Machine learning

Lecture 1

Lecturer: Haim Permuter *Scribe: Gal Rattner*

In this lecture we introduce machine learning, define the relevant notations, and examine the popular machine learning algorithm K-Nearest Neighbors. Most of the material for this lecture was taken from the work of Cover and Hart on K-NN [1].

I. Introduction to Machine Learning

The goal of machine learning is to program computers to use training data or gained experience to solve given problems. We can broadly say that a machine learns whenever it changes its structure, program or data such that its future expected performance is improved. Many successful applications of machine learning already exist today including systems that predict stock prices, face recognition technology incorporated in digital cameras or facebook, speech recognition in Google Assistance, Siri or Alexa, safety systems in car (e.g., Mobiley), advertising on the web, auto-completion text, and ”spam” mail detection programs. Recently, we are also experiencing a huge success in Generative AI such Chat-GPT, Claude, Gemini and more. Today, machine learning is entering every engineering field that you may think about and is expected to have a great impact in the next decades since the technology is progressing significantly.

A. Type of learning problems

The type of problems that we try to solve by machine learning can be divided into two categories:

- **Classification problems:** Given stochastic observation x, classification is the problem of associating x with one class from among a set of classes.

  For example:

  - Associating a recorded speech with the speaker (speaker recognition).
  - Associating an image with a letter (digit recognition).

  - Associating an image with an object, e.g., recognizing a car (object recognition).

- **Regression problems:** Given stochastic observation x, regression associates a continuous value with x.

  For example:

  - Associating a price with a house for sale, using its location, size and other parameters.
  - Associating a price prediction with a stock, using its price over last few days, the most recent financial reports of the company, etc.
  - Forecasting temperatures based on the time of the year, measurements over the most recent days, etc.

In both regression and classification problems, we associate a label with a given test sample. In a classification problem, the label is chosen from a finite set, e.g., $\{1, 2, \ldots, M\}$ where $|\mathcal{M}| < \infty$. On the other hand, the label in a regression problem is a continuous value that has a defined sorting order, and one can define the distance between two different elements.

B. Training/Validation/Test Data

Classification and regression are both based on a training session comprising a set of training samples x_i and each sample has it's own fixed label l_i. The success rate of the classification is then evaluated with a set of validation samples $\{x_1, x_2, \ldots, x_n\}$ whose properties are similar to those of the training set. The evaluation of the classification/regression is done by comparing the original labels (using a loss function that we will explain later in the lecture) with the labels associated by the trained model. Usually, the learning model might still be changed (especially hyper-parameters, like the size or depth of the machine learning model) during the validation and therefore there is a need for final testing of the machine learning system on an additional data called the test data, where the model can not be changed.

C. Types of learning

Machine learning methods can be categorized into three main types of the data that it needs to learn from.

Supervised - In the training stage, the system is presented with labels for each sample. Sample-label couples $\{(x_1, l_1), (x_2, l_2), \ldots, (x_N, l_N)\}$ are given to the system along with the unlabeled test set $\{y_1, y_2, \ldots, y_M\}$. The system then associates labels with the test samples according to the statistic model based on the training set. The supervised learning system checks the correctness of its associating process by comparing the association results with the original test labels. Supervised learning is often used in classification and regression problems, for instance, in digit recognition, house price estimations, etc.

Unsupervised - In the training stage, the system is not given any fixed labels as inputs, and as such, the system must define relevant labels. This usually requires that the system learn the distribution of the sample set. Unsupervised learning is often used for segmentation, for instance, edge detection in image processing tasks, etc.

Reinforcement learning - An intermediate stage to supervised and unsupervised learning, reinforcement learning entails the system learning "on the fly" through its experience. For each attempt, the system receives some reward and then determines whether the attempt was a failure or a success. The system thus gains experience and learns which of its attempts were "good" by comparing between the rewards it received for the attempts. Reinforcement learning is used to train computers to be experts in defined tasks, for example, playing a game (e.g. https://www.youtube.com/watch?v=V1eYniJ0Rnk).

