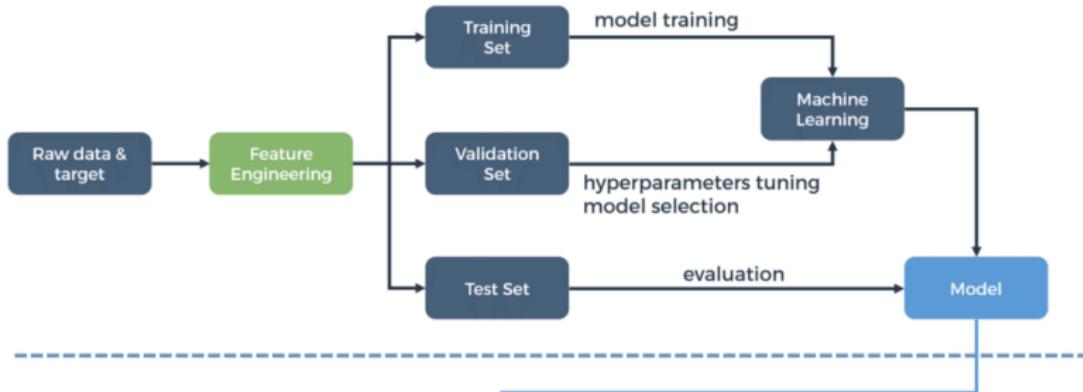
Finding the Right Complexity

- What is best?
 - A simple model that is stable but false? (*oversimplification*)
 - A very complex model that could be correct but is unstable? (*conspiracy theory*)
- Neither of them: tradeoff that depends on the dataset.

ML Pipeline

Introduction

TRAINING

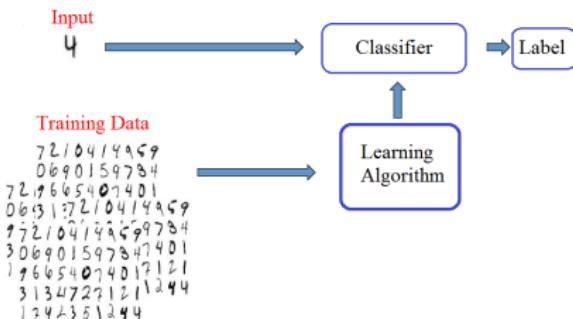


PREDICTING



Learning pipeline

- Test and compare models.
- Deployment pipeline is different!



Goal

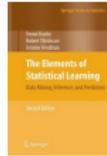
- Know the inner mechanism of the most classical supervised **ML methods** (Logistic, SVM, Neural Nets and Trees) in order to understand their strengths and limitations.
- Understand some **optimization** tools used in ML

Evaluation

- A project (homework) with R or Python

IFMA/IMP Bloc 3 ML - 5 Lectures (09h00-12h15)

- Tue. 21/01: Statistical Learning: Introduction and Cross Validation
 - Tue. 28/01: ML Methods: Probabilistic Point of View
 - Tue. 04/02: ML Methods: Optimization Point of View
 - Tue. 11/02: ML Methods: Trees and Ensemble Methods
 - Tue. 25/02: ML Methods: Neural Networks and Deep Learning
-
- 20/03: **Homework (project with 2-3 students)**



T. Hastie, R. Tibshirani, and J. Friedman.
The Elements of Statistical Learning.
Springer Series in Statistics, 2009



M. Mohri, A. Rostamizadeh, and A. Talwalkar.
Foundations of Machine Learning.
MIT Press, 2012



A. Géron.
Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow (2nd ed.)
O'Reilly, 2019

Outline

1 Introduction

2 Supervised Learning

3 Error Estimation

4 References

Supervised Learning Framework

- Input measurement $\underline{X} \in \mathcal{X}$
- Output measurement $Y \in \mathcal{Y}$.
- $(\underline{X}, Y) \sim \mathbb{P}$ with \mathbb{P} unknown.
- **Training data** : $\mathcal{D}_n = \{(\underline{X}_1, Y_1), \dots, (\underline{X}_n, Y_n)\}$ (i.i.d. $\sim \mathbb{P}$)
- Often
 - $\underline{X} \in \mathbb{R}^d$ and $Y \in \{-1, 1\}$ (classification)
 - or $\underline{X} \in \mathbb{R}^d$ and $Y \in \mathbb{R}$ (regression).
- A **predictor** is a function in $\mathcal{F} = \{f : \mathcal{X} \rightarrow \mathcal{Y} \text{ meas.}\}$

Goal

- Construct a **good** predictor \hat{f} from the training data.
- Need to specify the meaning of good.
- Classification and regression are almost the **same** problem!

Loss function for a generic predictor

- **Loss function:** $\ell(Y, f(\underline{X}))$ measures the goodness of the prediction of Y by $f(\underline{X})$
- Examples:
 - Prediction loss: $\ell(Y, f(\underline{X})) = \mathbf{1}_{Y \neq f(\underline{X})}$
 - Quadratic loss: $\ell(Y, f(\underline{X})) = |Y - f(\underline{X})|^2$

Risk function

- Risk measured as the average loss for a new couple:

$$\mathcal{R}(f) = \mathbb{E}_{(\underline{X}, Y) \sim \mathbb{P}} [\ell(Y, f(\underline{X}))]$$

- Examples:
 - Prediction loss: $\mathbb{E} [\ell(Y, f(\underline{X}))] = \mathbb{P}(Y \neq f(\underline{X}))$
 - Quadratic loss: $\mathbb{E} [\ell(Y, f(\underline{X}))] = \mathbb{E} [|Y - f(\underline{X})|^2]$

- **Beware:** As \hat{f} depends on \mathcal{D}_n , $\mathcal{R}(\hat{f})$ is a random variable!

- The best solution f^* (which is independent of \mathcal{D}_n) is

$$f^* = \arg \min_{f \in \mathcal{F}} R(f) = \arg \min_{f \in \mathcal{F}} \mathbb{E} [\ell(Y, f(\underline{X}))] = \arg \min_{f \in \mathcal{F}} \mathbb{E}_{\underline{X}} \left[\mathbb{E}_{Y|\underline{X}} [\ell(Y, f(\underline{X}))] \right]$$

Bayes Predictor (explicit solution)

- In binary classification with 0 – 1 loss:

$$f^*(\underline{X}) = \begin{cases} +1 & \text{if } \mathbb{P}(Y = +1|\underline{X}) \geq \mathbb{P}(Y = -1|\underline{X}) \\ & \Leftrightarrow \mathbb{P}(Y = +1|\underline{X}) \geq 1/2 \\ -1 & \text{otherwise} \end{cases}$$

- In regression with the quadratic loss

$$f^*(\underline{X}) = \mathbb{E}[Y|\underline{X}]$$

Issue: Solution requires to **know** $\mathbb{E}[Y|\underline{X}]$ for all values of \underline{X} !

Machine Learning

- Learn a rule to construct a **predictor** $\hat{f} \in \mathcal{F}$ from the training data \mathcal{D}_n s.t. **the risk** $\mathcal{R}(\hat{f})$ is **small on average** or with high probability with respect to \mathcal{D}_n .
- In practice, the rule should be an algorithm!

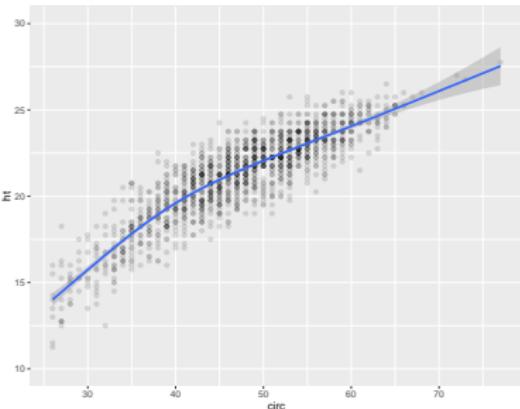
Canonical example: Empirical Risk Minimizer

- One restricts f to a subset of functions $\mathcal{S} = \{f_\theta, \theta \in \Theta\}$
- One replaces the minimization of the average loss by the minimization of the empirical loss

$$\hat{f} = \hat{f}_\theta = \operatorname{argmin}_{f_\theta, \theta \in \Theta} \frac{1}{n} \sum_{i=1}^n \ell(Y_i, f_\theta(\underline{X}_i))$$

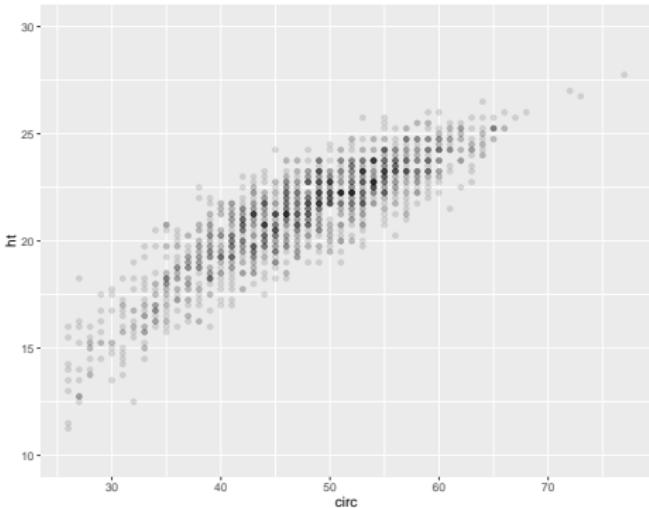
- Examples:
 - Linear regression
 - Linear discrimination with

$$\mathcal{S} = \{\underline{x} \mapsto \operatorname{sign}\{\underline{x}^\top \beta + \beta^{(0)}\} / \beta \in \mathbb{R}^d, \beta^{(0)} \in \mathbb{R}\}$$



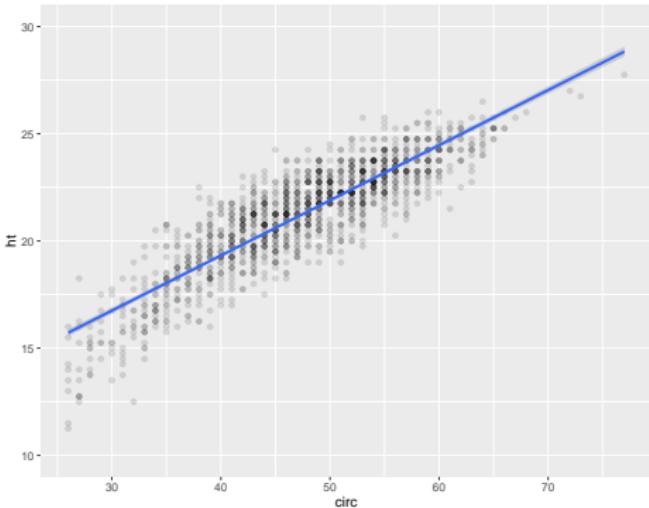
Height estimation

- Simple (and classical) dataset.
- **Task:** predict the height from circumference.
- **Data:** $X = \text{circumference}$ / $Y = \text{height}$.
- **Performance measure:** means squared error.



