Lecture-16: Nearest neighbors

1 Introduction

The idea of nearest neighbor algorithms is to predict the label of a new example based on the labels of closest neighbors in the training set.

Assumption 1.1. Close points in feature space have the same label.

Remark 1. Nearest neighbor search algorithms are fast even for a very large training set.

2 k-nearest neighbors

Assumption 2.1. Input space $\mathcal{X}$ is a normed space with distance between two examples $x, x' \in \mathcal{X}$ is defined as $\rho(x,x') \triangleq \|x - x'\|$.

Let $S \in (\mathcal{X} \times \mathcal{Y})^m$ be an m-length sequence of training sample. For each test example $x \in \mathcal{X}$, let $\pi^x : [m] \to [m]$ be a permutation of training example indices in non-decreasing order of distance from x. That is, for $i \in [m-1]$, we define $\rho_i \triangleq \rho(x, x_i)$ and

$$\rho_{\pi_i^x} = \rho(x, x_{\pi_i^x}) \leqslant \rho(x, x_{\pi_{i+1}^x}) = \rho_{\pi_{i+1}^x}.$$

Algorithm 1 k-nearest neighbor algorithm for binary classification $\mathcal{Y} = \{-1, 1\}$

```
procedure kNearestNeighbor(test sample $S = ((x_1, y_1), \dots, (x_m, y_m))$, test example $x$)
    Find the permutation $\pi^x : [m] \to [m]$ such that $(\rho_{\pi_i^x} : i \in [m])$ is non-decreasing
    Find the majority in $h_S(x) \triangleq \left\{ y_{\pi_i^x} : i \in [k] \right\}$
    return $h_S(x)$
```

Remark 2. For 1-nearest neighbor, we have $h_S(x) = y_{\pi_1^x}$.

Definition 2.2 (Regression output). When $\mathcal{Y} = \mathbb{R}$, one can define the prediction $h_S : \mathcal{X} \to \mathcal{Y}$ to be the average target of the k nearest neighbors. That is, $h_S(x) = \sum_{i=1}^{k} y_{\pi_i^x}$.

Definition 2.3 (General k-nearest neighbor). When $\mathcal{Y} = \mathbb{R}$, we can define a function $\phi : (\mathcal{X} \times \mathcal{Y})^k \to \mathcal{Y}$ such that the prediction $h_S : \mathcal{X} \to \mathcal{Y}$ is defined as $h_S(x) = \phi((x_{\pi_i^x}, y_{\pi_i^x}) : i \in [k])$.

Example 2.4. Let $z \in (\mathcal{X} \times \mathcal{Y})^k$. For binary classification $\phi(z) = \text{sign}(\sum_{i=1}^{k} y_i)$. For regression, $\phi(z) = \frac{1}{k}\sum_{i=1}^{k} y_i$. For generative models, $\phi(z) = \sum_{i=1}^{k} \frac{\rho(x,x_i)}{\sum_{j=1}^{k}\rho(x,x_j)} y_i$.

3 Finite sample analysis

3.1 Generalization bound for the 1-NN rule

Consider 1-NN rule for binary classification such that $\mathcal{Y} = \{0, 1\}$, loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \{0, 1\}$ defined as $\ell(h(x), y) = \mathbb{1}_{\{h(x) \neq y\}}$, input space $\mathcal{X} = [0,1]^d$ equipped with Euclidean norm $\|x\| \triangleq (\sum_{i=1}^{d} x_i^2)^{\frac{1}{2}}$ for any $x \in \mathcal{X}$. Let $\mathcal{D}$ be the distribution over $\mathcal{X} \times \mathcal{Y}$ and $\mathcal{D}_{\mathcal{X}}$ the induced marginal over input space $\mathcal{X}$, and $\eta : \mathbb{R}^d \to \mathbb{R}$ be the conditional probability over the labels. That is,

$$\eta(x) = P(\{Y = 1\} \mid \{X = x\}) = \lim_{h \to 0} \frac{\mathcal{D}((x',1) : x' \in B(x,h))}{\mathcal{D}((x',y') : x' \in B(x,h), y \in \mathcal{Y})}.$$

Recall that the Bayes optimal rule $h_B(x) = \arg\min_y L_{\mathcal{D}}(h) = \arg\min_y \mathbb{E}_{\mathcal{D}} \ell(h(X),Y)$, and is given by

$$h_B(x) = \mathbb{1}_{\{\eta(x) > \frac{1}{2}\}}.$$

Assumption 3.1. Conditional probability function η is c-Lipschitz for some $c > 0$. That is,

$$|\eta(x) - \eta(x')| \leqslant c \left\| x - x' \right\|.$$

Remark 3. This is technical assumption that ensures that if feature vectors are close, then their labels are likely to be close.