In this course, we will focus on supervised learning, though unsupervised learning often constitutes an integral step in training the supervised model. We have a whole course only on Reinforcement learning (http://www.ee.bgu.ac.il/~haimp/RL/index.html).

D. Types of models

Generative model: The learning model is probabilistic and it models the joint distribution of the samples and the labels, i.e., $P(x, l)$. It is called generative since often one can use it in order to generate data similar to the one it learns from. An example of generative model is the Gaussian Mixture Model that we learn later in the course.

<u>Discriminative model</u>: The learning model learns to discriminate the feature x into labels l, namely, it learns a mapping $\phi : x \mapsto l$, or the conditional probability $P(l|x)$ but not the joint as in the generative model. Discriminative model are using for supervised learning and very rarely for unsupervised learning. Examples of discriminative models that we will learn in the course are Logistic regression model and Neural network model.

E. Generative AI

Currently, the biggest success is Generative AI where we actually generate new data based on some input, like text or sound or image. This task uses generative models and it train them in a sophisticated way that does not require additional labels. For instance, in text it builds a model that predict the next word and it generate text by predicting each time the next word. In images it build a model that de-noise images where noise was added till the point that it generate images from pure noise.

II. Notation

Throughout the course we will use the following notation:

- X - random variable
- $\mathcal{X}$ - alphabet of X. The alphabet of X is the set of all possible outcomes. For instance, if X is a binary random variable then $\mathcal{X} = \{0, 1\}$. We denote sets by calligraphic letters, such as $\mathcal{A}, \mathcal{B}, ...$
- x - an observation or a specific value. Clearly, $x \in \mathcal{X}$.
- $P_X(x)$ - the probability that the random variable X gets the value x, i.e., $P_X(x) = \Pr\{X = x\}$.
- P_X or $P_X(\cdot)$- denotes the whole vector of probabilities, also known as probability mass function (pmf).
- $P(x)$ - this is a short notation for $P_X(x)$.
- $\mathbb{E}[X]$ - expectation, i.e.,

$$\mathbb{E}[X] \triangleq \sum_{x \in \mathcal{X}} x P(x) \tag{1}$$

Similarly $\mathbb{E}[g(X)]$ is

$$\mathbb{E}[g(X)] \triangleq \sum_{x \in \mathcal{X}} g(x)P(x) \tag{2}$$

III. Problem settings and criteria decision (MAP, MLE) when probability is known

Let $\{\eta_1, \eta_2, \ldots, \eta_M\}$, $\eta_i > 0 \ \forall i$, $\sum \eta_i = 1$ be the prior probability of the M classes, and $f_i(x)$ be the probability density of each class at x. The distribution of X is thus:

$$\begin{array}{rcc} & \textbf{Class} & \boldsymbol{P(x|class)} \\ X \sim & \eta_1 = P(Class = 1) & f_1(x) = f(x|Class = 1) \\ & \eta_2 = P(Class = 2) & f_2(x) = f(x|Class = 2) \\ & \vdots & \vdots \\ & \eta_M = P(Class = M) & f_M(x) = f(x|Class = M). \end{array}$$

Definition 1 (Loss Function) Let X be a random variable with the classes set $\{1, 2, \ldots, n\}$. The *loss function* $L(i, j)$ is the loss incurred by associating the observation to class j when in fact it belongs to class i ,$\forall i, j \in \{1, 2, \ldots, n\}$.

Example 1 (Right/wrong loss function) Consider M=2, and the loss function is the right/wrong function, such that a correct association yields no loss and an incorrect association, or an error, yields a loss of 1. The *loss matrix* in this case is:

$$L = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

For the case of right/wrong loss function, we notice that in the case of an error, the loss always counts as 1, while a correct association counts as 0 loss. The *loss matrix* is therefore the 0-1 matrix of $M \times M$ size. For example, consider the right/wrong case were $M = 3$, then

$$L = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}.$$

In general (not right/wrong case), a loss matrix can also be asymmetric, for instance, $L(1,2) > L(2,1)$. In this case loss matrix is warranted when the system is tasked with deciding whether a given person is a terrorist based on a recorded phone call. Failure to identify a terrorist may result in much greater loss than falsely deciding that an innocent person is a terrorist. Therefore, we can expect the representative loss matrix to be in many cases asymmetric.

