e-Chapter 9

Learning Aides

Our quest has been for a low out-of-sample error. In the context of specific learning models, we have discussed techniques to fit the data in-sample, and ensure good generalization to out of sample. There are, however, additional issues that are likely to arise in any learning scenario, and a few simple enhancements can often yield a drastic improvement. For example: Did you appropriately preprocess the data to take into account arbitrary choices that might have been made during data collection? Have you removed any irrelevant dimensions in the data that are useless for approximating the target function, but can mislead the learning by adding stochastic noise? Are there properties that the target function is known to have, and if so, can these properties help the learning? Have we chosen the best model among those that are available to us? We wrap up our discussion of learning techniques with a few general tools that can help address these questions.

9.1 Input Preprocessing

do 9.1

Exercise 9.1

The Bank of Learning (BoL) gave Mr. Good and Mr. Bad credit cards based on their (Age, Income) input vector.

	Mr. Good	Mr. Bad
(Age in years, Income in thousands of $)	(47,35)	(22,40)

Mr. Good paid off his credit card bill, but Mr. Bad defaulted. Mr. Unknown who has 'coordinates' (21yrs,$36K) applies for credit. Should the BoL give him credit, according to the nearest neighbor algorithm? If income is measured in dollars instead of in "K" (thousands of dollars), what is your answer?

One could legitimately debate whether age (maturity) or income (financial resources) should be the determining factor in credit approval. It is, however, decidedly not recommended for something like a credit approval to hinge on an apparently arbitrary choice made during data collection, such as the denomination by which income was measured. On the contrary, many standard design choices when learning from data intend each dimension to be treated equally (such as using the Euclidean distance metric in similarity methods or using the sum of squared weights as the regularization term in weight decay). Unless there is some explicit reason not to do so, the data should be presented to the learning algorithm with each dimension on an equal footing. To do so, we need to transform the data to a standardized setting so that we can be immune to arbitrary choices made during data collection.

Recall that the data matrix $\mathrm{X} \in \mathbb{R}^{n\times d}$ has, as its rows, the input data vectors $\mathbf{x}_1, \ldots, \mathbf{x}_{\rm N}$, where $\mathbf{x}_n \in \mathbb{R}^d$ (not augmented with a 1),

$$\mathrm{X} = \begin{bmatrix} — & \mathbf{x}_1^{\rm T} & — \\ — & \mathbf{x}_2^{\rm T} & — \\ & \vdots & \\ — & \mathbf{x}_{\rm N}^{\rm T} & — \end{bmatrix}.$$

The goal of input preprocessing is to transform the data $\mathbf{x}_n \mapsto \mathbf{z}_n$ to obtain the transformed data matrix Z which is standardized in some way. Let $\mathbf{z}_n = \Phi(\mathbf{x}_n)$ be the transformation. It is the data $(\mathbf{z}_n, y_n)$ that is fed into the learning algorithm to produce a learned hypothesis $\tilde{g}(\mathbf{z})$.[1] The final hypothesis g is

$$g(\mathbf{x}) = \tilde{g}(\Phi(\mathbf{x})).$$

Input Centering. Centering is a relatively benign transformation which removes any bias in the inputs by translating the origin. Let $\bar{\mathbf{x}}$ be the in-sample mean vector of the input data, $\bar{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N} \mathbf{x}_n$; in matrix notation, $\bar{\mathbf{x}} = \frac{1}{N}\mathrm{X}^{\rm T}\mathbf{1}$ ($\mathbf{1}$ is the column vector of N 1's). To obtain the transformed vector, simply subtract the mean from each data point,

$$\mathbf{z}_n = \mathbf{x}_n - \bar{\mathbf{x}}.$$

By direct calculation, one can verify that $\mathrm{Z} = \mathrm{X} - \mathbf{1}\bar{\mathbf{x}}^{\rm T}$. Hence,

$$\bar{\mathbf{z}} = \tfrac{1}{N}\mathrm{Z}^{\rm T}\mathbf{1} = \tfrac{1}{N}\mathrm{X}^{\rm T}\mathbf{1} - \tfrac{1}{N}\bar{\mathbf{x}}\mathbf{1}^{\rm T}\mathbf{1} = \bar{\mathbf{x}} - \tfrac{1}{N}\bar{\mathbf{x}} \cdot N = \mathbf{0},$$

where we used $\mathbf{1}^{\rm T}\mathbf{1} = N$ and the definition of $\bar{\mathbf{x}}$. Thus, the transformed vectors are 'centered' in that they have zero mean, as desired. It is clear that no information is lost by centering (as long as one has retained $\bar{\mathbf{x}}$, one can always recover $\mathbf{x}$ from $\mathbf{z}$). If the data is not centered, we can always center it, so from now on, for simplicity and without loss of generality, we will assume that the input data is centered, and so $\mathrm{X}^{\rm T}\mathbf{1} = \mathbf{0}$.

