Fondamentaux de l'Apprentissage Automatique

Lecturer: Yann Chevaleyre
Scribe: Ellington Kirby

Lecture n°1 #
30/09/2020

1 Formalization of Supervised Learning

Definition 1. *Commonly used notations*

- $\mathcal{X}$: The set inputs. Also referred to as the set of examples. For example $\mathcal{X} = \mathbb{R}^d$, representing a continuous space of vectors, where each vector could be thought of as representing a set of features.
- $\mathcal{Y}$: The set of labels
- $\hat{\mathcal{Y}}$: The set of predictions
- $S = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$: The dataset
- $h : \mathcal{X} \to \mathcal{Y}$: A classifier
- $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$: A loss function, which represents the strength of one single prediction of the model. For example $\ell^{0/1}(h(x), y) = \mathbf{1}_{[h(x) \neq y]}$ is the loss function of a standard binary classifier, and $\ell^{MSE}(h(x), y) = (h(x) - y)^2$ is the standard loss function of a Regression problem.

Some example classifiers :

- Linear Regression : $\mathcal{X} \in \mathbb{R}^d$, $\mathcal{Y} \in \mathbb{R}$, $\hat{\mathcal{Y}} \in \mathbb{R}$.
- Binary Classifier : $\mathcal{X} \in \mathbb{R}^d$, $\mathcal{Y} \in \{0, 1\}$, $\hat{\mathcal{Y}} \in \{0, 1\}$. Note : in some cases the classes are recorded as $\mathcal{Y} \in \{-1, 1\}$.
- Binary Classifier with Probabilities : $\mathcal{X} \in \mathbb{R}^d$, $\mathcal{Y} \in \{0, 1\}$, $\hat{\mathcal{Y}} \in [0, 1]$. We predict $h(x) = \hat{p}(Y = 1 | X = x)$

2 Empirical Risk Minimization

Given a loss function, we encounter a problem : how can we evaluate the loss of our entire dataset ?

Definition 2. *Empirical Risk*

$\hat{\mathcal{R}}_n(f) = \frac{1}{n} \sum_i^n \ell(f(x_i), y_i)$

Definition 3. *Empirical Risk Minimization*

An algorithm is said to be an Empirical Risk Minimizer (ERM) if it outputs a function which minimizes $\hat{\mathcal{R}}_n(f)$ by minimizing over the set of all possible functions $\mathcal{F}$.

$\hat{f} = \arg\min_{f \in \mathcal{F}} \hat{\mathcal{R}}_n(f)$

This is referred to as the Empirical Risk Minimization Principle [1].

Only a subset of algorithms are strictly ERM : examples include unregularized linear regression and basic Decision Trees. Some frequently seen algorithms are not ERM, such as K-Nearest Neighbors (KNN).

Examples of ERM algorithms as minimization problems :

- Linear Regression : for the function space $\mathcal{F} \to \{x \to ax + b | a \in \mathbb{R}^d, b \in \mathbb{R}\}$ we have $\hat{a}, \hat{b} = \arg\min_{a,b} \frac{1}{n} \sum_i^n (ax_i + b - y_i)^2$