Lemma 3.2. *Let $\mathcal{X} = [0,1]^d, \mathcal{Y} = \{0,1\}$, and $\mathcal{D}$ be a distribution over $\mathcal{X} \times \mathcal{Y}$ for which the conditional probability function η is a c-Lipschitz function. Let $S \in (\mathcal{X} \times \mathcal{Y})^m$ be an* i.i.d. *sample and let h_S be its corresponding* 1*-NN hypothesis. Let h_B be the Bayes optimal rule for η. Then,*

$$\mathbb{E}[L_{\mathcal{D}}(h_S)] \leqslant 2L_{\mathcal{D}}(h_B) + c\mathbb{E}\left\| x - x_{\pi_1^x} \right\|.$$

Proof. Since $L_{\mathcal{D}}(h_S) = \mathbb{E}_{\mathcal{D}} \mathbb{1}_{\{h_S(x) \neq y\}}$, we obtain that $\mathbb{E}_S L_{\mathcal{D}}(h_S)$ is the probability to sample a training set S and an additional example (x,y), such that $y_{\pi_1^x} \neq y$. In other words, we can first sample m unlabeled examples $S_{\mathcal{X}} = (x_1, \dots, x_m)$ and an additional unlabeled example x, all *i.i.d.* according to $\mathcal{D}_{\mathcal{X}}$. Then find π_1^x to be the nearest neighbor of x in $S_{\mathcal{X}}$, and finally sample y according to $\eta(x)$ and $y_{\pi_1^x}$ according to $\eta(x_{\pi_1^x})$. It follows that

$$\mathbb{E}[L_{\mathcal{D}}(h_S)] = \mathbb{E}\mathbb{1}_{\left\{y_{\pi_1^x} \neq y'\right\}} = \mathbb{E}_{S_{\mathcal{X}} \sim \mathcal{D}_{\mathcal{X}}^m, x \sim \mathcal{D}_{\mathcal{X}}}[P_{y \sim \eta(\pi_1^x), y' \sim \eta(x)} \left\{y \neq y'\right\}].$$

We can write the inner probability as

$$P_{y \sim \eta(x), y' \sim \eta(x')} \left\{y \neq y'\right\} = \eta(x')(1 - \eta(x)) + (1 - \eta(x'))\eta(x) = 2\eta(x)(1 - \eta(x)) + (\eta(x) - \eta(x'))(2\eta(x) - 1).$$

Since $|2\eta(x) - 1| \leqslant 1$ and η is c-Lipschitz, we obtain an upper bound

$$P_{y \sim \eta(x), y' \sim \eta(x')} \left\{y \neq y'\right\} \leqslant 2\eta(x)(1 - \eta(x)) + c \left\| x - x' \right\|.$$

We also observe that $L_{\mathcal{D}}(h_B) \geqslant \mathbb{E}[\eta(x) \wedge (1 - \eta(x))] \geqslant \mathbb{E}[\eta(x)(1 - \eta(x))]$. This concludes the proof. □

Lemma 3.3. *Consider a collection $\{C_i \subseteq \mathcal{X} : i \in [r]\}$. Let $S \in \mathcal{X}^m$ be a sequence of m points sampled* i.i.d. *according to some probability distribution $\mathcal{D}$ over $\mathcal{X}$. Then,*

$$\mathbb{E}[\sum_{i : C_i \cap S = \emptyset} P(C_i)] \leqslant \frac{r}{me}.$$

Proof. From the linearity of expectation, we obtain $\mathbb{E}[\sum_{i : C_i \cap S = \emptyset} P(C_i)] = \sum_{i=1}^r P(C_i) \mathbb{E}\mathbb{1}_{\{C_i \cap S = \emptyset\}}$. Since each example is distributed *i.i.d.* , we can write $P\{C_i \cap S = \emptyset\} = \prod_{j=1}^m P\left\{x_j \notin C_i\right\} = (1 - P(C_i))^m \leqslant e^{-mP(C_i)}$. Combining the two results, we obtain

$$\mathbb{E}[\sum_{i : C_i \cap S = \emptyset} P(C_i)] \leqslant \sum_{i=1}^r P(C_i) e^{-mP(C_i)} \leqslant r \max_{i \in [r]} P(C_i) e^{-mP(C_i)}.$$