[1] Note that higher-level features or nonlinear transforms can also be used to transform the inputs before input processing.

do 9.2

Exercise 9.2

Define the matrix $\gamma = \mathrm{I} - \frac{1}{N}\mathbf{1}\mathbf{1}^{\mathrm{T}}$. Show that $\mathrm{Z} = \gamma\mathrm{X}$. ($\gamma$ is called the centering operator (see Appendix B.3) which projects onto the space orthogonal to $\mathbf{1}$.)

Input Normalization. Centering alone will not solve the problem that arose in Exercise 9.1. The issue there is one of scale, not bias. Inflating the scale of the income variable exaggerates differences in income when using the standard Euclidean distance, thereby affecting your decision. One solution is to ensure that all the input variables have the same scale. One measure of scale (or spread) is the standard deviation. Since the data is now centered, the in-sample standard deviation σ_i of input variable i is defined by

$$\sigma_i^2 = \frac{1}{N}\sum_{n=1}^{N} x_{ni}^2,$$

where x_{ni} is the ith component of the nth data point. Input normalization transforms the inputs so that each input variable has unit standard deviation (scale). Specifically, the transformation is

$$\mathbf{z}_n = \begin{bmatrix} z_{n1} \\ \vdots \\ z_{nd} \end{bmatrix} = \begin{bmatrix} x_{n1}/\sigma_1 \\ \vdots \\ x_{nd}/\sigma_d \end{bmatrix} = \mathrm{D}\mathbf{x}_n,$$

where D is a diagonal matrix with entries $D_{ii} = 1/\sigma_i$. The scale of all input variables is now 1, since $\tilde{\sigma}_i^2 = 1$ as the following derivation shows ($\tilde{\sigma}_i^2 = 1$ is the variance, or scale, of dimension i in the $\mathcal{Z}$ space).

$$\sigma_i^2(\mathbf{z}) = \frac{1}{N}\sum_{n=1}^{N} z_{ni}^2 = \frac{1}{N}\sum_{n=1}^{N}\frac{x_{ni}^2}{\sigma_i^2} = \frac{1}{\sigma_i^2}\cdot\underbrace{\left(\frac{1}{N}\sum_{n=1}^{N} x_{ni}^2\right)}_{\sigma_i^2} = 1.$$

Exercise 9.3

Consider the data matrix X and the transformed data matrix Z. Show that

$$\mathrm{Z} = \mathrm{X}\mathrm{D} \qquad \text{and} \qquad \mathrm{Z}^{\mathrm{T}}\mathrm{Z} = \mathrm{D}\mathrm{X}^{\mathrm{T}}\mathrm{X}\mathrm{D}.$$

Input Whitening Centering deals with bias in the inputs. Normalization deals with scale. Our last concern is correlations. Strongly correlated input variables can have an unexpected impact on the outcome of learning. For example, with regularization, correlated input variables can render a friendly target function unlearnable. The next exercise shows that a simple function may require excessively large weights to implement if the inputs are correlated. This means it is hard to regularize the learning, which in turn means you become susceptible to noise and overfitting.

Exercise 9.4

Let $\hat{x}_1$ and $\hat{x}_2$ be independent with zero mean and unit variance. You measure inputs $x_1 = \hat{x}_1$ and $x_2 = \sqrt{1-\epsilon^2}\hat{x}_1 + \epsilon\hat{x}_2$.

a-d

(a) What are variance(x_1), variance(x_2) and covariance(x_1, x_2)?

(b) Suppose $f(\hat{\mathbf{x}}) = \hat{w}_1\hat{x}_1 + \hat{w}_2\hat{x}_2$ (linear in the independent variables). Show that f is linear in the correlated inputs, $f(\mathbf{x}) = w_1x_1 + w_2x_2$. (Obtain w_1, w_2 as functions of $\hat{w}_1, \hat{w}_2$.)

(c) Consider the 'simple' target function $f(\hat{\mathbf{x}}) = \hat{x}_1 + \hat{x}_2$. If you perform regression with the correlated inputs $\mathbf{x}$ and regularization constraint $w_1^2 + w_2^2 \le C$, what is the maximum amount of regularization you can use (minimum value of C) and still be able to implement the target?

(d) What happens to the minimum C as the correlation increases ($\epsilon \to 0$).

(e) Assuming that there is significant noise in the data, discuss your results in the context of bias and var.

The previous exercise illustrates that if the inputs are correlated, then the weights cannot be independently penalized, as they are in the standard form of weight-decay regularization. If the measured input variables are correlated, then one should transform them to a set that are uncorrelated (at least in-sample). That is the goal of input whitening.[2]

Remember that the data is centered, so the in-sample covariance matrix is

$$\Sigma = \frac{1}{N}\sum_{n=1}^{N} \mathbf{x}_n\mathbf{x}_n^{\mathrm{T}} = \frac{1}{N}\mathrm{X}^{\mathrm{T}}\mathrm{X}. \tag{9.1}$$