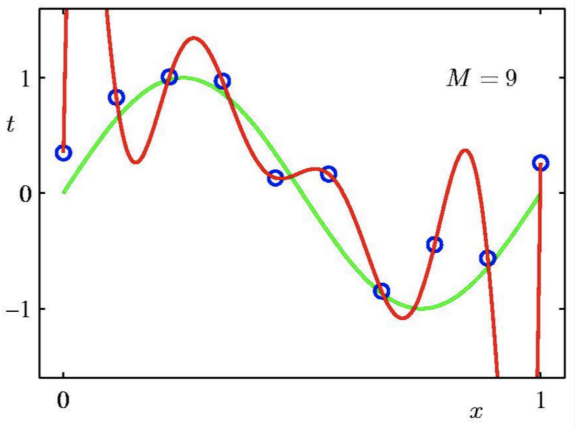


FIGURE 1 – Over fitting with $M = 9$

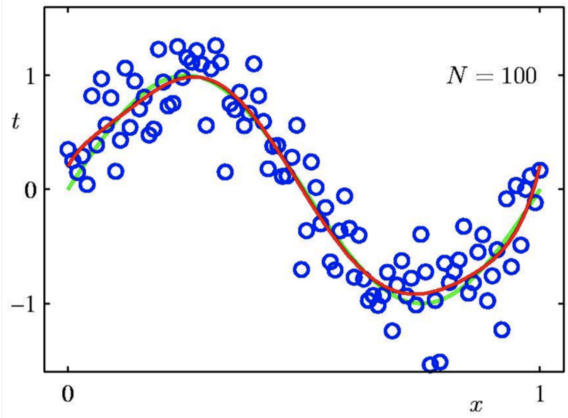


FIGURE 2 – $M = 9$ but with far more examples overfitting is avoided.

— Linear Classifier with 0-1 Loss : using the same function space above

$\hat{a}, \hat{b} = \arg\min_{a,b} \frac{1}{n} \sum_i^n \mathbf{1}_{[h(x) \neq y_i]}$ where $h(x) = \{1 \text{ if } ax_i + b > 0, \text{ else } 0\}$

Exploration : Are ERM Algorithms "Good" ?

We examine a polynomial curve fitting algorithm. We are attempting to learn the best parameters $w_0 ... w_m$ for $f(x) = w_0 + w_1 x + w_2 x^2 + ... + w_m x^m$, $\mathcal{X} = \mathcal{Y} = \hat{\mathcal{Y}} = \mathbb{R}$, using the squared loss $\ell^{sq}(\hat{y}, y) = (\hat{y} - y)^2$. We let M be a hyper parameter.

Figure 1 shows the case where we have chosen $M = 9$ and have "catastrophically overfit" our data. Here it is important to note that $\hat{\mathcal{R}}_n(f) = 0$, meaning we have completely minimized the empirical risk. However when compared to the green curve representing the true definition of our function, we can see we are far from the correct prediction. This effect can be minimized by using more examples in our training, as is seen in Figure 2.

These examples illustrate that pure ERM alone is not a sufficient learning strategy, because of overfitting, meaning that having a very low empirical risk does not guarantee prediction accuracy on future data.

Definition 4. *Overfitting*

A model can be said to be "overfit" when the performance of the model on training data is high (low ER), but poor on new data from the same source distribution (high loss). In

some sense the model can be said to have learned the noise of the distribution.

A method for estimating if a model is overfit is the "Train-Test" split method. A portion of the dataset is reserved and never used for training. After training the model is evaluated on this held out test set. A model that performs well in training but poorly on the test set can be said to have overfit. However this can only be done once training has been completed, which leaves the question of how to detect overfitting during training ?

3 Measuring Performance of a Classifier

In the statistical Fixed Design Setting we assume there exists an optimal function $f^* : \mathcal{X} \to \mathcal{Y}$ such that $y_i = f^*(x) + \epsilon_i$. The y_i are then the deterministic output of this function $f^(x)$.

In the Machine Learning setting we instead assume that there exists an unknown data generating distribution $\mathbb{P}$ over $\mathcal{X} \times \mathcal{Y}$, with all input and output pairs drawn independently and identically distributed (IID). This assumption of the existence of a joint probability distribution over $\mathcal{X}$ and $\mathcal{Y}$ introduces uncertainty in our predictions. The y_i are a random variable conditioned on a fixed x. This allows us to define the notion of True Risk.

Definition 5. *True Risk*

$\mathcal{R}(f) = \mathbb{E}_{\mathcal{X},\mathcal{Y}}[\ell(f(x), y)] = \int_{X,Y} \ell(f(x), y)\ d\mathbb{P}(X, Y)$

Note that our notation here is not a conditional expectation, and instead implies that $\mathcal{X}$ and $\mathcal{Y}$ are drawn from $\mathbb{P}$, in other words $\mathbb{E}_{\mathcal{X},\mathcal{Y}}[\cdot]$ implies $\mathcal{X}, \mathcal{Y} \sim \mathbb{P}[\cdot]$

Here ℓ represents our loss function, for example the squared loss in the case of linear regression. In Empirical Risk Minimization our underlying objective is to minimize the True Risk. However computing the True Risk directly is impossible, because it would require direct knowledge of our unknown distribution $\mathbb{P}(\mathcal{X}, \mathcal{Y})$. Thus we use our estimation $\hat{\mathcal{R}}_n$.