The result follows from the fact that $\max_a ae^{-ma} \leqslant \frac{1}{me}$. □

Theorem 3.4. *Let $\mathcal{X} = [0,1]^d, \mathcal{Y} = \{0,1\}$, and $\mathcal{D}$ be a distribution over $\mathcal{X} \times \mathcal{Y}$ for which the conditional probability function η is a c-Lipschitz function. Let h_S denote the result of applying the* 1*-NN rule to an* i.i.d. *random sample $S \in (\mathcal{X} \times \mathcal{Y})^m$ with distribution $\mathcal{D}$. Then,*

$$\mathbb{E}L_{\mathcal{D}}(h_S) \leqslant 2L_{\mathcal{D}}(h_B) + 4c\sqrt{d}m^{-\frac{1}{d+1}}.$$

Proof. Fix $T \in \mathbb{N}$ and define $\epsilon \triangleq 1/T$ and $r \triangleq T^d$. Let $(C_1,\dots,C_r)$ be the cover of the set $\mathcal{X}$ using d-dimensional boxes of length ϵ. Then for every $\alpha \in [T]^d$, there exists a set C_i of the form

$$\cap_{j=1}^{d} \left\{ x \in [0,1]^d : x_j \in \left[\frac{(\alpha_j - 1)}{T}, \frac{\alpha_j}{T} \right] \right\}.$$

Each point $x \in \mathcal{X}$ falls in one of the sets C_i. Therefore, for each $x, x' \in \mathcal{X}$, we have

$$\left\| x - x' \right\| \leqslant \begin{cases} \sqrt{d}\epsilon, & \text{if } x, x' \text{ in the same box,} \\ \sqrt{d}, & \text{else.} \end{cases}$$

The test point $x \in C_i$ for some $i \in [r]$. It is possible that none of the training points S intersect with that set or at least one does. If none of the training points intersect with C_i, then the distance $\left\| x - x_{\pi_1^x} \right\| \leqslant \sqrt{d}$. If at least one of the training points intersect with C_i, then the distance from the closest point is $\left\| x - x_{\pi_1^x} \right\| \leqslant \sqrt{d}\epsilon$. Therefore, given a sample S, we can write

$$\left\| x - x_{\pi_1^x} \right\| \leqslant \sqrt{d} P(\cup_{i:C_i \cap S = \emptyset} C_i) + \epsilon \sqrt{d} P(\cup_{i:C_i \cap S \neq \emptyset} C_i).$$

Taking the mean on both sides, linearity of expectation, union bound, and the fact that $P(\cup_{i:C_i \cap S \neq \emptyset} C_i) \leqslant 1$, we obtain

$$\mathbb{E} \left\| x - x_{\pi_1^x} \right\| \leqslant \sqrt{d} \left[\frac{r}{me} + \epsilon \right] = \sqrt{d} \left[\frac{1}{me\epsilon^d} + \epsilon \right].$$

Taking $\epsilon = m^{-\frac{1}{d+1}}$, we obtain that $\mathbb{E} \left\| x - x_{\pi_1^x} \right\| \leqslant \sqrt{d} m^{-\frac{1}{d+1}} \left[\frac{1}{e} + 1 \right] \leqslant 2\sqrt{d} m^{-\frac{1}{d+1}}$. The result follows from substituting this upper bound in right hand side of Lemma 3.2. □

Remark 4. The theorem implies that if we first fix the data-generating distribution and then let m go to infinity, then the error of the 1-NN rule converges to twice the Bayes error. The analysis can be generalized to larger values of k, showing that the expected error of the k-NN rule converges to $(1 + \sqrt{\frac{8}{k}})$ times the error of the Bayes classifier.

3.2 Curse of dimensionality

Remark 5. The generalization upper bound on the performance of 1-NN grows with the Lipschitz coefficient c of η with the Euclidean dimension d of the input space $\mathcal{X}$. We observe that the term $4c\sqrt{d} m^{-\frac{1}{d+1}} \leqslant \epsilon$ if $m \geqslant (4c\frac{\sqrt{d}}{\epsilon})^{d+1}$. That is, the size of the training set should increase exponentially with the dimension.