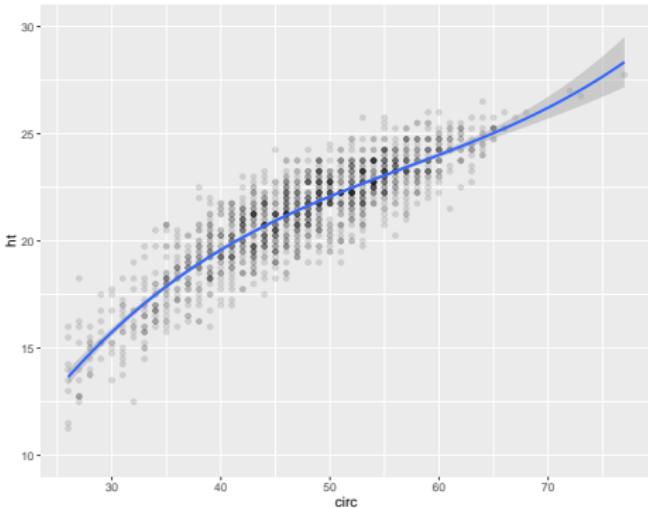
Dataset - P.A. Cornillon

- Real dataset of 1429 eucalyptus obtained by P.A. Cornillon:
 - X: circumference / Y: height
- Can we predict the height from the circumference?



Dataset - P.A. Cornillon

- Real dataset of 1429 eucalyptus obtained by P.A. Cornillon:
 - X: circumference / Y: height
- Can we predict the height from the circumference?
 - by a **line**?

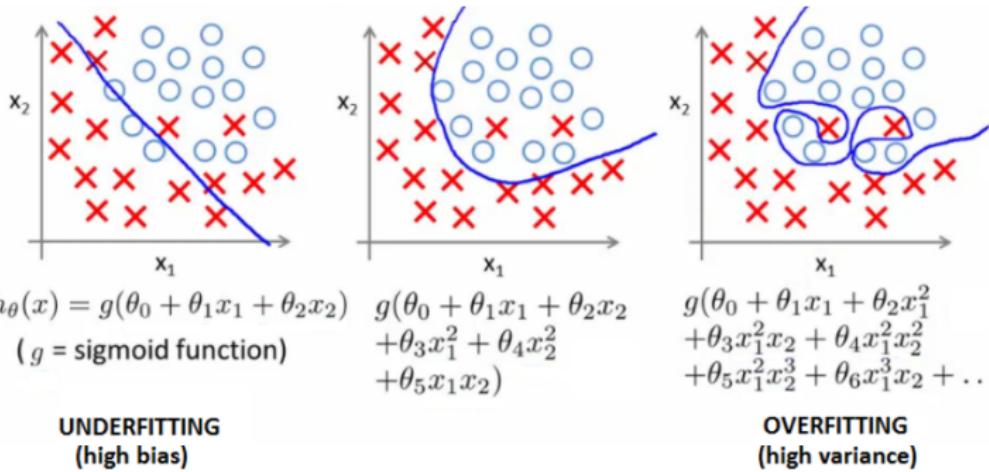


Dataset - P.A. Cornillon

- Real dataset of 1429 eucalyptus obtained by P.A. Cornillon:
 - X: circumference / Y: height
- Can we predict the height from the circumference?
 - by a **line**? by a more complex formula?

Under-fitting / Over-fitting Issue

Supervised Learning

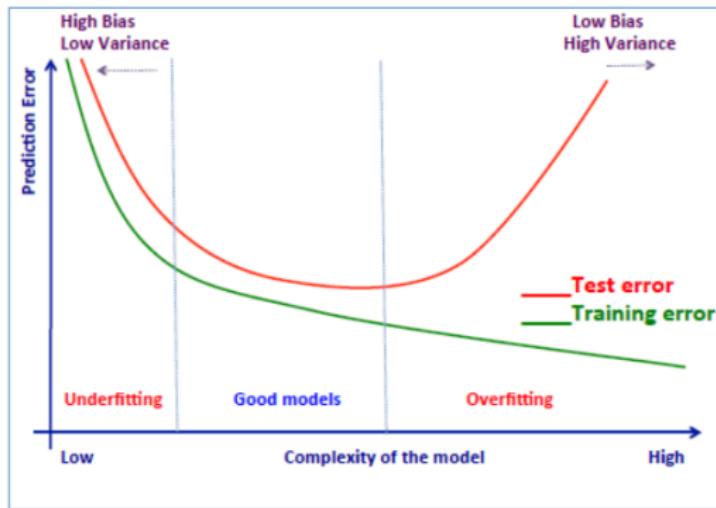


Model Complexity Dilemma

- What is best a simple or a complex model?
- Too simple to be good? Too complex to be learned?

Under-fitting / Over-fitting Issue

Supervised Learning



Under-fitting / Over-fitting

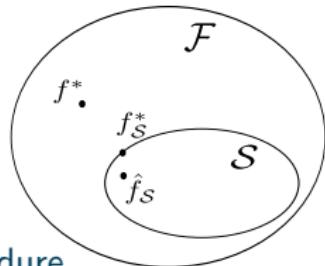
- **Under-fitting:** simple model are too simple.
- **Over-fitting:** complex model are too specific to the training set.

Bias-Variance Dilemma

Supervised Learning

- General setting:

- $\mathcal{F} = \{\text{measurable functions } \mathcal{X} \rightarrow \mathcal{Y}\}$
- Best solution: $f^* = \operatorname{argmin}_{f \in \mathcal{F}} \mathcal{R}(f)$
- Class $\mathcal{S} \subset \mathcal{F}$ of functions
- Ideal target in \mathcal{S} : $f_S^* = \operatorname{argmin}_{f \in \mathcal{S}} \mathcal{R}(f)$
- Estimate in \mathcal{S} : \hat{f}_S obtained with some procedure



Approximation error and estimation error (Bias/Variance)

$$\mathcal{R}(\hat{f}_S) - \mathcal{R}(f^*) = \underbrace{\mathcal{R}(f_S^*) - \mathcal{R}(f^*)}_{\text{Approximation error}} + \underbrace{\mathcal{R}(\hat{f}_S) - \mathcal{R}(f_S^*)}_{\text{Estimation error}}$$

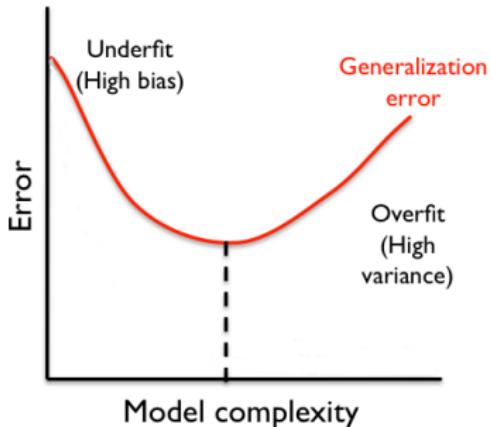
- Approx. error can be large if the model \mathcal{S} is not suitable.
- Estimation error can be large if the model is complex.

Agnostic approach

- No assumption (so far) on the law of (\underline{X}, Y) .

Under-fitting / Over-fitting Issue

Supervised Learning



- Different behavior for different model complexity
- **Low complexity model** are easily learned but the approximation error (**bias**) may be large (**Under-fit**).
- **High complexity model** may contain a good ideal target but the estimation error (**variance**) can be large (**Over-fit**)

Bias-variance trade-off \iff avoid **overfitting** and **underfitting**

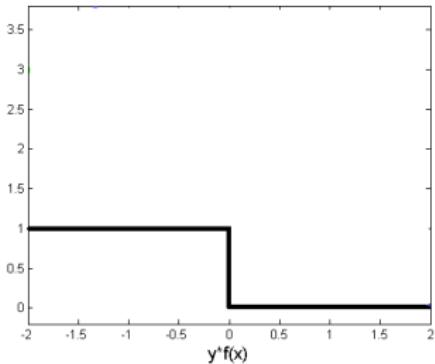
- **Rk:** Better to think in term of method (including feature engineering and specific algorithm) rather than only of model.

Statistical Learning Analysis

- Error decomposition:

$$\mathcal{R}(\hat{f}_S) - \mathcal{R}(f^*) = \underbrace{\mathcal{R}(f_S^*) - \mathcal{R}(f^*)}_{\text{Approximation error}} + \underbrace{\mathcal{R}(\hat{f}_S) - \mathcal{R}(f_S^*)}_{\text{Estimation error}}$$

- Bound on the approximation term: approximation theory.
- Probabilistic bound on the estimation term: probability theory!
- **Goal: Agnostic bounds**, i.e. bounds that do not require assumptions on \mathbb{P} ! (Statistical Learning?)
- Often need mild assumptions on \mathbb{P} ... (Nonparametric Statistics?)

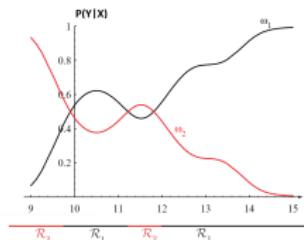


Empirical Risk Minimizer

$$\hat{f} = \operatorname{argmin}_{f \in \mathcal{S}} \frac{1}{n} \sum_{i=1}^n \ell^{0/1}(Y_i, f(\underline{x}_i))$$

- Classification loss: $\ell^{0/1}(y, f(\underline{x})) = \mathbf{1}_{y \neq f(\underline{x})}$
- Not convex and not smooth!

