The Elements of Statistical Learning - Chapter 2 Exercises

Exercise 2.1

Suppose each of K-classes has an associated target t_k, which is a vector or all zeros, except a one in the kth position. Show that classifying to the largest element of $\hat{y}$ amount to choosing the closest target, $\min_k \|t_k - \hat{y}\|$, if the elements of $\hat{y}$ sum to one.

Solution

Write $\hat{y} = \sum_{k=1}^{K} a_k t_k$. Then

$$\begin{aligned} \|t_{k_1} - \hat{y}\| \geq \|t_{k_2} - \hat{y}\| &\iff (1 - a_{k_1})^2 + a_{k_2}^2 \geq a_{k_1}^2 + (1 - a_{k_2})^2 \\ &\iff 2a_{k_1} \leq 2a_{k_2} \\ &\iff a_{k_1} \leq a_{k_2}. \end{aligned}$$

The claim follows.

Exercise 2.2

Show how to compute the Bayes decision boundary for the simulation example in Figure 2.5.

Solution

The Bayes decision boundary for binary classification is

$$\{x \mid P(\text{Orange} \mid X = x) = \tfrac{1}{2}\} = \{x \mid P(\text{Blue} \mid X = x) = \tfrac{1}{2}\}.$$

By Bayes' Theorem,

$$P(\text{Blue} \mid X = x) = \frac{P(X = x \mid \text{Blue})\, P(\text{Blue})}{P(X = x \mid \text{Blue})\, P(\text{Blue}) + P(X = x \mid \text{Orange})\, P(\text{Orange})}.$$

Therefore, if f_O and f_B are the probability density functions for the two classes then the Bayes' decision boundary is the line

$$\{x \mid \frac{f_B(x)}{f_B(x) + f_O(x)} = \tfrac{1}{2}\}.$$

Exercise 2.3

Derive equation (2.24).

Solution

Let $X_1, \ldots, X_N$ be random variables representing the N data points and let D denote the distance from the origin to the closest data point. Since the X_i are i.i.d,

$$\begin{aligned} P(D \geq d) &= P\left(\bigcap_{n=1}^{N} (\|X_n\| \geq d)\right) \\ &= \prod_{n=1}^{N} P(\|X_n\| \geq d) \\ &= P(\|X\| \geq d)^N \end{aligned}$$

where $X = X_1$. So

$$P(D \geq d) = \frac{1}{2} \iff P(\|X\| \geq d) = \left(\frac{1}{2}\right)^{1/N}.$$

Let $V(d, p)$ denote the volume of a p-dimensional sphere of radius d. Since the data points are distributed uniformly in the ball,

$$\begin{aligned} P(\|X\| \leq d) &= \frac{V(d,p)}{V(1,p)} \\ P(\|X\| \geq d) &= 1 - \frac{V(d,p)}{V(1,p)}. \end{aligned}$$

So, the median d of D satisfies

$$1 - \frac{V(d,p)}{V(1,p)} = \left(\frac{1}{2}\right)^{1/N} \Rightarrow \frac{V(d,p)}{V(1,p)} = 1 - \left(\frac{1}{2}\right)^{1/N}.$$

But $V(d, p)$ is proportional to d^p and therefore

$$d = \left(1 - \left(\frac{1}{2}\right)^{1/N}\right)^{1/p}.$$

Exercise 2.4

The edge effect problem discussed on page 23 is not peculiar to uniform sampling from bounded domains. Consider inputs drawn from a spherical multinormal distribution $X \sim N(0, \mathbf{I}_p)$. The squared distance from any sample point to the origin has a χ^2_p distribution with mean p. Consider a prediction point x_0 drawn from this distribution, and let $a = x_0/\|x_0\|$ be an associated unit vector. Let $z_i = a^T x_i$ be the projection of each of the training points on this direction.

Show that the z_i are distributed $N(0, 1)$ with expected squared distance from the origin 1, while the target point has expected squared distance p from the origin.