Theorem 3.5. *For any $c > 1$, and every learning rule L, there exists a distribution over $[0,1]^d \times \{0,1\}$, such that $\eta(x)$ is c-Lipschitz, the Bayes error of the distribution is 0, but for sample sizes $m \leqslant \frac{1}{2}(c+1)^d$, the true error of the rule L is greater than $\frac{1}{4}$.*

Proof. Fix any values of c and d. Let G_c^d be the grid on $[0,1]^d$ with distance of $\frac{1}{c}$ between points on the grid. That is, each point on the grid is of the form $(\frac{a_1}{c},\dots,\frac{a_d}{c})$ where $a_i \in \{0,\dots,c-1,c\}$. Note that, since any two distinct points on this grid are at least $\frac{1}{c}$ apart, any function $\eta : G_c^d \to [0,1]$ is a c-Lipschitz function. It follows that the set of all c-Lipschitz functions over G_c^d contains the set of all binary valued functions over that domain. We can therefore invoke the No-Free-Lunch result to obtain a lower bound on the needed sample sizes for learning that class. The number of points on the grid is $(c+1)^d$. Hence, if $m < \frac{1}{2}(c+1)^d$, Theorem 5.1 implies the lower bound we are after. □

Remark 6. The exponential dependence on the dimension is known as the *curse of dimensionality*.

Theorem 3.6 (No-Free-Lunch). *Let A be any learning algorithm for the task of binary classification with respect to the* $0-1$ *loss over an input space $\mathcal{X}$. Let the training set size $m \leqslant \frac{|\mathcal{X}|}{2}$. Then, there exists a distribution $\mathcal{D}$ over $\mathcal{X} \times \{0,1\}$ such that*

1. *There exists a function $f : \mathcal{X} \to \{0,1\}$ with $L_{\mathcal{D}}(f) = 0$.*
2. *With probability of at least $\frac{1}{7}$ over the choice of* i.i.d. *sample S, we have that $L_{\mathcal{D}}(A(S)) \geqslant \frac{1}{8}$.*

Proof. Let $C \subseteq \mathcal{X}$ such that $|C| = 2m$. The intuition of the proof is that any learning algorithm that observes only half of the instances in C has no information on what should be the labels of the rest of the instances in C. Therefore, there exists a 'reality,' that is, some target function f, that would contradict the labels that $A(S)$ predicts on the unobserved instances in C. □

4 Nonparametric regression

Let $S \in (\mathcal{X} \times \mathcal{Y})^m$ be *i.i.d.* with a fixed but unknown distribution $\mathcal{D}$ over space $\mathcal{X} \times \mathcal{Y}$. We assume a model $y = m_0(x) + \epsilon$ where the function m_0 is unknown x, ϵ are independent random variables, and distribution of noise ϵ is unknown. We are interested in estimating function m_0, and denote its estimate by $\hat{m}$.

Definition 4.1 (Mean square error). The performance measure of estimate $\hat{m}$ is the mean square error $\mathbb{E}(Y - \hat{m}(X))^2$. The mean square error estimate is the one that minimizes this error.

Remark 7. If the noise $\epsilon = Y - m_0(X)$ is zero mean, then we can write the mean square error in terms of variance of noise, bias $m_0(X) - \mathbb{E}\hat{m}(X)$, and variance $\operatorname{Var}\hat{m}(X)$, as

$$\begin{aligned}\mathbb{E}(Y - \hat{m}(X))^2 &= \mathbb{E}(Y - m_0(X) + m_0(X) - \mathbb{E}\hat{m}(X) + \mathbb{E}\hat{m}(X) - \hat{m}(X))^2 \\ &= \mathbb{E}\epsilon^2 + \mathbb{E}(m_0(X) - \mathbb{E}\hat{m}(X))^2 + \operatorname{Var}\hat{m}(X).\end{aligned}$$

The first term is variance of the noise, and that can't be reduced. Second term is the mean of bias squared and can be reduced by choosing from a larger class of functions. However, this increases variance $\operatorname{Var}\hat{m}(X)$. This is bias variance tradeoff.

Remark 8. Recall that the generalization risk for an ERM hypothesis $h_S \in \mathcal{H}$ can be written as

$$R(h_S) = (R(h_S) - R(h^*)) + (R(h^*) - R^*) + R^*,$$

where $R(h_S) - R(h^*)$ is the estimation error in the class $\mathcal{H}$, $R(h^*) - R^*$ is the approximation error of the class $\mathcal{H}$, and R^* is the Bayes error. Recall $R(h_S^{\text{ERM}}) - R(h^*) \leqslant 2\sup_{h \in \mathcal{H}}(R(h) - R(\hat{h}))$ which is upper bounded by the Rademacher complexity of class $\mathcal{H}$.