Ideal Solution and Estimation



- The best solution f^* (which is independent of \mathcal{D}_n) is

$$f^* = \arg \min_{f \in \mathcal{F}} R(f) = \arg \min_{f \in \mathcal{F}} \mathbb{E} [\ell(Y, f(\underline{X}))] = \arg \min_{f \in \mathcal{F}} \mathbb{E}_{\underline{X}} \left[\mathbb{E}_{Y|\underline{X}} [\ell(Y, f(\underline{x}))] \right]$$

Bayes Predictor (explicit solution)

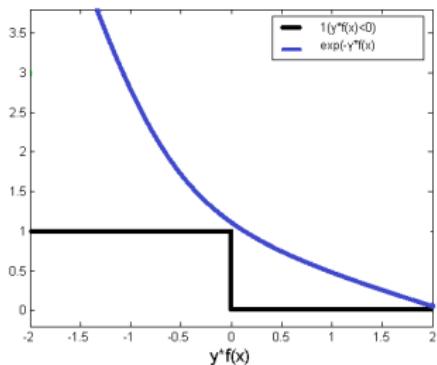
In binary classification with 0 – 1 loss:

$$f^*(\underline{X}) = \begin{cases} +1 & \text{if } \mathbb{P}(Y = +1|\underline{X}) \geq \mathbb{P}(Y = -1|\underline{X}) \\ -1 & \text{otherwise} \end{cases}$$

- Issue:** Solution requires to know $\mathbb{E}[Y|\underline{X}]$ for all values of \underline{X} !
- Solution:** Replace it by an estimate.

Optimization Point of View Loss Convexification

Supervised Learning



Minimizer of the risk

$$\hat{f} = \operatorname{argmin}_{f \in \mathcal{S}} \frac{1}{n} \sum_{i=1}^n \ell^{0/1}(Y_i, f(\underline{X}_i))$$

- **Issue:** Classification loss is not convex or smooth.
- **Solution:** Replace it by a convex majorant.

Probabilistic and Optimization Framework

Supervised Learning

How to find a good function f with a *small* risk

$$R(f) = \mathbb{E} [\ell(Y, f(\underline{X}))] \quad ?$$

Canonical approach: $\hat{f}_S = \operatorname{argmin}_{f \in S} \frac{1}{n} \sum_{i=1}^n \ell(Y_i, f(\underline{X}_i))$

Problems

- How to choose \mathcal{S} ?
- How to compute the minimization?

A Probabilistic Point of View

Solution: For \underline{X} , estimate $Y|\underline{X}$ plug this estimate in the Bayes classifier: **(Generalized) Linear Models, Kernel methods, k-nn, Naive Bayes, Tree, Bagging...**

An Optimization Point of View

Solution: If necessary replace the loss ℓ by an upper bound ℓ' and minimize the empirical loss: **SVR, SVM, Neural Network, Tree, Boosting...**

Outline

1 Introduction

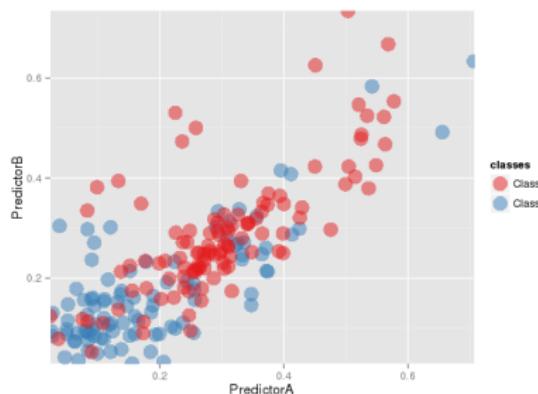
2 Supervised Learning

3 Error Estimation

4 References

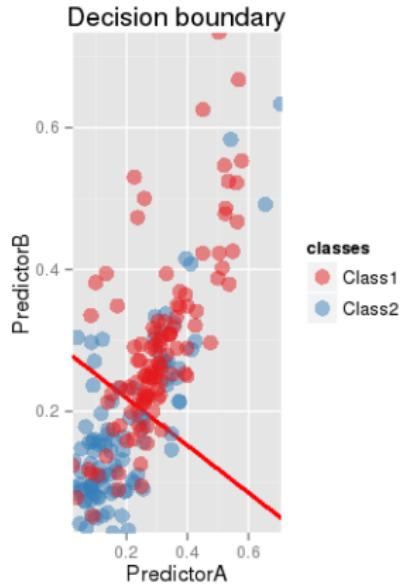
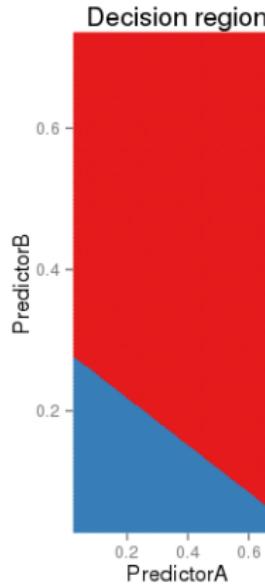
Synthetic Dataset

- Two features/covariates.
- Two classes.
- Dataset from *Applied Predictive Modeling*, M. Kuhn and K. Johnson, Springer
- Numerical experiments with R and the caret package.



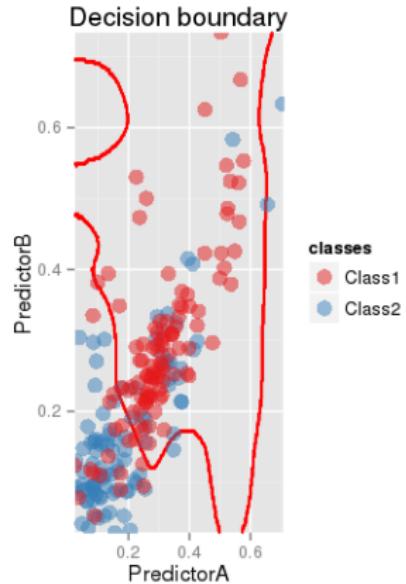
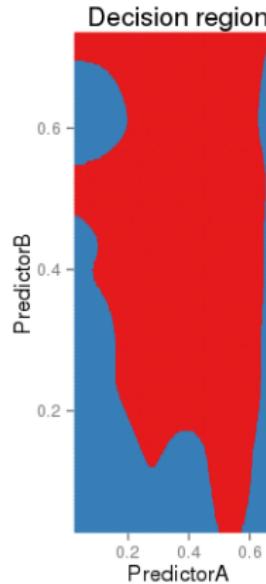
Example: Linear Discrimination

Error Estimation



Example: More Complex Model

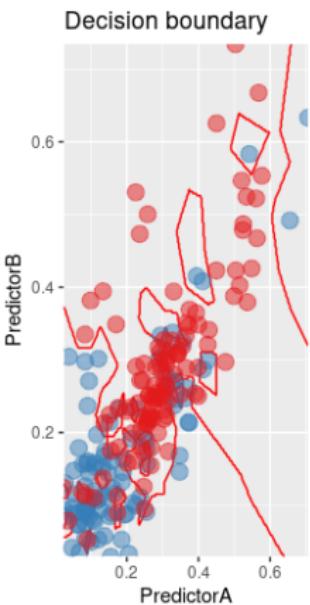
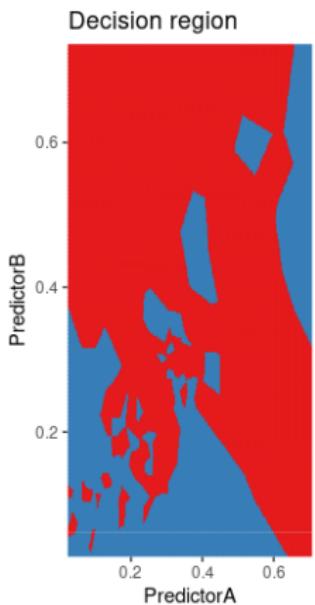
Error Estimation



Example: KNN

Error Estimation

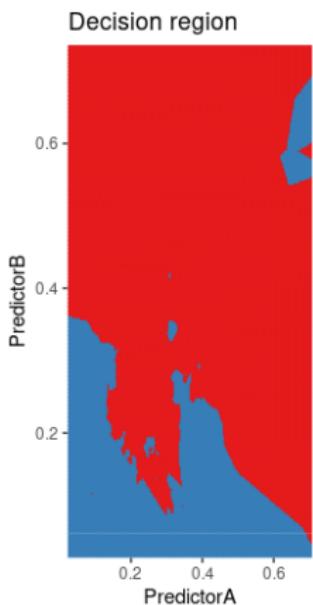
k-NN with k=1



Example: KNN

Error Estimation

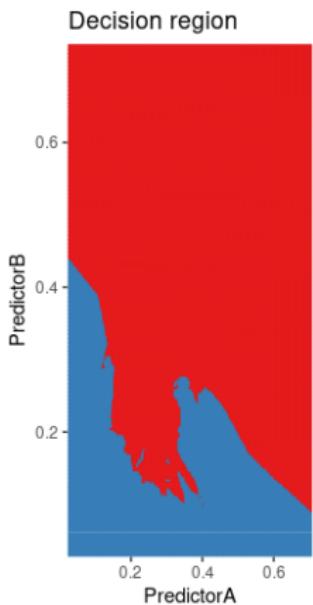
k-NN with k=5



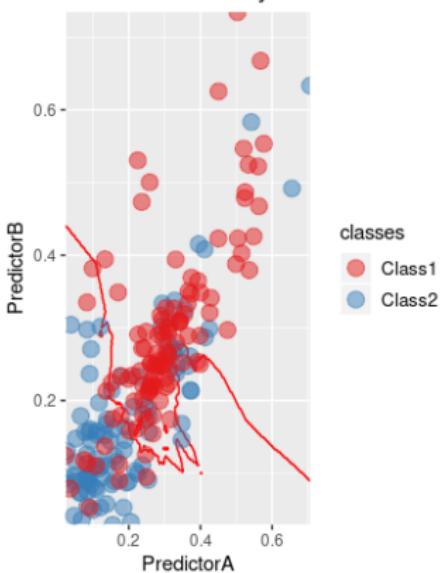
Example: KNN

Error Estimation

k-NN with k=9



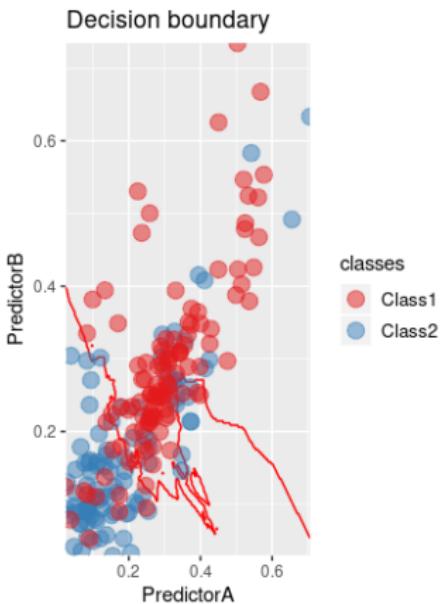
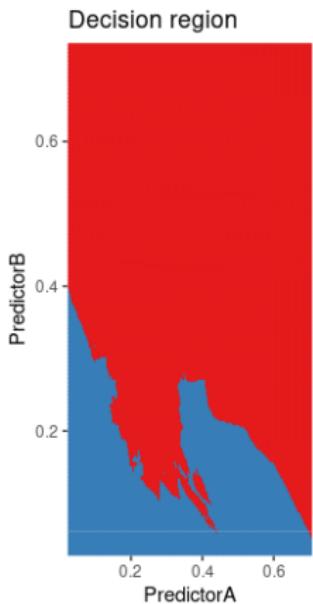
Decision boundary