Hence for $p = 10$, a randomly drawn test point is about 3.1 standard deviations from the origin, while all the training points are on average one standard deviation along direction a. So most prediction points see themselves as lying on the edge of the training set.

Solution

The distribution $N(0, \mathbf{I}_p)$ is spherically symmetric. Since a is normalised, z_i is independent of x_0. Hence is suffices to determine the distribution of one of the components of x_i.

Let $\pi : \mathbb{R}^p \to \mathbb{R}$ denote the projection onto its first component and let f_X denote the probability density function of X. Then

$$P(\pi(X) \leq a) = \int f_X(x)\, \mathrm{d}x,$$

where the integral is taken over all $x \in \mathbb{R}^p$ whose first component is less than or equal to a. Since f_X is a product of standard normal pdfs f_N for the different components, the integral above simplifies to

$$\left(\int_{x \leq a} f_N(x)\, \mathrm{d}x\right) \cdot \left(\int_{\mathbb{R}} f_N(x)\, \mathrm{d}x\right)^{N-1} = \int_{x \leq a} f_N(x)\, \mathrm{d}x.$$

Thus z_i has a standard normal distribution and $E(Z^2) = \mathrm{Var}(Z) + E(Z)^2 = 1$.

Exercise 2.5

(a) Derive equation (2.27). The last line makes use of (3.8) through a conditioning argument.

(b) Derive equation (2.28), making use of the cyclic property of the trace operator (trace(AB) = trace(BA)), and its linearity (which allows us to interchange the order of trace and expectation).

Solution

(a) We have

$$\begin{aligned} \mathrm{EPE}(x_0) &= \mathrm{E}_{y_0, \mathcal{T}}\left((y_0 - \hat{y}_0)^2\right) \\ &= \mathrm{E}_{y_0}\left(\mathrm{E}_{\mathcal{T}}\left((y_0 - \hat{y}_0)^2 \mid y_0\right)\right) \\ &= \mathrm{E}_{y_0}\left(\mathrm{E}_{\mathcal{T}}(\hat{y}_0^2) - 2y_0 \mathrm{E}_{\mathcal{T}}(\hat{y}_0) + y_0^2\right) \\ &= \mathrm{E}_{\mathcal{T}}(\hat{y}_0^2) - 2\mathrm{E}_{y_0}(y_0)\mathrm{E}_{\mathcal{T}}(\hat{y}_0) + \mathrm{E}_{y_0}(y_0^2) \\ &= \mathrm{Var}(\hat{y}_0) + \left(\mathrm{E}(\hat{y}_0) - \mathrm{E}(y_0)\right)^2 + \mathrm{Var}(y_0) \\ &= \mathrm{Var}(x_0^{\mathrm{T}}\hat{\beta}) + \left(x_0^{\mathrm{T}}\beta - x_0^{\mathrm{T}}\beta\right)^2 \sigma^2 \\ &= x_0^{\mathrm{T}}\mathrm{Var}(\hat{\beta})x_0 + \sigma^2. \end{aligned}$$

We can split $\mathcal{T} = (\mathbf{X}, \mathbf{y})$. So using the conditional variance identity:

$$\begin{aligned} \mathrm{Var}_{\mathcal{T}}(\hat{\beta}) &= \mathrm{E}_{\mathbf{X}}\left(\mathrm{Var}_{\mathbf{y}}(\hat{\beta} \mid \mathbf{X})\right) + \mathrm{Var}_{\mathbf{X}}\left(\mathrm{E}_{\mathbf{y}}(\hat{\beta} \mid \mathbf{X})\right) \\ &= \mathrm{E}_{\mathbf{X}}\left((\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\sigma^2\right) + \mathrm{Var}_{\mathbf{X}}(\beta) \\ &= \mathrm{E}\left((\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\sigma^2\right), \end{aligned}$$

where the second line used (3.8). Therefore

$$\mathrm{EPE}(x_0) = \sigma^2 \mathrm{E}\left(x_0^{\mathrm{T}}(\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}x_0\right) + \sigma^2$$

as required.