Let us recall the joint probability equation

$$P(a,b) = P(a) \cdot P(b|a). \tag{3}$$

We can describe $P(x)$ as the sum of the joint probability over all the alphabet of X, and using eq. (3) we get

$$P(x) = \sum_{j=1}^{M} P(j,x) = \sum_{j=1}^{M} P(j)P(x|j) = \sum_{j=1}^{M} \eta_j f_j(x). \tag{4}$$

The probability that X belongs to class $i, \forall i \in \{1, 2, \ldots, M\}$ given the samples x, is the posterior probability $\hat{\eta_i}(x)$:

$$\hat{\eta_i}(x) = P(class = i|x) = \frac{P(class = i, x)}{P(x)} = \frac{P(i)P(x|i)}{P(x)} = \frac{\eta_i f_i(x)}{\sum_{j=1}^{M} \eta_j f_j(x)}, \tag{5}$$

We can finally sort the class probabilities in vectors. These include the prior probability vector:

$$P(class) = \left[\eta_1, \eta_2, \ldots, \eta_M\right], \tag{6}$$

and the Posterior probability vector:

$$P(class|x) = \left[\frac{\eta_1 f_1(x)}{\sum_{j=1}^{M} \eta_j f_j(x)}, \frac{\eta_2 f_2(x)}{\sum_{j=1}^{M} \eta_j f_j(x)}, \ldots, \frac{\eta_M f_M(x)}{\sum_{j=1}^{M} \eta_j f_j(x)}\right]. \tag{7}$$

Definition 2 (Conditional loss) The conditional loss denoted by $r_j(x)$ is the loss incurred by associating observation x with class j, then:

$$r_j(x) = \mathbb{E}\left[L(I,j) \mid X = x\right] = \sum_{i} P(class = i|x)L(i,j) = \sum_{i=1}^{M} \hat{\eta_i}(x)L(i,j). \tag{8}$$

Because our goal is to minimize the conditional loss, we therefore define $r^*(x)$ and R^* that is the one that corresponds for the minimum choice of class $j \in \{1, 2, \ldots, N\}$.

Definition 3 (Conditional Bayes risk) The conditional Bayes risk denoted by $r^*(x)$, is the loss incurred by associating x with class j that has the lowest cost out of all classes, i.e.

$$r^*(x) \triangleq \min_j \{r_j(x)\} = \min_j \left\{ \sum_{i=1}^{M} \hat{\eta}_i(x) L(i,j) \right\}. \tag{9}$$

Definition 4 (Bayes risk) The Bayes risk denoted by R^* is the resulting overall minimum expected risk, i.e,

$$R^* \triangleq \mathbb{E}\left[r^*(X)\right], \tag{10}$$

where the expectation is described in terms of the compound density function

$$f(x) = \sum_i \eta_i f_i(x). \tag{11}$$

Example 2 (Decision due to minimum loss) Consider the next loss matrix L, where all failures have the same loss value:

$$L = \begin{bmatrix} 0 & 1 & 1 & \ldots & 1 \\ 1 & 0 & 1 & \ldots & 1 \\ 1 & 1 & 0 & \ldots & 1 \\ \vdots & & & \ddots & \vdots \\ 1 & 1 & \ldots & 1 & 0 \end{bmatrix}.$$

Then for each class $j \in \{1, 2, \ldots, N\}$ the conditional loss is

$$r_j(x) = \sum_i \hat{\eta}_i(x) \cdot (1 - \delta(i,j)) = 1 - \hat{\eta}_j(x), \tag{12}$$

and therefore, by choosing the class that producing the minimum loss, we will minimize the Bayes risk, i.e.,

$$j = \text{argmax}\{\hat{\eta}_j(x)\} = \text{argmax}\{P(j|x)\}. \tag{13}$$

Minimizing the Bayes risk for the loss function of right/wrong yields the decision rule given in (13), which is also known as the Maximum a Posteriori (MAP) rule and is formally defined in the next definition.