Example: KNN

Error Estimation

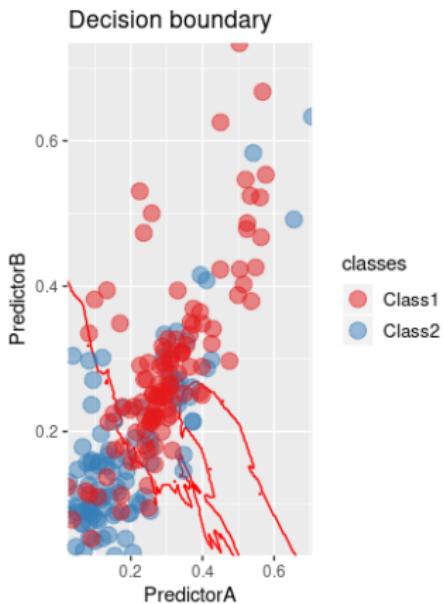
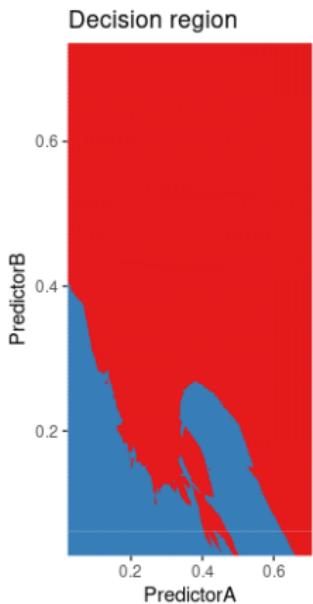
k-NN with k=13



Example: KNN

Error Estimation

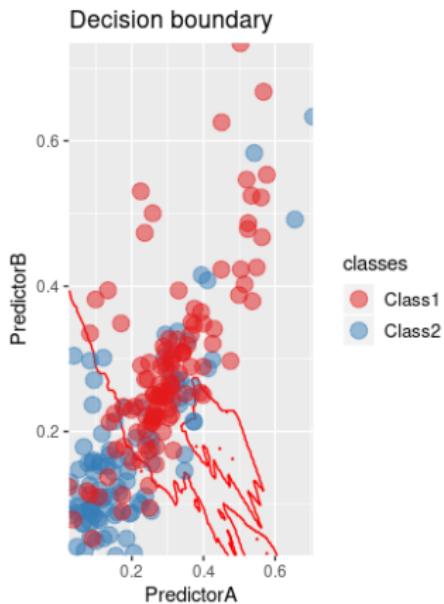
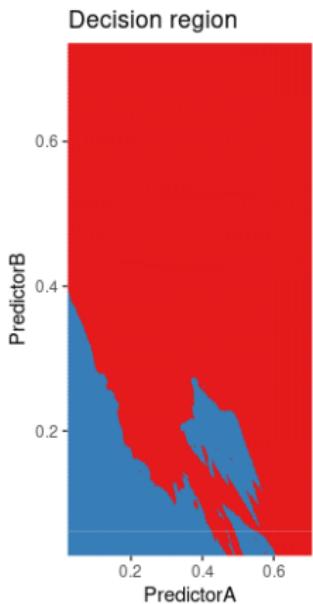
k-NN with k=17



Example: KNN

Error Estimation

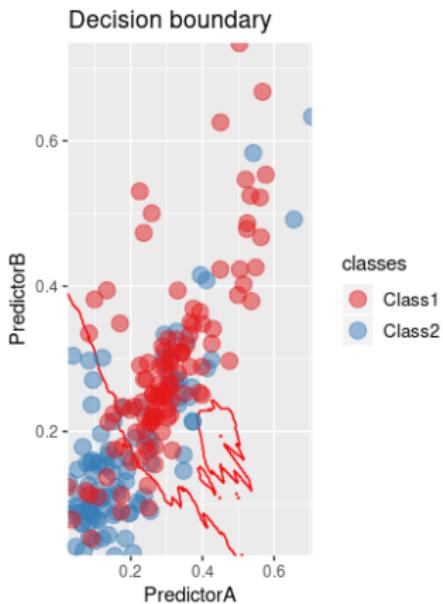
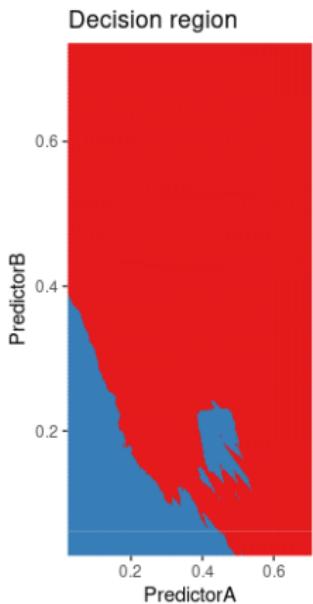
k-NN with k=21



Example: KNN

Error Estimation

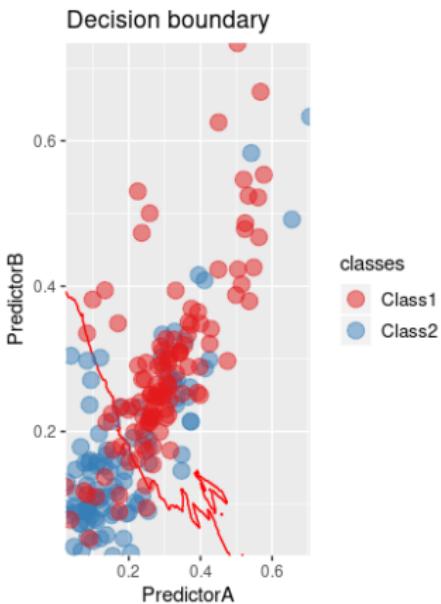
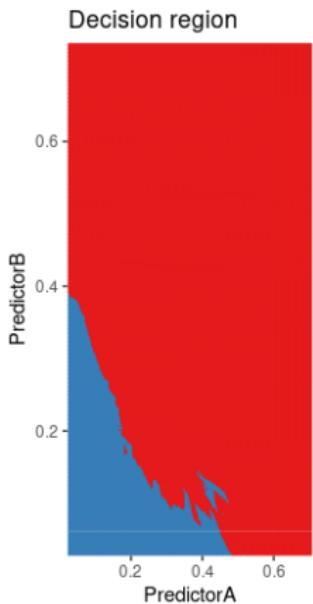
k-NN with k=25



Example: KNN

Error Estimation

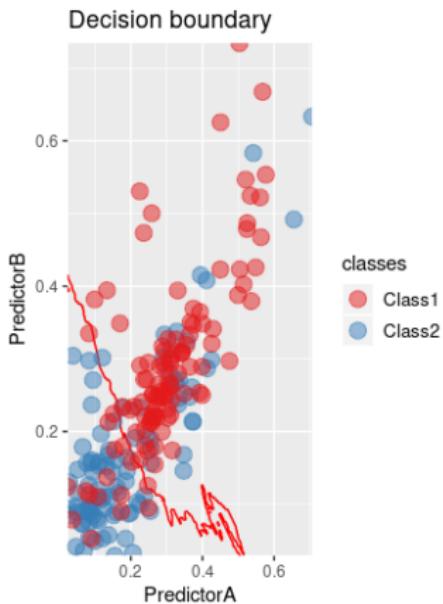
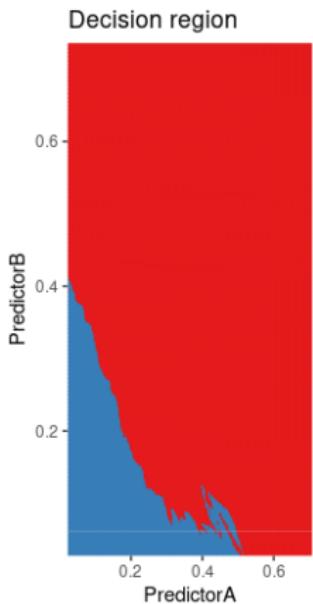
k-NN with $k=29$



Example: KNN

Error Estimation

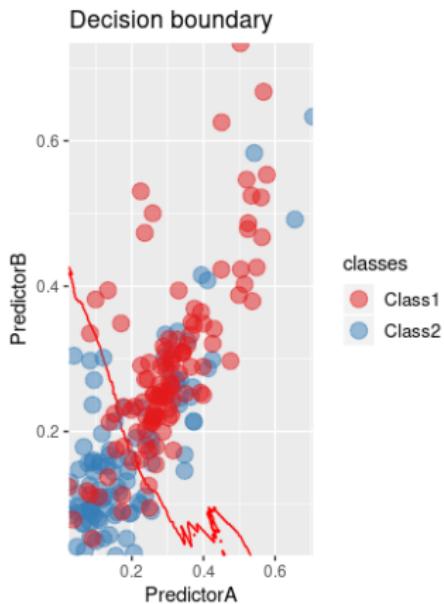
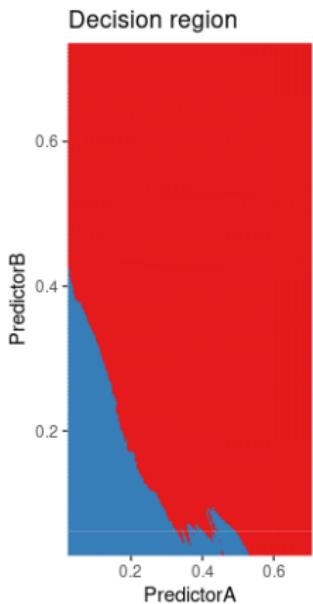
k-NN with k=33