(b) First observe that the (i, j) entry of $\mathbf{X}^{\mathrm{T}}\mathbf{X}$ is $\sum_k x_{ki}x_{kj}$, that is, N times the sample mean of $X_i X_j$. Since $\mathrm{E}(X) = 0$ by assumption, the covariance matrix $\mathrm{Var}(X)$ has (i, j) entry $\mathrm{E}(X_i X_j)$ and so by the Law of Large Numbers $\mathbf{X}^{\mathrm{T}}\mathbf{X} \approx N\mathrm{Var}(X)$ for N large.

Using the previous result and the fact that $\mathrm{Var}(X)$ is independent of the sample $\mathcal{T}$,

$$\begin{aligned} \mathrm{E}_{x_0}\mathrm{EPE}(x_0) &= \mathrm{E}_{x_0}\left(\mathrm{E}_{\mathcal{T}}\left(\sigma^2 x_0^{\mathrm{T}}(N\mathrm{Var}(X))^{-1}x_0\right)\right) + \sigma^2 \\ &= \frac{\sigma^2}{N}\mathrm{E}_{x_0}\left(x_0^{\mathrm{T}}\mathrm{Var}(X)^{-1}x_0\right) + \sigma^2. \end{aligned}$$

Using the cyclic property of the trace and the fact that expectation is linear (and hence commutes with trace),

$$\begin{aligned} \mathrm{E}_{x_0}\left(x_0^{\mathrm{T}}\mathrm{Var}(X)^{-1}x_0\right) &= \mathrm{E}_{x_0}\left(\mathrm{Tr}\left(x_0^{\mathrm{T}}\mathrm{Var}(X)^{-1}x_0\right)\right) \\ &= \mathrm{E}_{x_0}\left(\mathrm{Tr}\left(\mathrm{Var}(X)^{-1}x_0x_0^{\mathrm{T}}\right)\right) \\ &= \mathrm{Tr}\left(\mathrm{E}_{x_0}\left(\mathrm{Var}(X)^{-1}x_0x_0^{\mathrm{T}}\right)\right) \\ &= \mathrm{Tr}\left(\mathrm{Var}(X)^{-1}\mathrm{E}_{x_0}\left(x_0x_0^{\mathrm{T}}\right)\right) \\ &= \mathrm{Tr}\left(\mathrm{Var}(X)^{-1}\mathrm{Var}(x_0)\right), \end{aligned}$$

as $\mathrm{E}(x_0) = 0$. But x_0 is just an observation of X, so $\mathrm{Var}(X)^{-1}\mathrm{Var}(x_0) = I_p$ and therefore

$$\mathrm{E}_{x_0}\mathrm{EPE}(x_0) = \left(\frac{p}{N}\right)\sigma^2 + \sigma^2.$$

Exercise 2.6

Consider a regression problem with inputs x_i and outputs y_i, and a parameterized model $f_\theta(x)$ to be fit by least squares. Show that if there are observations with tied or identical values of x, then the fit can be obtained from a reduced weighted least squares problem.

Solution

Assume without loss of generality that $x_1 = x_2$. Then

$$\begin{aligned} (y_1 - f_\theta(x_1))^2 + (y_2 - f_\theta(x_2))^2 &= y_1^2 + y_2^2 - 2(y_1 + y_2)f_\theta(x) + 2f_\theta(x)^2 \\ &= 2\left(\frac{y_1 + y_2}{2} - f_\theta(x)\right)^2 - y_1y_2 + \frac{y_1^2 + y_2^2}{2}, \end{aligned}$$

where we write $x = x_1 = x_2$. In choosing θ to minimise this we can ignore the last two terms and so the original least squares problem is equivalent to minimising

$$2\left(\frac{y_1 + y_2}{2} - f_\theta(x)\right)^2 + \sum_{i=3}^{N} (y_i - f_\theta(x_i))^2$$

over θ; a reduced weighted least squares problem.