This implies that ERM is only useful in the cases where Estimated Risk is close to True Risk. Thus we must try to understand how far $\hat{\mathcal{R}}$ is from $\mathcal{R}$.

— Can we use the Law of Large Numbers to state that $\hat{\mathcal{R}}$ converges to $\mathcal{R}$ in expectation ? No.

Let $\mathcal{S} = \{(x_1, y_1)...(x_n, y_n)\}$ be our set of samples.

The Strong Law of Large Numbers states that if a set of values $z_1...z_n$ are drawn IID from a distribution with some expected value $\mathbb{E}[z]$, then the average of these values converges to $\mathbb{E}[z]$ as $n \to \inf$

Comparing our definitions of Empirical Risk and True Risk we can see a similar comparison. In the Empirical Risk we average the loss function of f, while in the True Risk we take the expectation of the loss of f. Thus if f is independent of S then we fit the Strong Law of Large Numbers and can say that Empirical Risk converges to True Risk with large enough N.

However via the Empirical Risk Minimization Principle, we compute our loss function via $\hat{f}$. $\hat{f}$ has been learned from our set of samples $\mathcal{S}$ as the function that minimizes $\hat{\mathcal{R}}$. Thus $\hat{f}$ is not independent from $\mathcal{S}$.

Thus $\hat{\mathcal{R}}_N(\hat{f}) \not\to \mathcal{R}(\hat{f})$ as $N \to \inf$

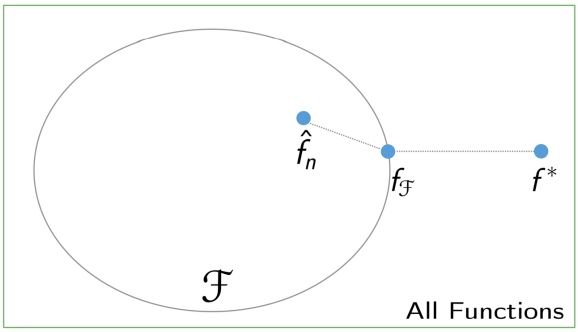


FIGURE 3 – Visualization of our three different functions

4 Risk Decomposition

Let $\mathcal{F}$ be the space of a specific class of functions. We can identify three variations of functions found by risk minimization based on the choice of function space for minimization.

$f^* = \arg\min_f \mathbb{E}_{\mathcal{X},\mathcal{Y}}[\ell(f(\mathcal{X},\mathcal{Y}]$

$f_{\mathcal{F}} = \arg\min_{f\in\mathcal{F}} \mathbb{E}_{\mathcal{X},\mathcal{Y}}[\ell(f(\mathcal{X},\mathcal{Y}]$

$\hat{f}_n = \arg\min_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n} \ell(f(x_i, y_i))$

Thus f^* represents the optimal function over all possible functions, $f_{\mathcal{F}}$ is the optimal function over the chosen class of functions $\mathcal{F}$, and $\hat{f}_n$ is our estimated best function. Figure 3 visually shows the relationship of these three functions.

We can thus decompose our risk, breaking down the Risk measurement of our model into the contribution from different sources.

Definition 6. *Error Decomposition of Empirical Risk :*

$\mathcal{R}(\hat{f}_n) = \mathcal{R}(\hat{f}_n) - \mathcal{R}(f_{\mathcal{F}}) + \mathcal{R}(f_{\mathcal{F}}) - \mathcal{R}(f^*) + \mathcal{R}(f^*)$

Note we find this decomposition by adding and subtracting $\mathcal{R}(f_{\mathcal{F}})$, $\mathcal{R}(f^*)$ from our definition of Empirical Risk.

Definition 7. *Approximation Error of $\mathcal{F}$*

$\mathcal{R}(f_{\mathcal{F}}) - \mathcal{R}(f^*)$

The approximation error represents the error introduced by the choice of class of functions $\mathcal{F}$.