Example: KNN

Error Estimation

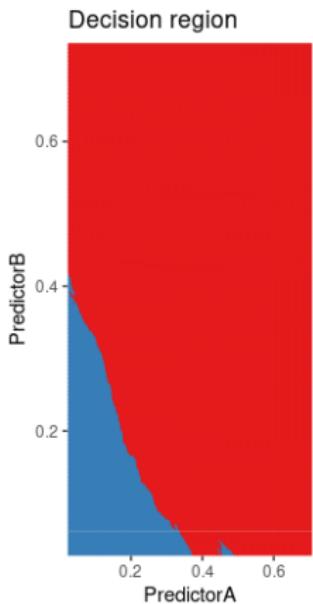
k-NN with k=37



Example: KNN

Error Estimation

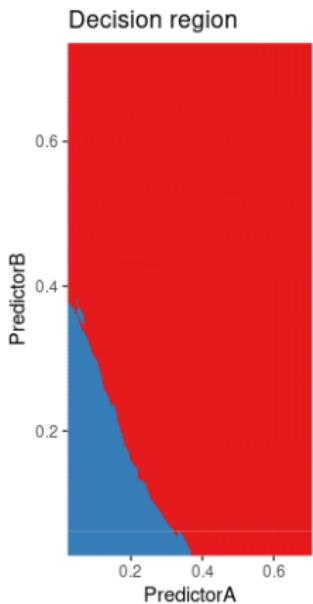
k-NN with k=45



Example: KNN

Error Estimation

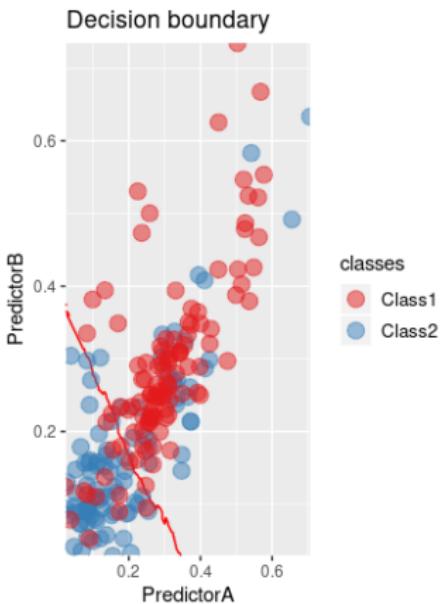
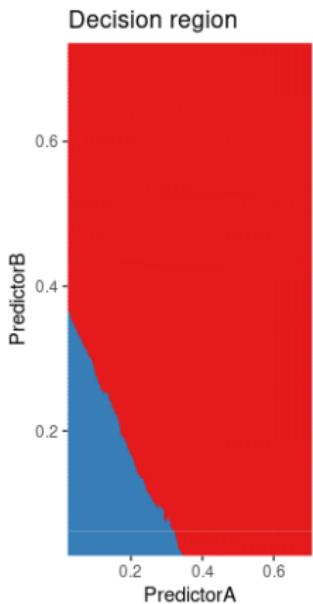
k-NN with k=53



Example: KNN

Error Estimation

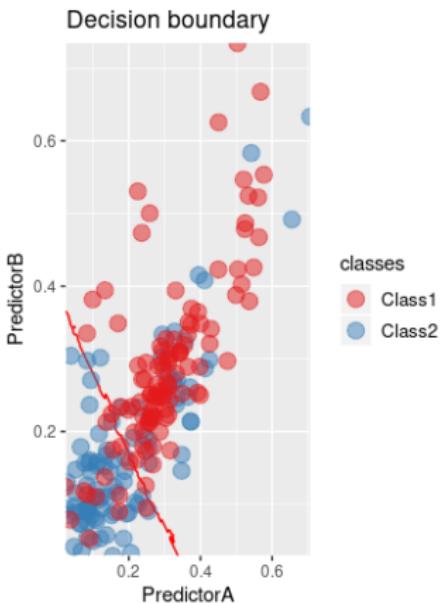
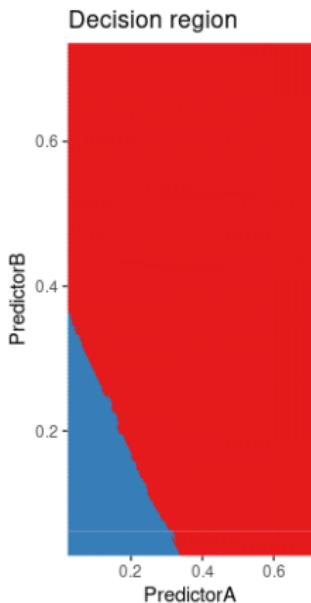
k-NN with k=61



Example: KNN

Error Estimation

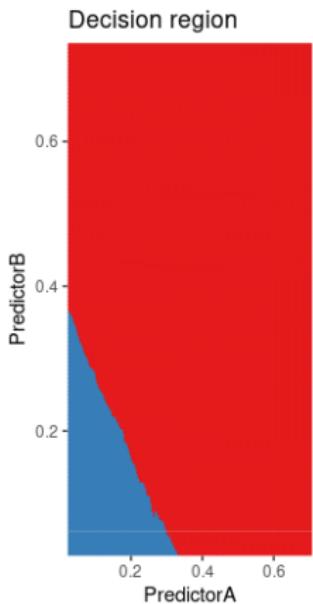
k-NN with $k=69$



Example: KNN

Error Estimation

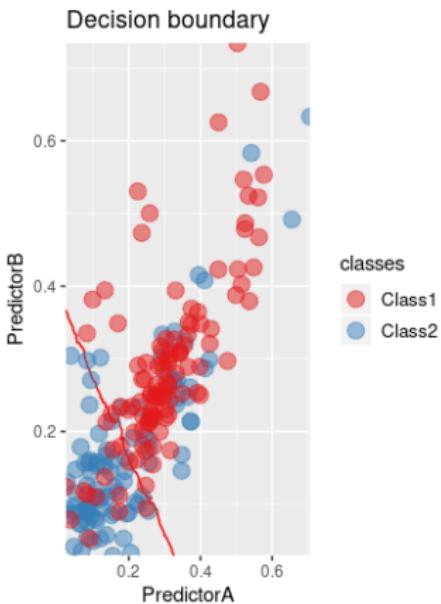
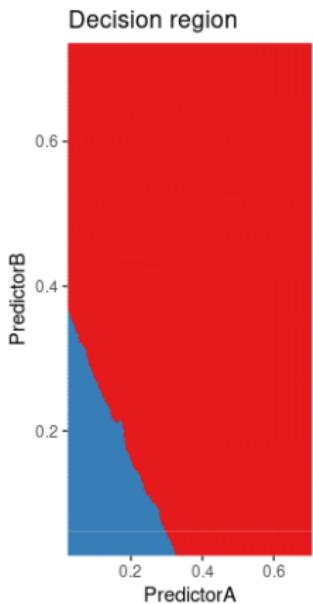
k-NN with k=77



Example: KNN

Error Estimation

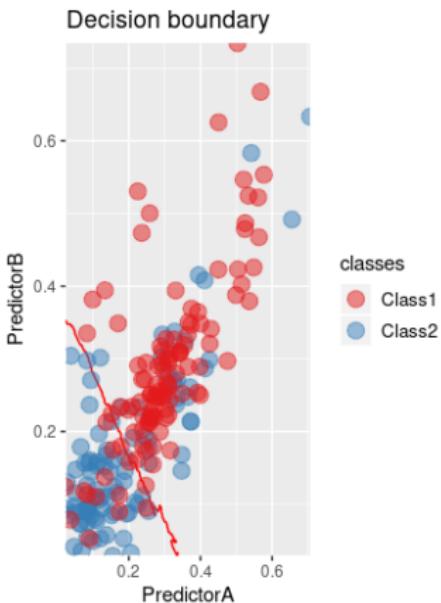
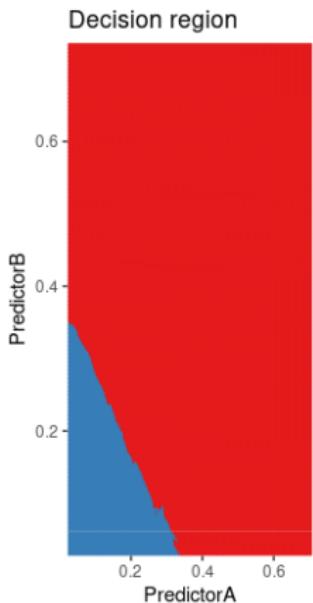
k-NN with k=85



Example: KNN

Error Estimation

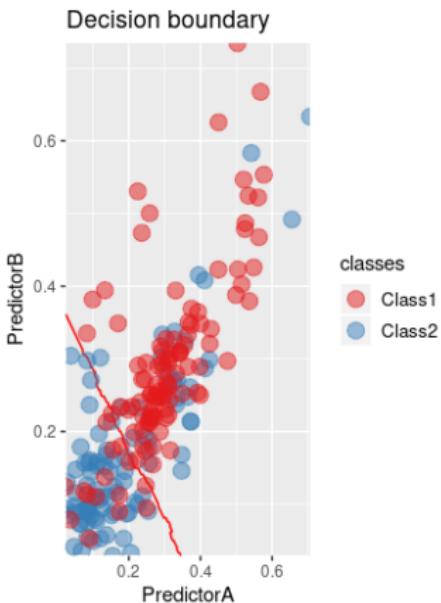
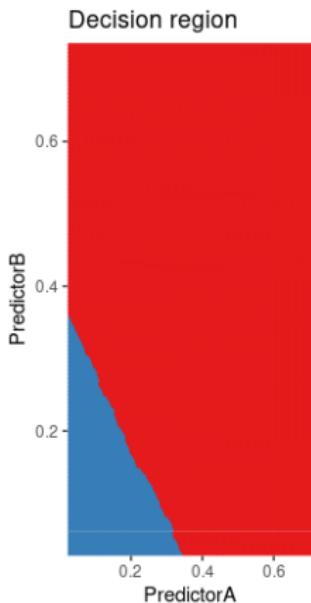
k-NN with k=101