Definition 5 (MAP) Maximum a Posteriori is the estimation method for a random variable J given observations $X = x$. The MAP estimator denoted by j^* is chosen according to the maximum value of the posterior probability function $P_{J|X}(j|x)$ i.e.,

$$j^* = \underset{j}{\operatorname{argmax}}\, P(j|x) = \underset{j}{\operatorname{argmax}}\, \eta_j f_j(x). \tag{14}$$

In our case, the posterior probability function is the vector

$$P_{J|X} = \left[\frac{\eta_1 f_1(x)}{\sum_{j=1}^{M} \eta_j f_j(x)}, \frac{\eta_2 f_2(x)}{\sum_{j=1}^{M} \eta_j f_j(x)}, \dots, \frac{\eta_M f_M(x)}{\sum_{j=1}^{M} \eta_j f_j(x)} \right]. \tag{15}$$

Given that the divisors are equal for all the elements, the maximum vector element is equal to $\operatorname{argmax}_j \eta_j f_j(x)$.

Now, for the case in which $\eta_1 = \eta_2 = \cdots = \eta_M$, meaning that all classes have the same prior probability, using the MAP method is similar to choosing the maximum of only $f_j(x)$, i.e.,

$$\operatorname{argmax} P(j|x) = \underset{j}{\operatorname{argmax}}\, f_j(x) = \underset{j}{\operatorname{argmax}}\, P(x|j), \tag{16}$$

and we can refer to another decision method, Maximum Likelihood Estimation (MLE).

Definition 6 (MLE) Maximum Likelihood Estimation is the method of estimating a random variable J due to observations $X = x$, by choosing estimator j^* to be the element with maximum conditional probability density function $f_j(x)$ at point x, i.e.

$$j^* = \underset{j}{\operatorname{argmax}}\, P(x|j) = \underset{j}{\operatorname{argmax}}\, f_j(x). \tag{17}$$

Example 3 (Two Gaussian distributed classes) Given two classes distributed normally over two dimensions. Using MLE, we choose the class that has the higher probability density function value at point x. Using this method entails an obligated error probability, in this case presented by the area trapped under the two Gaussian graphs:

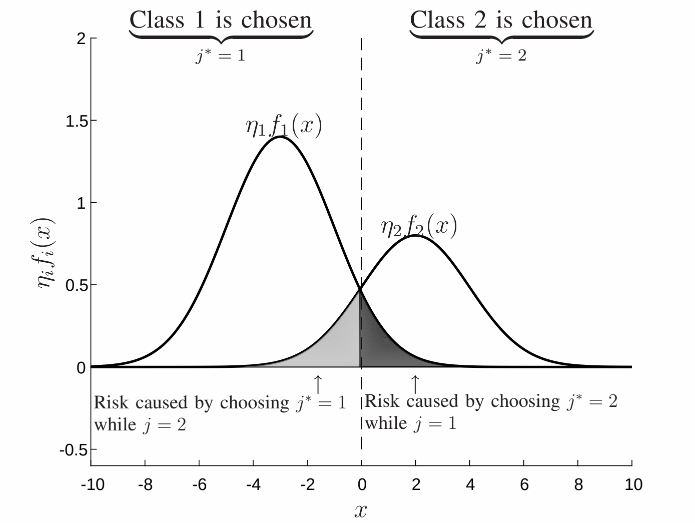

Figure 1. The total area trapped under the overlap between the two Gaussians ovelap (marked in color) is the total obligated error probability.