Definition 8. *Estimation error*

$\mathcal{R}(\hat{f}_n) - \mathcal{R}(f_{\mathcal{F}})$

The estimation error represents the error introduced by working on data drawn from the distribution but not directly on the distribution itself.

Definition 9. *Bayes Error*

$\mathcal{R}(f^*)$

The Bayes Error represents the error of the best approximable function on our distribution. This error is unavoidable in any predictor.

Later in the notes we will see an example of the Error decomposition of a Decision Tree classifier which will explain further the comparison of Approximation Error and Estimation Error. The important takeaway is that because the Bayes Error is unavoidable, minimizing risk requires a tradeoff between Approximation Error and the Estimation Error.

Definition 10. *Bayes Predictor*
$f^* \in \arg\min_{f \in \{\textit{measureable functions}\}} \mathcal{R}(f)$

The Bayes Predictor achieves the minimum possible risk over all functions. Can be referred to as the "target function" as it is the best possible prediction function.

4.1 Reminder on Properties of Expectations

Let C be any predicate, then
— $\mathbb{E}[\mathbf{1}(C)] = \mathbb{P}(C)$ where $\mathbf{1}(C)$ is the indicator function, returning 1 if the C holds, and 0 otherwise

Let $\mathcal{X},\ \mathcal{Y}$ be the domains of random variables $X,\ Y$ respectively, and let $g : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ be a real valued function. If $\mathcal{X},\ \mathcal{Y}$ are continuous spaces with a distribution P which admits a joint density function $p(X, Y)$ then :
— The Expectation of $g(X, Y)$ is :
$\mathbb{E}_{X,Y}[g(X, Y)] = \int_{\mathcal{X}\times\mathcal{Y}} g(X, Y)p(X, Y)dXdY$
— The Conditional Expectation of $g(X, Y)$ given X is :
$\mathbb{E}_Y[g(X, Y)|X] = \int_{\mathcal{Y}} g(X, Y)p(Y|X)dY$
— The Law of Total Expectation States :
$\mathbb{E}_{X,Y}[g(X, Y)] = \mathbb{E}_X[\mathbb{E}_Y[g(X, Y)|X]]$

If $\mathcal{X}$ is continuous while $\mathcal{Y}$ is discrete, with joint density function $p(X, Y)$ then :
— The Expectation of $g(X, Y)$ is :
$\mathbb{E}_{X,Y}[g(X, Y)] = \sum_{Y\in\mathcal{Y}} \int_{\mathcal{X}} g(X, Y)p(X, Y)dX$
— The Conditional Expectation over Y of $g(X, Y)$ given X :
$\mathbb{E}_Y[g(X, Y)|X] = \sum_{Y\in\mathcal{Y}} g(X, Y)p(Y|X)dY$
— The Law of Total Expectation States (unchanged) :
$\mathbb{E}_{X,Y}[g(X, Y)] = \mathbb{E}_X[\mathbb{E}_Y[g(X, Y)|X]]$

4.2 Example : Bayes Predictor for Binary Classifier

For the binary classifier case we have $\mathcal{Y} = \hat{\mathcal{Y}} = \{0, 1\}$.
We define the 0-1 loss as
$\ell(\hat{y}, y) = \mathbf{1}(\hat{y} \neq y) := \begin{cases} 1 & \text{if } \hat{y} \neq y \\ 0 & \text{otherwise} \end{cases}$
Then the Bayes Predictor can be found as
$f^*(x) \in \arg\max_{c\in\{0,1\}} P(Y = c|X = x)$