Exercise 2.7

Suppose we have a sample of N pairs x_i, y_i drawn i.i.d. from the distribution characterized as follows:

$$\begin{aligned} &x_i \sim h(x), \text{ the design density} \\ &y_i = f(x_i) + \epsilon_i, \text{ is the regression function} \\ &\epsilon_i \sim (0, \sigma^2) \text{ (mean zero, variance } \sigma^2) \end{aligned}$$

We construct an estimator for f linear in the y_i,

$$\hat{f}(x_0) = \sum_{i=1}^{N} l_i(x_i; \mathcal{X})y_i,$$

where the weights $l_i(x_i; \mathcal{X})$ do not depend on the y_i, but do depend on the entire training sequence of x_i, denoted here by $\mathcal{X}$.

(a) Show that linear regression and k-nearest-neighbor regression are members of this class of estimators. Describe explicitly the weights $l_i(x_i; \mathcal{X})$ in each of these cases.

(b) Decompose the conditional mean-squared error

$$\mathrm{E}_{\mathcal{Y}|\mathcal{X}}\left((f(x_0) - \hat{f}(x_0))^2\right)$$

into a conditional squared bias and a conditional variance component. Like $\mathcal{X}$, $\mathcal{Y}$ represents the entire training sequence of y_i.

(c) Decompose the (unconditional) mean-squared error

$$\mathrm{E}_{\mathcal{Y},\mathcal{X}}\left((f(x_0) - \hat{f}(x_0))^2\right)$$

into a squared bias and a variance component.

(d) Establish a relationship between the squared biases and variances in the above two cases.

Solution

(a) In linear regression we have

$$\hat{f}(x_0) = x_0^{\mathrm{T}}\hat{\beta} = x_0^{\mathrm{T}}(\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y} = \sum_{i=1}^{N} l_i(x_i; \mathcal{X})y_i,$$

where $l_i(x_i; \mathcal{X})$ is the ith element of $x_0^{\mathrm{T}}(\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}$.

In k-nearest neighbour,

$$\hat{f}(x_0) = \frac{1}{k}\sum_{x_i \in N_k(x_0)} y_i = \sum_{i=1}^{N} l_i(x_i; \mathcal{X})y_i,$$

where

$$l_i(x_i; \mathcal{X}) = \begin{cases} \frac{1}{k} & \text{if } x_i \in N_k(x_0) \\ 0 & \text{otherwise.} \end{cases}$$

(b) We have

$$\mathrm{E}_{\mathcal{Y}|\mathcal{X}}\left(\hat{f}(x_0)\right) = \sum_{i=1}^{N} l_i(x_i; \mathcal{X})\mathrm{E}_{\mathcal{Y}|\mathcal{X}}(y_i) = \sum_{i=1}^{N} l_i(x_i; \mathcal{X})f(x_i)$$

and

$$\mathrm{Var}_{\mathcal{Y}|\mathcal{X}}\left(\hat{f}(x_0)\right) = \sum_{i=1}^{N} l_i(x_i; \mathcal{X})\mathrm{Var}_{\mathcal{Y}|\mathcal{X}}(y_i) = \left(\sum_{i=1}^{N} l_i(x_i; \mathcal{X})^2\right)\sigma^2.$$

So the bias-variance decomposition is

$$\begin{aligned} \mathrm{E}_{\mathcal{Y}|\mathcal{X}}\left((f(x_0) - \hat{f}(x_0))^2\right) &= \mathrm{Var}_{\mathcal{Y}|\mathcal{X}}\left(\hat{f}(x_0)\right) + \mathrm{Var}_{\mathcal{Y}|\mathcal{X}}\left(\hat{f}(x_0)\right)^2 \\ &= \left(\sum_{i=1}^{N} l_i(x_i; \mathcal{X})^2\right)\sigma^2 + \left(\sum_{i=1}^{N} l_i(x_i; \mathcal{X})f(x_i) - f(x_0)\right)^2. \end{aligned}$$

(c), (d) I really don't know what they're looking for here.