4.1 k-NN regression

We assume $S \in (\mathcal{X} \times \mathcal{Y})^m$ *i.i.d.* sample with $y = m_0(x) + \epsilon$ where the mean of the noise $\mathbb{E}\epsilon = 0$, and $\mathcal{X} \subseteq \mathbb{R}^d$ is equipped with a Euclidean norm. The goal is to estimate $m_0(x)$ at the test point $x \in \mathcal{X}$. Find k-nearest neighbors of x in the sample $S_{\mathcal{X}}$, and call them $(x_{\pi_1^x}, \dots, x_{\pi_k^x})$ as before. Then, $\hat{m}(x) \triangleq \frac{1}{k}\sum_{i=1}^k y_{\pi_i^x}$. If $k = n$, then $\hat{m}(x) = \frac{1}{n}\sum_{i=1}^n y_i$ is the sample mean, same for all test points x.

Remark 9. For a good estimate of $m_0(x)$, the number of nearest neighbors k should be large to reduce the effect of noise. However, k should be not too large as nearest neighbors should be not too far from x. This is due to the fact that if $\|x - x'\|$ is large then so is $\|m_0(x') - m_0(x)\|$, even for a continuous function m_0. If m_0 is discontinuous, then this estimator can be bad in the neighborhood of the discontinuity.

Theorem 4.2 (Universal consistency). *Consider a sequence $k : \mathbb{N} \to \mathbb{N}$ such that $\lim_{m\to\infty} k_m = \infty$ and $\lim_{m\to\infty} \frac{k_m}{m} = 0$. If $\mathbb{E}Y^2 < \infty$, then for an m-size sample estimate $\hat{m}_m$, we have* $\lim_{m\to\infty} \mathbb{E}(\hat{m}_m(x) - m_0(x))^2 = 0$.

Theorem 4.3. *If m_0 is ℓ-Lipschitz for a finite ℓ, input space $\mathcal{X} \subseteq \mathbb{R}^d$, and $k_m = o(m^{-\frac{2}{2+d}})$, then $\mathbb{E}(\hat{m}_m(x) - m_0(x))^2 = o(m^{-\frac{2}{2+d}})$.*

Remark 10. If $\mathbb{E}(\hat{m}_m(x) - m_0(x))^2 = \delta$, then the sample complexity $m \approx O(\delta^{\frac{2+d}{2}})$. That is, the number of samples needed for a given mean squared error increases exponentially with the number of dimensions. This is called the *curse of dimensionality*. Most estimators suffer from this curse.

Remark 11. The asymptotic rate of convergence given above, is optimal.

Remark 12. One can find k-nearest neighbors efficiently in sub-linear time, and hence it is a preferred algorithm.

4.2 Random Forest

Sample $S \in (\mathcal{X} \times \mathcal{Y})^m$ *i.i.d.* from a distribution $\mathcal{D}$ to form L decision trees. A test input x is passed through all the decision trees in the forest, and one can estimate $m_0(x)$ by taking average of the outputs of the L decision trees in the forest. The ith decision tree output label $y_i = \hat{m}_i(x)$ and the overall estimate is

$$\hat{m}(x) = \frac{1}{L}\sum_{i=1}^L y_i.$$

4.2.1 Learning the tree

Learning for regression trees can be done similar to decision trees with the following change for impurity measure.

Remark 13 (Classification impurity). Recall that in the greedy method for finding a classifier decision tree, the next node $\mathbf{n}_i$ and question $\mathbf{q}_i = x_i < t_i$ are selected as the one which minimizes the impurity. The classification impurity is measured in terms of Entropy, Gini index, or misclassification probability.

Remark 14 (Regression impurity). If $(y_1, \dots, y_j)$ are the outputs at a leaf, then the estimator is given by $\hat{m}(x) = \frac{1}{j}\sum_{i=1}^{j} y_i$. For a sample $S \in (\mathcal{X} \times \mathcal{Y})^m$, the mean square error is estimated as

$$\sum_{x_i \text{ that reach the leaf}} (y_k - \frac{1}{j}\sum_{i=1}^{j} y_i)^2$$

This mean square estimate is the impurity measure used at a node, in the case of regression.