Démonstration. $R(f) = \mathbb{E}_{X,Y}[\mathbf{1}(f(X) \neq Y]$
$= \mathbb{E}_X[\mathbb{E}_Y[\mathbf{1}(f(X) \neq Y]]$
$= \mathbb{E}_X[P_Y(f(X) \neq Y|X)]$ with $Y = \{0, 1\}$
$= \mathbb{E}_X[P_Y(f(X) = 1, Y = 0|X) + P_Y(f(X) = 0, Y = 1|X)]$
Note that $f(X) = 1$ or $f(X) = 0$ is not a probability, so we can transform this into
$= \mathbb{E}_X[P_Y(Y = 0|X) \cdot \mathbf{1}(f(X) = 1) + P_Y(Y = 1|X) \cdot \mathbf{1}(f(X) = 0)]$
$= \mathbb{E}_X[(1 - P_Y(Y = 1|X) \cdot \mathbf{1}(f(X) = 1) + P_Y(Y = 1|X) \cdot \mathbf{1}(f(X) = 0)]$
Let $\phi(X) = (1 - P_Y(Y = 1|X) \cdot \mathbf{1}(f(X) = 1) + P_Y(Y = 1|X) \cdot \mathbf{1}(f(X) = 0)$
Then $R(f) = \mathbb{E}_X[\phi(X)]$ where $\phi(X) = \begin{cases} 1 - P_Y(Y = 1|X) & \text{if } f(X) = 1 \\ P(Y = 1|X) & \text{if } f(X) = 0 \end{cases}$
The function f^* which minimizes this is the Bayes Predictor and is

$f^* = \begin{cases} 1 & \text{if } P_Y(Y=1|X) > P_Y(Y=0|X) \\ 0 & \text{if } P_Y(Y=0|X) > P_Y(Y=1|X) \end{cases} = \arg\max_{c \in \{0,1\}} P(Y=c|X=x)$

We can then define the Bayes Risk as $\mathcal{R}(f^*) = \mathbb{E}_X[\min(P_Y(Y=1|X), P_Y(Y=0|X))]$

□

4.3 Example : Bayes Predictor for Least Squares

We consider the classic regression case with a Least Squares loss function $\ell(\hat{y}, y) = (\hat{y} - y)^2$ with $\mathcal{Y} \in \mathbb{R}$.

The Bayes Predictor can thus be found as $f^*(x) = \mathbb{E}[Y|X=x]$

Démonstration. $R(f) = \mathbb{E}_{X,Y}[(f(X)-Y)^2] = \mathbb{E}_X[\mathbb{E}_Y[(f(X)-Y)^2|X]]$

$= \mathbb{E}_X[\mathbb{E}_Y[(f(X) - \mathbb{E}[Y|X] + \mathbb{E}[Y|X] - Y)^2|X]]$

Via adding and subtracting $\mathbb{E}[Y|X]$

$= \mathbb{E}_X[\mathbb{E}_Y[(f(X) - \mathbb{E}[Y|X])^2 + (\mathbb{E}[Y|X] - Y)^2 + 2((f(X) - \mathbb{E}[Y|X])(\mathbb{E}[Y|X] - Y))|X]]$

Considering the term, we can see that ultimately $2((f(X) - \mathbb{E}[Y|X])(\mathbb{E}[Y|X] - Y)) = 0$

This is because given X, $\mathbb{E}_Y[f(X) - \mathbb{E}[Y|X]$ is constant and

$\mathbb{E}_Y[\mathbb{E}[Y|X] - Y|X] = \mathbb{E}_Y[Y|X] - \mathbb{E}[Y|X] = 0$

Thus $2((f(X) - \mathbb{E}[Y|X])(\mathbb{E}[Y|X] - Y)) = 0$

Continuing, we have

$\mathbb{E}_X[\mathbb{E}_Y[(f(X) - \mathbb{E}[Y|X])^2 + (\mathbb{E}[Y|X] - Y)^2 + 0]]$

$= \mathbb{E}_X[\mathbb{E}_Y[(f(X) - \mathbb{E}[Y|X])^2|X] + \mathbb{E}_Y[(\mathbb{E}[Y|X] - Y)^2|X]]$

Our left term can be eliminated when $f(X) = \mathbb{E}[Y|X]$, thus our Bayes Predictor which minimizes the above is $f^* = \mathbb{E}[Y|X]$

Thus our Bayes Error is $\mathcal{R}(f^*) = \mathbb{E}_X[\mathbb{E}_Y[(Y - \mathbb{E}[Y|X])^2|X]]$

