Appendix A

An analytic learning setup

In this appendix, we introduce an analytic setup for the learning problem that is both tractable and versatile. The setup uses a squared loss function and a linear learning model with non-linear transformations. Highly sophisticated target functions and learning models, as well as noise, can be handled under this setup, and we are able to get analytic solutions for the out-of-sample error in this framework. Various chapters of the thesis use the results of this section.

We begin by defining the notation. Let $R = \{x_i, y_i\}_{i=1}^N$ be the training set, with $x_i \in \mathcal{X}$, and $y_i \in \mathcal{Y}$. Assume x_i are iid $\sim P_R$, where P_R is the training distribution. Let the target function be $f : \mathcal{X} \to \mathcal{Y}$, and let ϵ be the stochastic noise process, where ϵ_i is the realization corresponding for x_i, so that $y_i = f(x_i) + \epsilon_i$. Let $\mathcal{H}$ be the hypothesis set used by the learning algorithm, where each $h \in \mathcal{H}$ is $h : \mathcal{X} \to \mathcal{Y}$. Finally we assume the learning algorithm returns a final hypothesis $g \in \mathcal{H}$ that minimizes the squared loss function $\ell_2 : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$, $\ell_2(h(x_i), y_i) = (h(x_i) - y_i)^2$. That is,

$$g = \arg\min_h \sum_{i=1}^{N} (y_i - h(x_i))^2. \tag{A.1}$$

We let $\mathcal{H}$ be the set of linear functions in some transformed space $\mathcal{Z}_m$ of the input space $\mathcal{X}$, so that

$$h(x; \theta) = \theta^T \phi_M(x). \tag{A.2}$$

where $\theta, \phi_M(x) \in \mathcal{Z}_M$. For simplicity we let

$$z_M = \phi_M(x) \tag{A.3}$$

In these terms, the goal of the learning algorithm is to find θ^*, where $g(x, R, f, \epsilon) = h(x, \theta^*)$.

We further characterize the target functions by expressing them in terms of non-linear transfor-

mations, so that $f(x) = \tilde{\theta}^T \phi(x)$, where

$$\phi(x) = [\phi_M(x) \quad \phi_C(x)]^T \tag{A.4}$$

and $\phi_C(x) \in \mathcal{Z}_C$ represents the features of the target function that cannot be captured by the model. By letting the dimension of $\mathcal{Z}_C$ grow as desired, and allowing arbitrary non-linear transformation, f can be as complex as we want. Following the same notation as before, we have $z = [z_M^T \quad z_C^T]^T$, and $\theta = [\theta_M^T \quad \theta_C^T]^T$. Figure 2.5 shows sample target functions that can be generated if we use Fourier harmonics as the features of the non-linear transformation, with $\phi(x) \in \mathbb{R}^{10}$, and $\mathcal{X} = [-1, 1]$. As it is clear from the figure, there is great variety that can be achieved with this model.

We are interested in finding an expression for the out-of-sample error E_{out} in this framework. We begin by finding E_{out} at a point $x \in \mathcal{X}$. This error is also a function of R, f, ϵ, and we denote it by $E_{\text{out}}(x, R, f, \epsilon)$. Since both the stochastic noise ϵ and the complexity of the target function (also known as the deterministic noise) vary from problem to problem, we take the expected value with respect to these quantities. To do this, we make the usual assumption about ϵ, which is that it has zero mean ($\mathbb{E}[\epsilon] = 0$), and diagonal covariance matrix, $\mathbb{E}[\epsilon\epsilon^T] = \sigma_N^2 I$, where I is the identity matrix and σ_N is the standard deviation of the stochastic noise. For the target functions, we make the simplifying assumption that the coefficients of the features outside the model, namely $\theta_C \in \mathcal{Z}_C$, have covariance matrix $\mathbb{E}_[\theta_C \theta_C^T] = \sigma_C^2 I$. Then, the expected out-of-sample error is given by

$$\mathbb{E}_{f,\epsilon}[E_{\text{out}}(x, R, f, \epsilon)] = \mathbb{E}_{f,\epsilon}[(f(x) - g(x, R, f, \epsilon))^2]. \tag{A.5}$$

The final hypothesis $g(x, R, f, \epsilon)$ is obtained by minimizing the squared loss function on the training set, and the solution is given by the pseudo-inverse of the data matrix. To be more precise, let

$$Z = \begin{bmatrix} -z_1^T- \\ -z_2 T- \\ \vdots \\ -z_N^T- \end{bmatrix} \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \tag{A.6}$$

where Z is the data matrix. Now

$$Z = [Z_M \quad Z_C]. \tag{A.7}$$

In matrix form, the learning problem reduces to the following quadratic program,

$$\min_{\theta} ||y - Z\theta||^2 \tag{A.8}$$

with analytic solution $\theta^\star$ found through basic calculus:

$$\theta^\star \;=\; (Z^TZ)^{-1}Z^Ty \;=\; (Z^TZ)^{-1}Z^T(Z\theta+\epsilon) \;=\; \theta + Z^\dagger\epsilon, \tag{A.9}$$

where $Z^\dagger = (Z^TZ)^{-1}Z^T$ is the so-called pseudo-inverse of matrix Z. However, as stated before, we assume the target function is more complex than the learning model. Hence, the matrix Z cannot be constructed as ϕ_C is unknown. Instead, we use the matrix Z_M. In this case the learning algorithm will output the parameter vector $\hat{\theta} \in \mathcal{Z}_M$ given by

$$\hat{\theta} \;=\; Z_M^\dagger y \;=\; Z_M^\dagger Z^T\theta \;=\; Z_M^\dagger(Z_M\theta_M + Z_C\theta_C + \epsilon) \tag{A.10}$$

Hence,

$$g(x,R,f,\epsilon) = z^TZ_M^\dagger(Z_M\theta_M + Z_C\theta_C + \epsilon) \tag{A.11}$$

Substituting, the expected out-of-sample error is given by

$$\begin{aligned}
&\mathbb{E}_{f,\epsilon}\left[E_{\text{out}}(x,R,f\epsilon)\right] \\
&= \mathbb{E}_{f,\epsilon}\left[\|z^T\theta + \epsilon_0 - z_M^T(Z_M^\dagger(Z_M\theta_M + Z_C\theta_C+\epsilon)\|^2\right] \\
&= \mathbb{E}_{\epsilon,f}\left[\|z_C^T\theta_C + \epsilon_0 - z_M^TZ_M^\dagger(Z_C\theta_C+\epsilon)\|^2\right] \\
&= \mathbb{E}_f\left[(z_C^T - z_M^TZ_M^\dagger Z_C)\theta_C\theta_C^T(z_C^T - z_M^TZ_M^\dagger Z_C)^T\right] + \mathbb{E}_\epsilon\left[\epsilon_0^2 + z_M^TZ_M^\dagger\epsilon\epsilon^T(Z_M^\dagger)^Tz_M\right] \\
&= \mathbb{E}_f\left[Tr\left((z_C^T - z_M^TZ_M^\dagger Z_C)\theta_C\theta_C^T(z_C^T - z_M^TZ_M^\dagger Z_C)^T\right)\right] + \mathbb{E}_\epsilon\left[Tr\left(z_M^TZ_M^\dagger\epsilon\epsilon^T(Z_M^\dagger)^Tz_M\right)\right] + \sigma_N^2 \\
&= \mathbb{E}_f\left[Tr\left(\theta_C\theta_C^T(z_C^T - z_M^TZ_M^\dagger Z_C)^T(z_C^T - z_M^TZ_M^\dagger Z_C)\right)\right] + \mathbb{E}_\epsilon\left[Tr(\epsilon\epsilon^T(Z_M^\dagger)^Tz_Mz_M^TZ_M^\dagger)\right] + \sigma_N^2 \\
&= Tr\left(\mathbb{E}_f\left[\theta_C^T\theta_C\right](z_C^T - z_M^TZ_M^\dagger Z_C)^T(z_C^T - z_M^TZ_M^\dagger Z_C)\right) + Tr\left(\mathbb{E}_\epsilon\left[\epsilon\epsilon^T\right](Z_M^\dagger)^Tz_Mz_M^TZ_M^\dagger\right)] + \sigma_N^2 \\
&= Tr\left(\sigma_C^2(z_C^T - z_M^TZ_M^\dagger Z_C)^T(z_C^T - z_M^TZ_M^\dagger Z_C)\right) + Tr\left(\sigma_N^T(Z_M^\dagger)^Tz_Mz_M^TZ_M^\dagger\right) + \sigma_N^2 \\
&= \sigma_C^2(z_C^T - z_M^TZ_M^\dagger Z_C)^T(z_C^T - z_M^TZ_M^\dagger Z_C) + \sigma_N^2Tr\left(z_M^TZ_M^\dagger(Z_M^\dagger)^Tz_M\right) \\
&= \sigma_C^2\|z_C^T - z_M^TZ_M^\dagger Z_C\|^2 + \sigma_N^2Tr\left(z_M^T(Z_M^TZ_M)^{-1}Z_M^TZ_M(Z_M^TZ_M)^{-1}z_M\right) \\
&= \sigma_C^2\|z_C^T - z_M^TZ_M^\dagger Z_C\|^2 + \sigma_N^2z_M^T(Z_M^TZ_M)^{-1}z_M + \sigma_N^2,
\end{aligned} \tag{A.12}$$

where ϵ_0 denotes the stochastic noise at the point x, and $Tr(A)$ denotes the trace of matrix A. The above derivation reorganizes the expression using the fact that the trace of a scalar is the scalar itself, and the fact that $Tr(AB) = Tr(BA)$. Finally, we use the assumptions on the stochastic and deterministic noise to find the expected values.