4.2.2 Random forest

We get L trees $T_1, \dots, T_L$ where each tree is obtained in the following fashion, similar to how decision tree classifiers are obtained.

1. Take a random subsample S' from the sample S
2. Take a random subset of features from $[d]$
3. From subsample S' and subset of features get tree T_1
4. Repeat the process L times to get trees $T_1, \dots, T_L$

Theorem 4.4. *Let each tree be formed from a_m subsamples of m-sized sample S, and let k_m be the number of training samples in each leaf. To make the trees, we select a feature at each time randomly with probability $\frac{1}{d}$. Let $\lim_{m\to\infty} k_m = \infty$ and $\lim_{m\to\infty} \frac{k_m}{m} = 0$. Then, the mean square error $\lim_{m\to\infty} \mathbb{E}(\hat{m}_m(x) - m_0(x))^2 = 0$ if $\mathbb{E}Y^2 < \infty$.*

Remark 15. In the above theorem on random forest, the trees are not formed using Greedy algorithm but each feature is selected randomly. Random forests work well in practice.

4.3 Kernel Smoothing

The kernel here is *not necessarily*, the RKHS kernels we have talked before.

Definition 4.5. We define kernel function $K : \mathbb{R}^d \to \mathbb{R}$ such that $\int_{x\in\mathbb{R}^d} K(x)dx = 1$, $\int_{x\in\mathbb{R}^d} xK(x)dx = 0$, and $0 < \int_{x\in\mathbb{R}^d} x^2K(x)dx < \infty$.

Remark 16. Any density function with zero mean and finite mean will work as a kernel function.

Example 4.6 (Gaussian kernel). For $d = 1$, the Gaussian density with zero mean and unit variance is defined as $K(x) \triangleq \frac{1}{\sqrt{2\pi}}e^{-\frac{x^2}{2}}$ for $x \in \mathbb{R}$.

Example 4.7 (Box kernel). For $d = 1$, the Box kernel with zero mean and finite variance is defined as $K(x) \triangleq \mathbb{1}_{[-0.5,0.5]}(x)$ for $x \in \mathbb{R}$.

Definition 4.8 (Nadraya-Watson Kernel regression estimator). For a positive bandwidth $h > 0$ and kernel $K : \mathbb{R}^d \to \mathbb{R}$, we define estimator

$$\hat{m}(x) \triangleq \frac{\sum_{i=1}^{m} K(\frac{x-x_i}{h})y_i}{\sum_{i=1}^{m} K(\frac{x-x_i}{h})}$$

Remark 17. This estimator can be thought of as generalization of k-nearest neighbor. When compared to k-nearest neighbors, we observe the following.

1. If K is a continuous function, then $\hat{m}(x)$ is a smooth estimator,

2. Different examples in the sample gets assigned different weights for $\hat{m}(x)$. Points closer to x should be given higher weights.
3. This estimator does suffer from a poor bias on the boundary of the sample.

Theorem 4.9 (Universal consistency). *Let $\mathbb{E}Y^2 < \infty$, kernel K is supported on a compact region, and bandwidth sequence $h : \mathbb{N} \to \mathbb{N}$ such that $\lim_{m\to\infty} h_m = 0$ and $\lim_{m\to\infty} mh_m = \infty$. Then, for an m-size sample estimate $\hat{m}_m$, we have $\lim_{m\to\infty} \mathbb{E}(\hat{m}_m(x) - m_0(x))^2 = 0$.*

Remark 18. Optimal sequence $h_m \approx m^{-\frac{1}{4+d}}$, then mean square error $\approx m^{-\frac{4}{4+d}}$.

4.3.1 Kernel Method RKHS

Given *i.i.d.* sample $S \in (\mathcal{X} \times \mathcal{Y})^m$, we wish to estimate $m_0(x)$ for a test example x. Given kernel function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$, a feature map $\Phi : \mathcal{X} \to \mathbb{H}$, and Hilbert space $\mathbb{H}$, optimal weight $\mathbf{w}^*$ is obtained as the solution to the optimization problem

$$\arg\min_{\mathbf{w}} \lambda \left\|\mathbf{w}\right\|_{\mathbb{H}}^2 + \sum_{i=1}^{m} (\langle \mathbf{w}, \Phi(x_i) \rangle - y_i)^2.$$

The estimator for test example x is $\langle \mathbf{w}^*, \Phi(x) \rangle$.