Now we can calculate the overall risk by integrating over $\min\{\hat{\eta_1}(x)f_1(x), \hat{\eta_2}(x)f_2(x)\}$ to obtain the trapped area:

$$
\begin{aligned}
R^* &= \mathbb{E}\left[r^*(X)\right] \\
&= \int r^*(x)f(x)\,dx \\
&= \int\limits_{\eta_1 f_1(x) > \eta_2 f_2(x)} \hat{\eta_2}(x)f(x)\,dx + \int\limits_{\eta_1 f_1(x) < \eta_2 f_2(x)} \hat{\eta_1}(x)f(x)\,dx \\
&= \int\limits_{\eta_1 f_1(x) > \eta_2 f_2(x)} \eta_2 f_2(x)\,dx + \int\limits_{\eta_1 f_1(x) < \eta_2 f_2(x)} \eta_1 f_1(x)\,dx.
\end{aligned}
\tag{18}
$$

Choosing best classifier when probability is unknown but we have samples via Empirical Risk Minimization (ERM)

In the previous section we defined the conditional Bayes risk $r^*(x)$ in Eq. (9) and the Bayes risk $R = E[r^*(X)]$ in (10), which are optimal but one need to know the probabilities of the classes i.e., η_i and the conditional probability $f_i(x)$ for all possible i and x. In many cases, we have several classifiers (or a set of classifiers) which can also be called hypothesis. The hypothesis set may not include the optimal one, namely, the

Bayes classifier that we saw in the previous subsection. The ERM idea is a very simple idea that tells us which hypothesis to choose from the set of hypotheses.

For each classifier/hypothesis h let's define a Risk

$$R(h) = E[L(h(X), Y)], \tag{19}$$

where $L(\cdot, \cdot)$ is the loss function (as defined in previous subsection in Def. 1) and y is the label associated with x. In general, we would like to choose the classifier/hypothesis h from the possible set $\mathcal{H}$ that minimize the risk, i.e.,

$$h^* = \min_{h \in \mathcal{H}} R(h). \tag{20}$$

However, in order to compute the risk for specific hypotheses h, i.e., $R(h)$, one need to know the joint probability $p(x, y)$ for all possible samples, x and labels, y. In practice, the joint probability $p(x, y)$, is unknown (this situation is called as *agnostic learning*). However, one can assume that we have samples and labels $\{x_i, y_i\}_{i=1}^n$ drawn from the joint $p(x, y)$. The set of samples that are available is called the training set. The *empirical risk* of a specific classifier h is defined as

$$R_{emp}^{(n)}(h) = \frac{1}{n} \sum_{i=1}^{n} L(h(x_i), Y_i) \tag{21}$$

and the *minimum empirical risk* idea is

$$\hat{h} = \min_{h \in \mathcal{H}} R_{emp}^{(n)}(h). \tag{22}$$

Assuming $\{x_i, y_i\}_{i=1}^n$ are i.i.d., (or at least stationary and ergodic) then by the law of large numbers $\lim_{n \to \infty} R_{emp}^{(n)}(h) = R(h)$ with probability 1, hence the minimum empirical risk convergence to the minimum risk if the number of samples is large enough.

Summary of the ERM idea: The ERM idea is very simple and extremely useful. We have a set of classifiers/hypothesis $\mathcal{H}$ and a set of samples (a.k.a tanning set). The ERM principle tells us to choose the classifier that minimize the emprical risk, i.e., Eq. (22).

IV. K-Nearest Neighbor Model

Definition 7 (KNN) K-Nearest Neighbor is a basic classification algorithm, that uses a predefined, classified finite training set and a defined distance function to estimate the class of a test element. Here is a description of the setting and the algorithm:

Training set: $(x_1, \theta_1)\ (x_2, \theta_2)\ \dots\ (x_n, \theta_n)$

Test: x

1-NN Algorithm: Find x' so that

$$d(x, x')\ \leq\ d(x, x_i)\ \forall\, i = 1, \dots, n$$

where θ' is the label of x'. The algorithm decides that the label of x is as the label of x', i.e., θ'. This algorithm can be extended to $k - NN$ by taking the label of the majority of k neighbours.

The prior probability to receive the element from class i is again η_i:

$$\begin{aligned}\eta_1 &= P(\theta = 1)\\ \eta_2 &= P(\theta = 2)\\ &\vdots\\ \eta_n &= P(\theta = n)\end{aligned}$$

The term $L(\theta, \theta'_n)$ is the right/wrong loss function.