Where $\mathbb{E}_Y[(Y - \mathbb{E}[Y|X])^2|X] = \text{var}(Y|X)$, so our Bayes Error is

$\mathcal{R}(f^*) = \mathbb{E}_X[\text{var}(Y|X)]$ □

4.4 Example : Risk Decomposition of Decision Trees

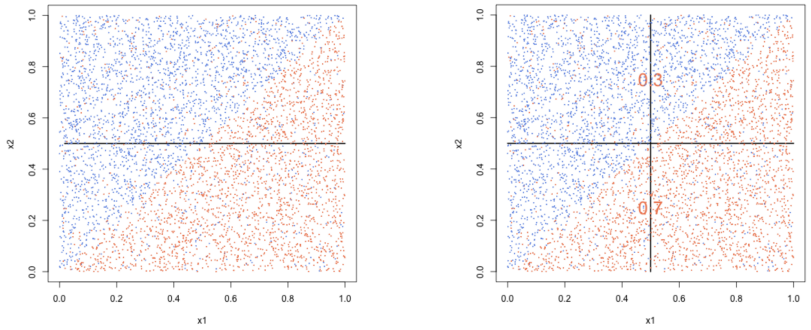
We consider a binary classification problem with $S \in [0,1]^2$ (all examples are coordinate pairs between 0 and 1).

Let $\mathcal{F}_d := \{$the set of all decision tree classifiers on with depth $\geq d\}$

We consider subsets of $\mathcal{F}_d$ in increasing order of depth, i.e.

$\mathcal{F}_1 \subset \mathcal{F}_2 \subset ... \subset \mathcal{F}_{15}$

We assume that the Bayes Error is 0.1 and our goal is to calculate the Approximation Error and Estimation Error for different subsets of $\mathcal{F}_d$ with the motivation of understand how different components of Error evolve as we increase the complexity of our prediction function.



(a) The best $\mathcal{F}_1$ classifier

(b) The $\mathcal{F}_1$ classifier drawn with the graph broken into quadrants.

FIGURE 4

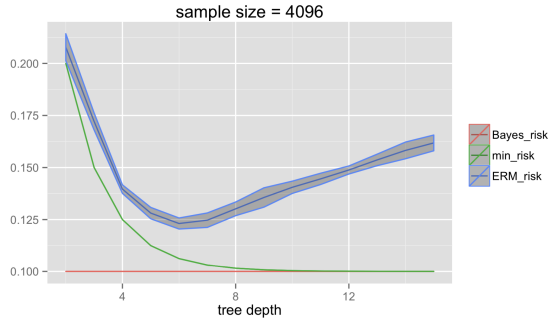


FIGURE 5 – Error decomposition of decision tree binary classifer with increasing depth.

4.4.1 $\mathcal{F}_1$

Figure 4 shows the output of a decision tree classifier with a depth of 1 on the dataset. To calculate the Approximation and Estimation errors it is helpful to visualize the dataset and predictor on a graph broken into four quadrants.

To calculate the Approximation error, we must first calculate $\mathcal{R}_{\mathcal{F}}$, the Risk of our best $\mathcal{F}_1$ classifier. Once this value is found, we can subtract the Bayes Error to find the Approximation error.

Note that each quadrant of our dataset can be said to be equiprobable, because S was drawn IID.

$\mathbb{P} = \mathbb{P}(\text{error}|Q_1)\mathbb{P}(Q_1) + \mathbb{P}(\text{error}|Q2)\mathbb{P}(Q_2) + \mathbb{P}(\text{error}|Q_3)\mathbb{P}(Q_3) + \mathbb{P}(\text{error}|Q_4)\mathbb{P}(Q_4)$

Where Q_i represents Quadrant i.