Example: KNN

Error Estimation

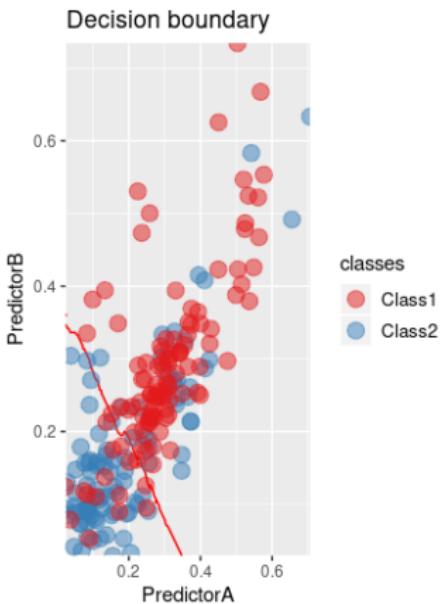
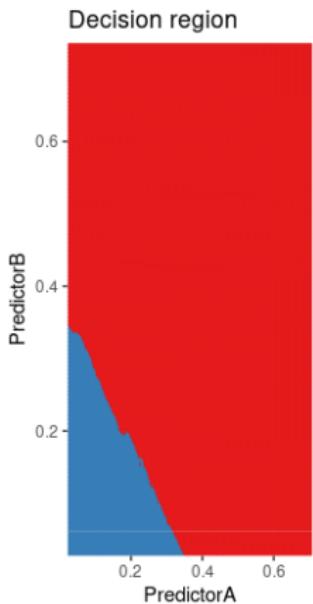
k-NN with $k=109$



Example: KNN

Error Estimation

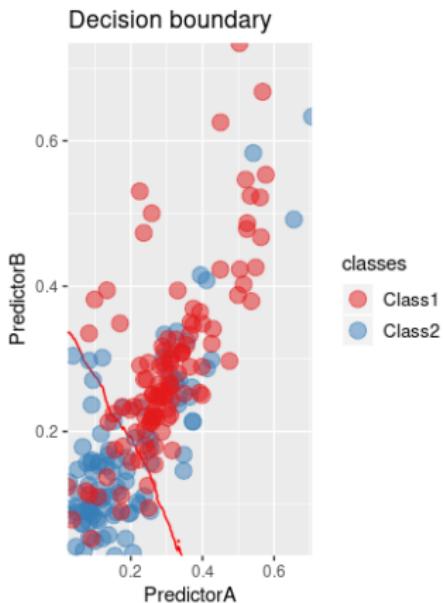
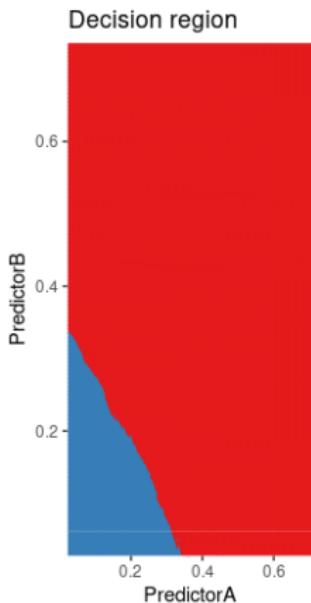
k-NN with k=117



Example: KNN

Error Estimation

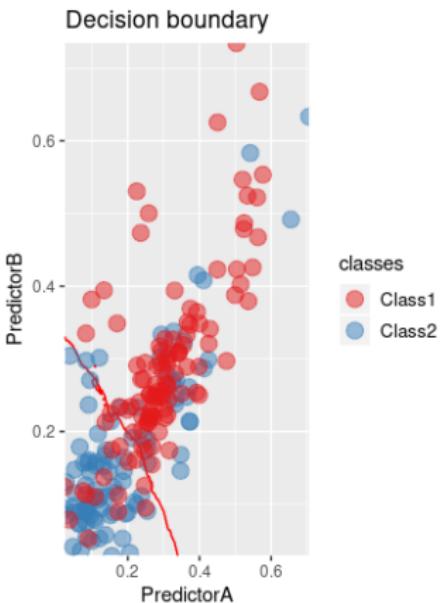
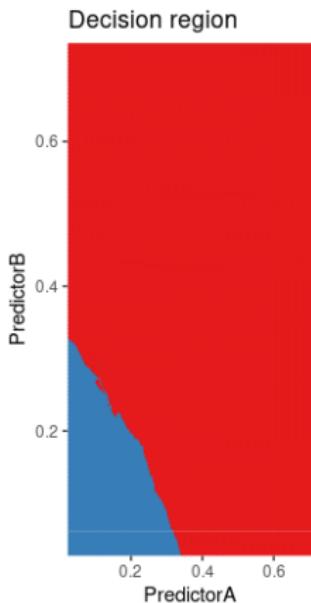
k-NN with k=125



Example: KNN

Error Estimation

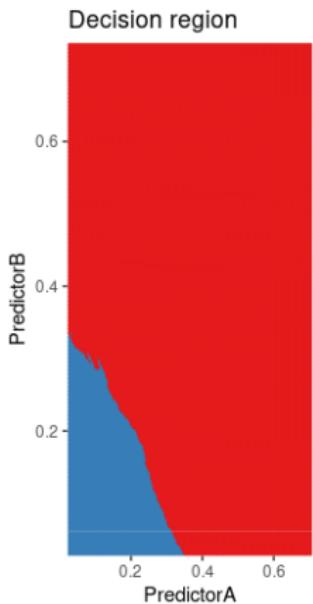
k-NN with k=133



Example: KNN

Error Estimation

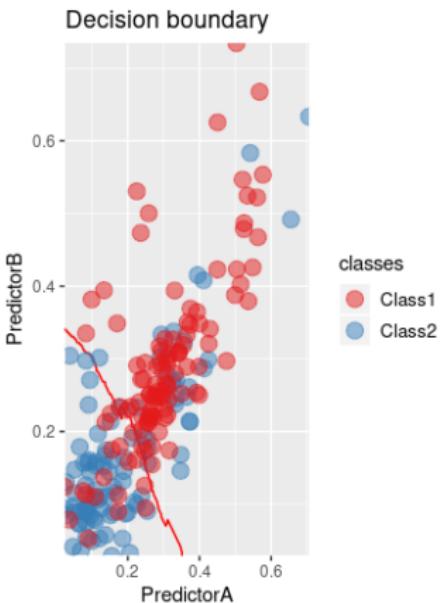
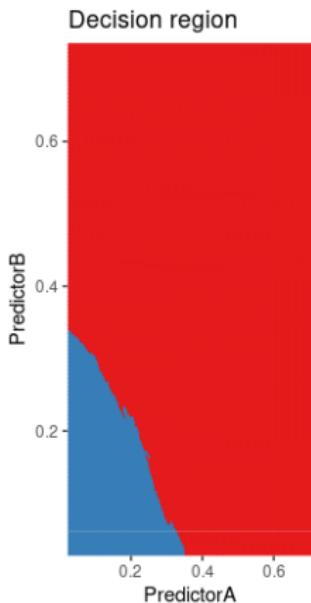
k-NN with k=141



Example: KNN

Error Estimation

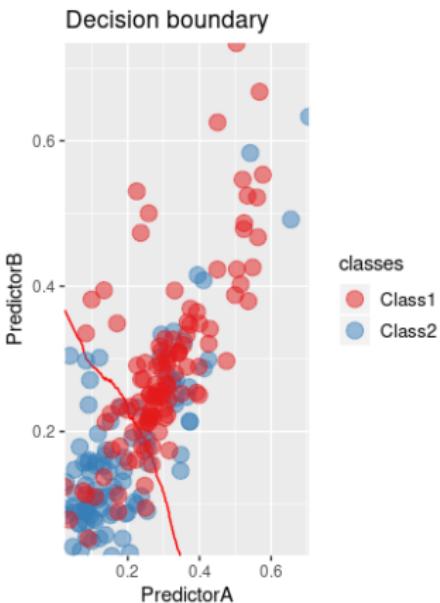
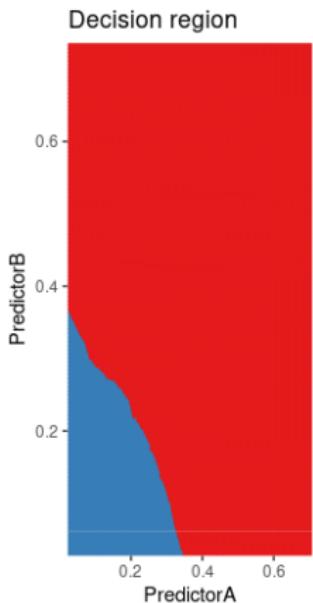
k-NN with k=149



Example: KNN

Error Estimation

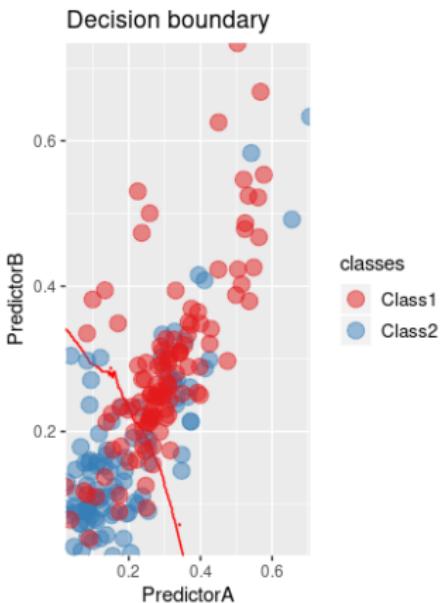
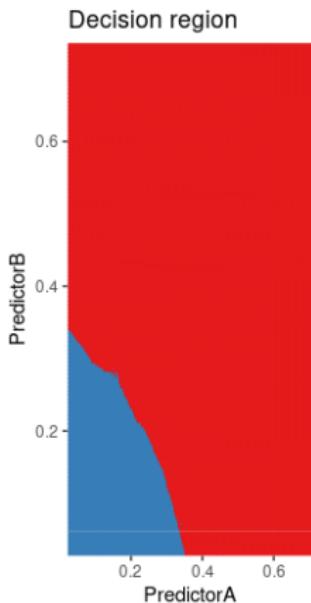
k-NN with k=157