Considering that $x \sim i.i.d \quad and \quad P(x|\theta = i) = f_i(x)$, we get the joint probability to be:

$$P(x, \theta, x_1, \theta_1, \dots, x_n, \theta_n) = P(\theta)P(x|\theta)P(\theta_1)P(x_1|\theta_2)P(\theta_2)P(x_2|\theta_2)\cdots \tag{23}$$

We define the n-sample Nearest Neighbor procedure risk to be (n holdss for the training number of samples) :

$$R(n) = \mathbb{E}\left[L(\theta, \theta'_n)\right] \tag{24}$$

And for a large number of training samples n, we define R to be the NN risk:

$$R = \lim_{n \to \infty} R(n) \tag{25}$$

Theorem 1 (Nearest neighbor risk bounds) Assume that there are two possible classes $\theta = 1, 2$,and that x with probability 1 is either a continuous point of f_1 or f_2 or a non-zero probability measure. The overall 1-NN risk R then has the bounds

$$R^* \leq R \leq 2R^* \cdot (1 - R^*), \tag{26}$$

where R^* is the minimum risk for right/wrong loss function, i.e., the Bayes risk which is obtained by MAP when f_1 and f_2 are known.

In the next part of this lecture, we will use a lemma and the definitions given above to show that these bounds hold. Before we get to the proof, note that the Bayes risk R^* can only get values in the section $\left[0, \frac{1}{2}\right]$, so that $0 \leq R^* \leq R \leq 2R^*(1 - R^*) \leq \frac{1}{2}$. In particular, for the edge cases, we get that $R^* = 0$ if and only if $R = 0$, and that $R^* = \frac{1}{2}$ if and only if $R = \frac{1}{2}$.

proof: Under the same assumptions on f_1, f_2 and x as in *Theorem 1*, the conditional NN risk $r(x, x'_n)$ is then given by:

$$\begin{aligned} r(x, x'_n) &= \mathbb{E}\left[L(\theta, \theta'_n)|x_n, x'_n\right] \\ &= P(\theta = 1, \theta'_n = 2|x, x'_n) + P(\theta = 2, \theta'_n = 1|x, x'_n) \\ &= P(\theta = 1|x) \cdot P(\theta'_n = 2|x'_n) + P(\theta = 2|x) \cdot P(\theta'_n = 1|x'_n)., \end{aligned} \tag{27}$$

where we use the conditional independence of θ and θ'_n to open the phrase.

Exercise 1 Prove the equation used above: $P(\theta, \theta'_n|x, x'_n) = P(\theta|x) \cdot P(\theta'_n|x'_n)$

Lemma 1 (Convergence of the nearest neighbor) : Under the assumption on f_1, f_2 that x with probability 1 is either a continuous point of f_1, f_2 or a non-zero probability measure. x'_n denotes the nearest neighbor to x within the set $\{x_1, x_2, \ldots, x_n\}$. Then we get $\lim_{n\to\infty} x'_n = x$, convergence of the nearest neighbor to x with high probability.

proof: Let $S_x(r),\ r > 0$ be the sphere of radius r centered at x, and $d(\cdot)$ is the metric defined on X. Considering the case where $S_x(r),\ r > 0$ has a non-zero probability measure, then for any $\delta > 0$

$$P\{\min_{k=1,2,\ldots,n} d(x, x_k) \geq \delta\} = (1 - P(S_x(\delta)))^n \to 0. \tag{28}$$

The distance of the nearest neighbor x'_n from x decreases monotonically with the increase in k. ■