$= (.5)(.25) + (.1)(.25) + (.5)(.25) + (.1)(.25) = 0.3$

Each quadrant has a $\frac{1}{4}$ chance of being chosen, and the chance of an error in a specific quadrant can be seen from the proportion of classes that lie within each quadrant. For example in Q_1 there is an even split of red and blue, so the chance of an error in classification via our horizontal line is $\frac{1}{2}$

Thus the Approximation Error for $\mathcal{F}_1$ is $\mathcal{R}(f_{\mathcal{F}}) - \mathcal{R}(f^*) = 0.3 - 0.1 = 0.2$

Figure 5 highlights the error decomposition of our classifier as we increase the complexity of the model. The Approximation error steadily decreases, but after a depth of 6 the Esti-

mation error begins to increase. The figure highlights that at a certain point of complexity the model begins to overfit. Overfitting can thus be thought of as a condition in which the Estimation Error of the model increases at a rate faster than the decrease of the Approximation Error. This effect could be avoided by increasing the size of our training dataset [2].

5 Bounds On Risk

The machine learning community has developed two important questions to ask when evaluating the Risk of an algorithm.

Definition 11. *F Consistent*

An algorithm is said to be F Consistent when estimation error approaches zero as the number of samples becomes larger.
$\mathcal{R}(\hat{f}_N) \to \mathcal{R}(f_{\mathcal{F}})$ as $N \to \infty$

Definition 12. *Bayes Consistent*

An algorithm is said to be Bayes consistent if the True Risk converges to Bayes Risk.
$\mathcal{R}(\hat{f}_N) \to \mathcal{R}(f^*)$

5.1 Is the 1 Nearest Neighbor Algorithm (1NN) Bayes Consistent ?

Let $h_S^{NN}(x)$ be the 1NN classifier taking neighbors from S
Does $\mathcal{R}(h_S^{NN}(x)) \to \mathcal{R}(f^*)$
Important simplification :
we will consider the two class setting where $\mathbb{P}(y = 1|X) = P(y = 1)$
We study $\mathbb{E}_S[\mathcal{R}(h_S^{NN}(x))]$ where $S \sim P_S$, the distribution our dataset is drawn from.
We define Y_{NN} as the class of the Nearest Neighbor of x in S
$= \mathbb{E}_{S\sim P_S}[\mathbb{E}_X[\mathbb{P}_Y(y = 0|x) \cdot \mathbf{1}(Y_{NN} = 1) + \mathbb{P}_Y(y = 1|x) \cdot \mathbf{1}(Y_{NN} = 0)]]$
We note that $\mathbb{P}(y = 0|x)$ does not depend on S, and S only affects the Nearest Neighbor calculation, and can thus reposition the expectations.
$= \mathbb{E}_X[\mathbb{P}(y = 0|x) \cdot \mathbb{E}_S[\mathbf{1}(Y_{NN} = 1)|x] + \mathbb{P}(y = 1|x) \cdot \mathbb{E}_S[\mathbf{1}(Y_{NN} = 0)|x]]$
$= \mathbb{E}_X[\mathbb{P}(y = 0|x) \cdot \mathbb{P}(Y_{NN} = 1|x) + \mathbb{P}(y = 1|x) \cdot \mathbb{P}(Y_{NN} = 0|x)]$
$= \mathbb{E}_X[p(1 - p) + (1 - p)p]$
$= 2p(1 - p)$
Thus given our simplifying assumption, as $N \to \infty$,
$\mathbb{E}_S[\mathcal{R}(h_S^{NN}(x))] \to 2p(1 - p)$
However without this assumption, thus in the general case, we have :
$\mathbb{E}_S[\mathcal{R}(h_S^{NN}(x))] \to \mathbb{E}_X[2\mathbb{P}(y = 1|x)(1 - \mathbb{P}(y = 1|x))]$
Recall that in 4.2 found the Bayes Error of in binary classification to be :
$\mathcal{R}(f^*) = \mathbb{E}_X[\min(P_Y(Y = 1|X), P_Y(Y = 0|X))]$
If we assume a value of p, say $p = \frac{1}{4}$, then we find that
$\min(p, 1 - p) = \frac{1}{4}$ while $2p(1 - p) = \frac{3}{8}$
Thus 1NN is not Bayes Consistent.

Références

[1] V. Vapnik, "Principles of risk minimization for learning theory," *Advances in neural information processing systems*, vol. 4, 1991.

[2] W. Zheng, "Revisiting machine learning 1 : overfitting."