Example: KNN

Error Estimation

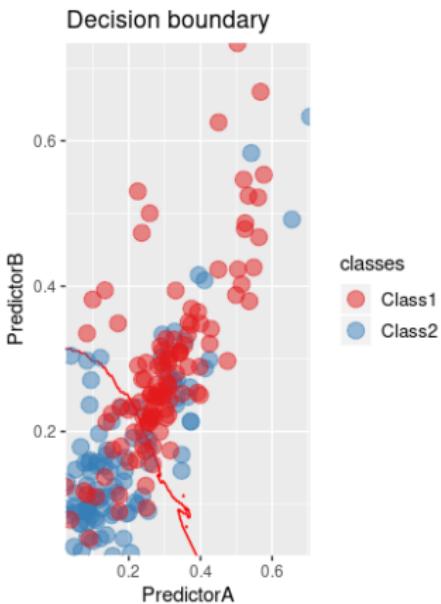
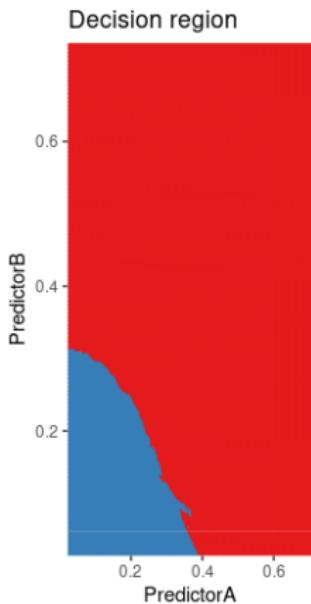
k-NN with k=165



Example: KNN

Error Estimation

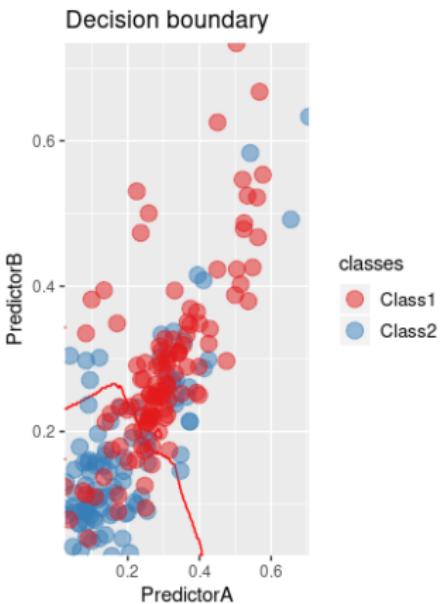
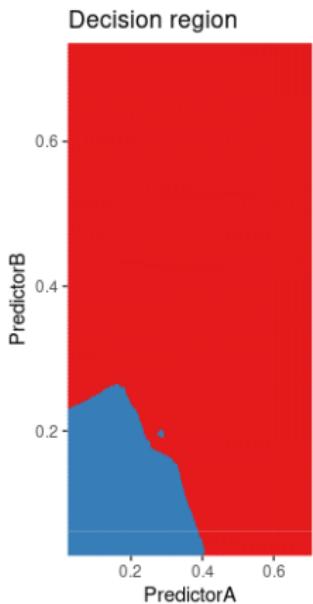
k-NN with k=173



Example: KNN

Error Estimation

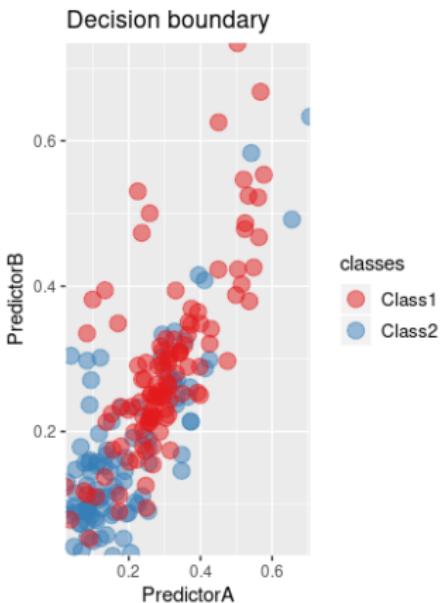
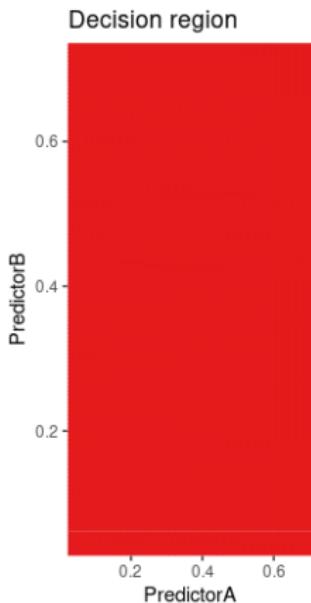
k-NN with k=181



Example: KNN

Error Estimation

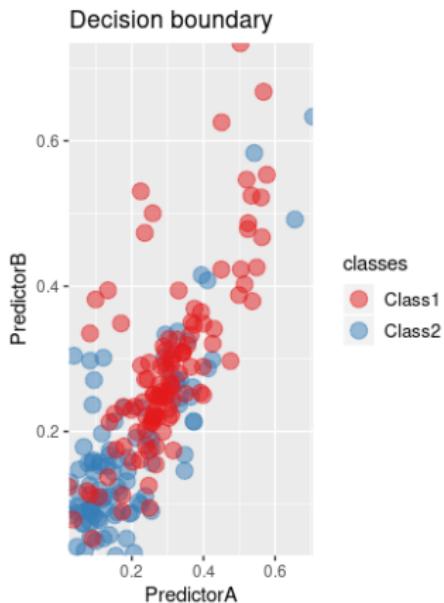
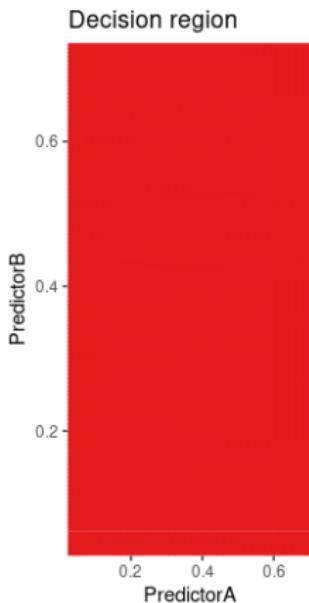
k-NN with k=189

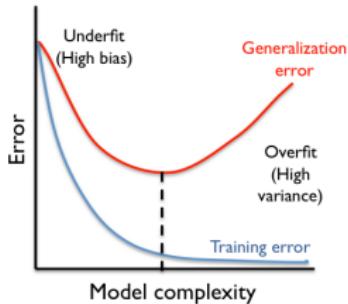


Example: KNN

Error Estimation

k-NN with $k=197$





Error behaviour

- Learning/training error (error made on the learning/training set) decays when the complexity of the **method** increases.
- Quite different behavior when the error is computed on new observations (generalization error).
- Overfit for complex methods: parameters learned are too specific to the learning set!
- General situation! (Think of polynomial fit...)
- Need to use a different criterion than the training error!

Predictor Error Estimation

- **Goal:** Given a predictor f assess its quality.
- **Method:** Hold-out error computation (/ Error correction).
- **Usage:** Compute an estimate of the error of a selected f using a **test set** to be used to monitor it in the future.
- Basic block very well understood.

Method Selection

- **Goal:** Given a ML method assess its quality.
- **Method:** Cross Validation (/ Error correction)
- **Usage:** Compute error estimates for several ML methods using **training/validation sets** to choose the most promising one.
- Estimates can be pointwise or better intervals.
- Multiple test issues in method selection.

Two Approaches

- **Cross validation:** Very efficient (and almost always used in practice!) but slightly biased as it target uses only a fraction of the data.
- **Correction approach:** use empirical loss criterion but *correct* it with a term increasing with the complexity of \mathcal{S}

$$R_n(\hat{f}_{\mathcal{S}}) \rightarrow R_n(\hat{f}_{\mathcal{S}}) + \text{cor}(\mathcal{S})$$

and choose the method with the smallest corrected risk.

Which loss to use?

- The loss used in the risk: most natural!
- The loss used to estimate $\hat{\theta}$: penalized estimation!



- **Very simple idea:** use a second learning/verification set to compute a verification error.
- Sufficient to remove the dependency issue!
- Implicit random design setting...

Cross Validation

- Use $(1 - \epsilon) \times n$ observations to train and $\epsilon \times n$ to verify!
- Possible issues:
 - Validation for a learning set of size $(1 - \epsilon) \times n$ instead of n ?
 - Unstable error estimate if ϵn is too small ?
- Most classical variations:
 - Hold Out,
 - Leave One Out,
 - V -fold cross validation.

Principle

- Split the dataset \mathcal{D} in 2 sets $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$ of size $n \times (1 - \epsilon)$ and $n \times \epsilon$.
- Learn \hat{f}^{HO} from the subset $\mathcal{D}_{\text{train}}$.
- Compute the empirical error on the subset $\mathcal{D}_{\text{test}}$:

$$\mathcal{R}_n^{HO}(\hat{f}^{HO}) = \frac{1}{n\epsilon} \sum_{(\underline{X}_i, Y_i) \in \mathcal{D}_{\text{test}}} \ell(Y_i, \hat{f}^{HO}(\underline{X}_i))$$

Predictor Error Estimation

- Use \hat{f}^{HO} as predictor.
- Use $\mathcal{R}_n^{HO}(\hat{f}^{HO})$ as an estimate of the error of this estimator.

Method Selection by Cross Validation

- Compute $\mathcal{R}_n^{HO}(\hat{f}_S^{HO})$ for all the considered methods,
- Select the method with the smallest CV error,
- Reestimate the \hat{f}_S with all the data.