We can now use the fact that $\lim_{n\to\infty} x'_n = x$ with probability one to show that the conditional NN risk $r(x, x'_n)$ converges to the limit $2r^*(x)(1-r^*(x))$. For large numbers of training samples n, we get

$$\lim_{n\to\infty} r(x, x'_n) \overset{(a)}{=} \lim_{n\to\infty} (\hat{\eta}_1(x)\hat{\eta}_2(x'_n) + \hat{\eta}_2(x)\hat{\eta}_1(x'_n)) \tag{29a}$$

$$\overset{(b)}{=} 2 \cdot \hat{\eta}_1(x) \cdot \hat{\eta}_2(x) \tag{29b}$$

$$\overset{(c)}{=} 2\hat{\eta}_1(x)(1 - \hat{\eta}_1(x)) \tag{29c}$$

$$\overset{(d)}{=} 2r^*(x)(1 - r^*(x)), \tag{29d}$$

where:

(a) holds using equation (27).

(b) holds since x'_n converge to x.

(c) holds since $\hat{\eta}_1(x), \hat{\eta}_2(x)$ are symmetric and $\hat{\eta}_2(x) = 1 - \hat{\eta}_1(x)$.

(d) holds since $r^*(x) = \min(\hat{\eta}_1(x), \hat{\eta}_2(x)) = \min(\hat{\eta}_1(x), 1 - \hat{\eta}_1(x))$.

Recalling equation (8) and by the total expectation law, we get

$$\mathbb{E}\left[r(x, x'_n)\right] = \mathbb{E}\left[\mathbb{E}\left[L(\theta, \theta'_n)|x, x'_n\right]\right] = \mathbb{E}\left[L(\theta, \theta'_n)\right]. \tag{30}$$

Now R is the limit of the expectation of $r(x, x'_n)$, and we can use the fact that $r(x, x'_n)$ is bounded to switch the order of expectation and the limit to get

$$R = \lim_{n\to\infty} \mathbb{E}\left[r(x, x'_n)\right] \overset{r(x,x'_n)<1}{=} \mathbb{E}\left[\lim_{n\to\infty} r(x, x'_n)\right]. \tag{31}$$

From the dominant convergence theorem we get that

$$R = \mathbb{E}\left[2\eta_1(x)\eta_2(x)\right] = \mathbb{E}\left[2r^*(x)(1 - r^*(x))\right], \tag{32}$$

and now we can write equation (31) as follows:

$$R = \mathbb{E}[r(x)] \tag{33a}$$

$$= \mathbb{E}[2r^*(x)(1 - r^*(x))] \tag{33b}$$

$$= \mathbb{E}[r^*(x) + r^*(x)(1 - 2r^*(x))] \tag{33c}$$

$$\overset{(a)}{=} R^* + \mathbb{E}[r^*(x)(1 - 2r^*(x))] \tag{33d}$$

$$\overset{(b)}{\geq} R^*, \tag{33e}$$

where:

(a) holds for the linearity of the expectation.

(b) holds since $r^* \in [0, \frac{1}{2}]$ and the expression inside the expectation is non negative over this section, and equality is achieved only if $r^*(x)(1 - 2r^*(x)) = 0$.

Using the first part of the equation above and the fact that R^* is the expectation of r^*, and that $Var(r^*(x)) \geq 0$ we can then write

$$\begin{aligned} R &= \mathbb{E}[2r^*(x)(1 - r^*(x))] \\ &= 2R^*(1 - R^*) - 2Var(r^*(x)) \\ &\leq 2R^*(1 - R^*), \end{aligned} \tag{34}$$

and

$$R^* = \mathbb{E}\left[r^*(x)\right] \tag{35a}$$

$$\overset{(a)}{\leq} \mathbb{E}\left[2r^*(x)(1 - r^*(x))\right] \tag{35b}$$

$$\overset{(b)}{\leq} 2R^*(1 - R^*), \tag{35c}$$

where:

(a) holds from equation (33e).

(b) holds since $r^*(x) \leq \frac{1}{2}$.

To conclude, we collect equations (33),(34) to obtain the bounds of the overall NN risk $R(n)$:

$$R^* \leq R \leq 2R^*(1 - R^*). \tag{36}$$

■

References

[1] T.M. Cover and P.E. Hart. *Nearest Neighbor Pattern Classification*. IEEE, 1967.