$\Sigma_{ij} = \mathrm{cov}(x_i, x_j)$ is the in-sample covariance of inputs i and j; $\Sigma_{ii} = \sigma_i^2$ is the variance of input i. Assume that Σ has full rank and let its matrix square root be $\Sigma^{\frac{1}{2}}$, which satisfies $\Sigma^{\frac{1}{2}}\Sigma^{\frac{1}{2}} = \Sigma$ (see Problem 9.3 for the computation of $\Sigma^{\frac{1}{2}}$). Consider the whitening transformation

$$\mathbf{z}_n = \Sigma^{-\frac{1}{2}}\mathbf{x}_n,$$

where $\Sigma^{-\frac{1}{2}}$ is the inverse of $\Sigma^{\frac{1}{2}}$. In matrix form, $\mathrm{Z} = \mathrm{X}\Sigma^{-\frac{1}{2}}$. Z is whitened if $\frac{1}{N}\mathrm{Z}^{\mathrm{T}}\mathrm{Z} = \mathrm{I}$. We verify that Z is whitened as follows:

$$\frac{1}{N}\mathrm{Z}^{\mathrm{T}}\mathrm{Z} = \Sigma^{-\frac{1}{2}}\left(\frac{1}{N}\mathrm{X}^{\mathrm{T}}\mathrm{X}\right)\Sigma^{-\frac{1}{2}} = \Sigma^{-\frac{1}{2}}\Sigma\Sigma^{-\frac{1}{2}} = \left(\Sigma^{-\frac{1}{2}}\Sigma^{\frac{1}{2}}\right)\left(\Sigma^{\frac{1}{2}}\Sigma^{-\frac{1}{2}}\right) = \mathrm{I},$$

[2] The term whitening is inherited from signal processing where white noise refers to a signal whose frequency spectrum is uniform; this is indicative of a time series of independent noise realizations. The origin of the term white comes from white light which is a light signal whose amplitude distribution is uniform over all frequencies.

where we used $\Sigma = \Sigma^{\frac{1}{2}}\Sigma^{\frac{1}{2}}$ and $\Sigma^{\frac{1}{2}}\Sigma^{-\frac{1}{2}} = \Sigma^{-\frac{1}{2}}\Sigma^{\frac{1}{2}} = \mathrm{I}$. Thus, for the transformed inputs, every dimension has scale 1 and the dimensions are pairwise uncorrelated. Centering, normalizing and whitening are illustrated on a toy data set in Figure 9.1.

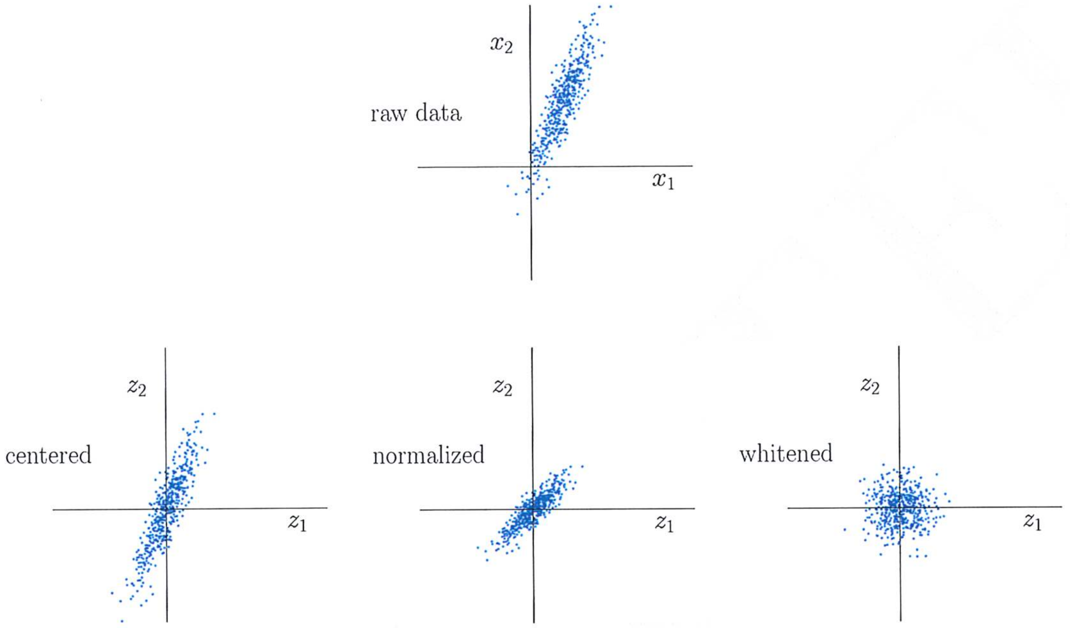

Figure 9.1: Illustration of centering, normalization and whitening.

It is important to emphasize that input preprocessing does not throw away any information because the transformation is invertible: the original inputs can always be recovered from the transformed inputs using $\bar{\mathbf{x}}$ and Σ.

WARNING! Transforming the data to a more convenient format has a hidden trap which easily leads to data snooping.

If you are using a test set to estimate your performance, make sure to determine any input transformation only using the training data. A simple rule: the test data should be kept locked away in its raw form until you are ready to test your final hypothesis. (See Example 5.3 for a concrete illustration of how data snooping can affect your estimate of the performance on your test set if input preprocessing in any way used the test set.) After you determine your transformation parameters from the training data, you should use these same parameters to transform your test data to evaluate your final hypothesis g.