Principle

- Split the dataset \mathcal{D} in 2 sets $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$ of size $n \times (1 - \epsilon)$ and $n \times \epsilon$.
- Learn \hat{f}^{HO} from the subset $\mathcal{D}_{\text{train}}$.
- Compute the empirical error on the subset $\mathcal{D}_{\text{test}}$:

$$\mathcal{R}_n^{HO}(\hat{f}^{HO}) = \frac{1}{n\epsilon} \sum_{(\underline{X}_i, Y_i) \in \mathcal{D}_{\text{test}}} \ell(Y_i, \hat{f}^{HO}(\underline{X}_i))$$

- Only possible setting for error estimation.

Hold Out Limitation for Method Selection

- Biased toward simpler method as the estimation does not use all the data initially.
- Learning variability of $\mathcal{R}_n^{HO}(\hat{f}^{HO})$ not taken into account.

V-fold Cross Validation

Error Estimation



Principle

- Split the dataset \mathcal{D} in V sets \mathcal{D}_v of almost equal size.
- For $v \in \{1, \dots, V\}$:
 - Learn \hat{f}^{-v} from the dataset \mathcal{D} minus the set \mathcal{D}_v .
 - Compute the empirical error:

$$\mathcal{R}_n^{-v}(\hat{f}^{-v}) = \frac{1}{n_v} \sum_{(\underline{X}_i, Y_i) \in \mathcal{D}_v} \ell(Y_i, \hat{f}^{-v}(\underline{X}_i))$$

- Compute the average empirical error:

$$\mathcal{R}_n^{CV}(\hat{f}) = \frac{1}{V} \sum_{v=1}^V \mathcal{R}_n^{-v}(\hat{f}^{-v})$$

- Estimation of the quality of a method not of a given predictor.
- Leave One Out : $V = n$.

Analysis (when n is a multiple of V)

- The $\mathcal{R}_n^{-v}(\hat{f}^{-v})$ are identically distributed variable but are not independent!
- Consequence:

$$\mathbb{E} [\mathcal{R}_n^{CV}(\hat{f})] = \mathbb{E} [\mathcal{R}_n^{-v}(\hat{f}^{-v})]$$

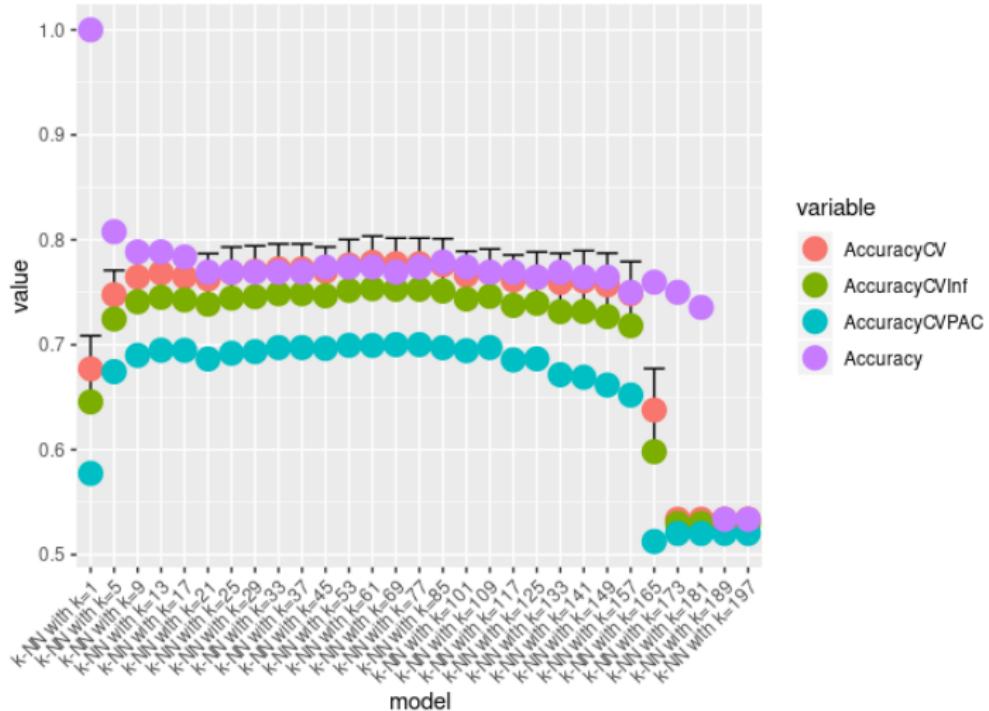
$$\text{Var} [\mathcal{R}_n^{CV}(\hat{f})] = \frac{1}{V} \text{Var} [\mathcal{R}_n^{-v}(\hat{f}^{-v})]$$

$$+ (1 - \frac{1}{V}) \text{Cov} [\mathcal{R}_n^{-v}(\hat{f}^{-v}), \mathcal{R}_n^{-v'}(\hat{f}^{-v'})]$$

- Average risk for a sample of size $(1 - \frac{1}{V})n$.
- Variance term much more complex to analyze!
- Fine analysis shows that the larger V the better...
- Accuracy/Speed tradeoff: $V = 5$ or $V = 10$!

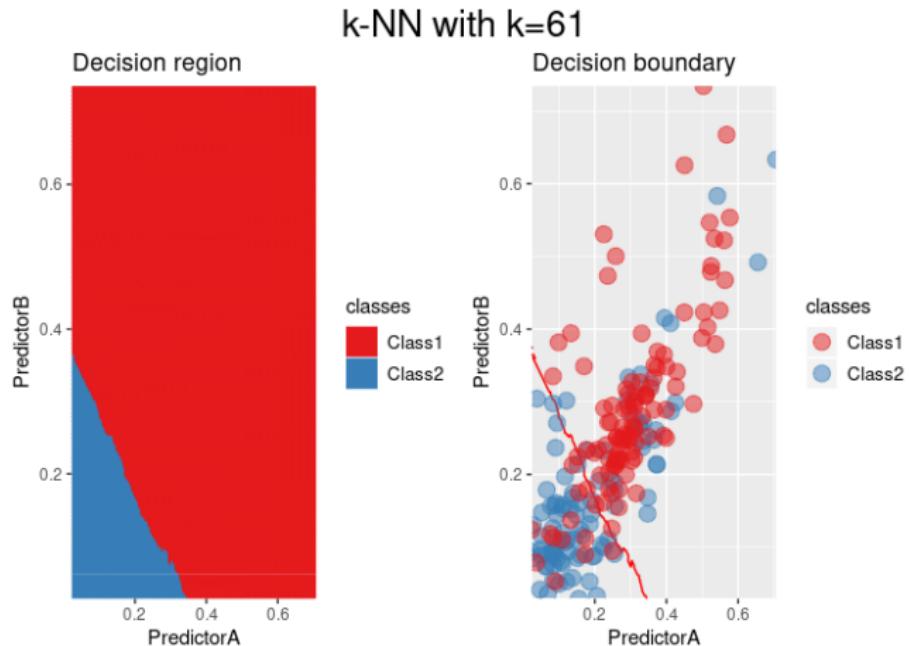
Cross Validation

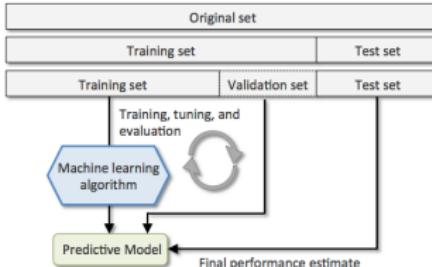
Error Estimation



Example: KNN ($\hat{k} = 61$ using cross-validation)

Error Estimation





- **Selection Bias Issue:**

- After method selection, the cross validation is biased.
- Furthermore, it qualifies the method and not the final predictor.
- Need to (re)estimate the error of the final predictor.

(Train/Validation)/Test strategy

- **Split** the dataset in two a (Train/Validation) and Test.
- Use **CV** with the (Train/Validation) to **select a method**.
- Train this method on (Train/Validation) to **obtain a single predictor**.
- Estimate the **performance of this predictor** on Test.

- Empirical loss of an estimator computed on the dataset used to choose it is biased!
- Empirical loss is an optimistic estimate of the true loss.

Risk Correction Heuristic

- Estimate an upper bound of this optimism for a given family.
 - Correct the empirical loss by adding this upper bound.
-
- **Rk:** Finding such an upper bound can be complicated!
 - Correction often called a **penalty**.

Penalized Loss

- Minimization of

$$\operatorname{argmin}_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^n \ell(Y_i, f_\theta(\underline{X}_i)) + \text{pen}(\theta)$$

where $\text{pen}(\theta)$ is an error correction (penalty).

Penalties

- Upper bound of the optimism of the empirical loss
- Depends on the loss and the framework!

Instantiation

- Mallows Cp: Least Squares with $\text{pen}(\theta) = 2\frac{d}{n}\sigma^2$.
- AIC Heuristics: Maximum Likelihood with $\text{pen}(\theta) = \frac{d}{n}$.
- BIC Heuristics: Maximum Likelihood with $\text{pen}(\theta) = \log(n)\frac{d}{n}$.
- Structural Risk Minimization: Pred. loss and clever penalty.

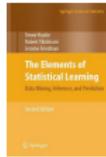
Outline

1 Introduction

2 Supervised Learning

3 Error Estimation

4 References



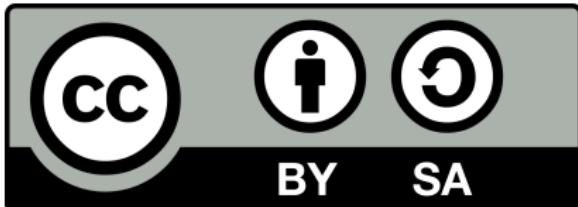
T. Hastie, R. Tibshirani, and J. Friedman.
The Elements of Statistical Learning.
Springer Series in Statistics, 2009



M. Mohri, A. Rostamizadeh, and A. Talwalkar.
Foundations of Machine Learning.
MIT Press, 2012



A. Géron.
Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow (2nd ed.)
O'Reilly, 2019



Creative Commons Attribution-ShareAlike (CC BY-SA 4.0)

- You are free to:
 - **Share:** copy and redistribute the material in any medium or format
 - **Adapt:** remix, transform, and build upon the material for any purpose, even commercially.
- Under the following terms:
 - **Attribution:** You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
 - **ShareAlike:** If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
 - **No additional restrictions:** You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Contributors

- S. Boucheron, A.K. Fermin, S. Gadat, S. Gaiffas, A. Guilloux, Ch. Kerbin, E. Le Pennec, E. Matzner, E. Scornet and X Exed team.