Exercise 2.8

Compare the classification performance of linear regression and k– nearest neighbor classification on the zipcode data. In particular, consider only the 2's and 3's, and $k = 1, 3, 5, 7$ and 15. Show both the training and test error for each choice.

Solution

Begin by importing packages. The models.py module was written for this exercise.

In [8]:

```
# Import packages
import numpy as np
import pandas as pd
import models
#from models import train_linear_model, apply_linear_model, apply_k_nearest_neighbour, classification_e
rror
```

First we use Pandas to import and clean the data

In [9]:

```
# Import data
train = pd.read_csv('zipcode_train.csv', sep = ' ', header = None)
test = pd.read_csv('zipcode_test.csv', sep = ' ', header = None)

# Rename first columns to y for clarity
train = train.rename(columns = {0 : 'y'})
test = test.rename(columns = {0 : 'y'})

# Restrict to 2s and 3s
train = train[train.y.isin([2, 3])]
test = test[test.y.isin([2, 3])]

# train has an extra NaN column for some reason - drop it
train = train.drop(columns = 257)
```

Now switch to Numpy array for linear algebra operations

In [10]:

```
# Set up numpy arrays
Xtrain = train.iloc[:, 1:].to_numpy()
ytrain = train.loc[:, 'y'].to_numpy()

Xtest = test.iloc[:, 1:].to_numpy()
ytest = test.loc[:, 'y'].to_numpy()

# Store dimensions
ptrain, Ntrain = Xtrain.shape
ptest, Ntest = Xtest.shape

# Replace 2, 3 in y with 0, 1 resp.
ytrain = np.where(ytrain == 2, 0, 1)
ytest = np.where(ytest == 2, 0, 1)
```

First train and test a linear model.

In [11]:

```
# Train linear model
beta = models.train_linear_model(Xtrain, ytrain)

# Calculate classification errors
model = models.apply_linear_model
model_args = [beta]

print('Classification Error for Linear Model\n')

error_train = models.classification_error(Xtrain, ytrain, model, *model_args)
error_test = models.classification_error(Xtest, ytest, model, *model_args)

print('Training error: {}'.format(error_train))
print('Test error: {}'.format(error_test))
```

```
Classification Error for Linear Model

Training error: 0.005759539236861051
Test error: 0.04120879120879121
```

Now k-nearest neighbour.

In [12]:

```
# Calculate classification errors
model = models.apply_k_nearest_neighbour

print('Classification Error for k-Nearest Neighbours')

for k in [1, 3, 5, 7, 15]:
    print('\n')
    model_args = [Xtrain, ytrain, k]

    error_train = models.classification_error(Xtrain, ytrain, model, *model_args)
    print('Training error (k = {}): {}'.format(k, error_train))

    error_test = models.classification_error(Xtest, ytest, model, *model_args)
    print('Test error (k = {}): {}'.format(k, error_test))
```

```
Classification Error for k-Nearest Neighbours

Training error (k = 1): 0.0
Test error (k = 1): 0.024725274725274724


Training error (k = 3): 0.005039596832253419
Test error (k = 3): 0.03021978021978022


Training error (k = 5): 0.005759539236861051
Test error (k = 5): 0.03021978021978022


Training error (k = 7): 0.0064794816414686825
Test error (k = 7): 0.03296703296703297


Training error (k = 15): 0.009359251259899209
Test error (k = 15): 0.038461538461538464
```

The results are surprising. We would expect the test error for k-nearest neighbours to be concave, but in fact it is most accurate on the test set when $k = 1$.

Exercise 2.9

Consider a linear regression model with p parameters, fit by least squares to a set of training data $(x_1, y_1), \ldots, (x_N, y_N)$ drawn at random from a population. Let $\hat{\beta}$ be the least squares estimate. Suppose we have some test data $(\tilde{x}_1, \tilde{y}_1), \ldots, (\tilde{x}_N, \tilde{y}_N)$ drawn at random from the same populations as the training data. If $R_{\mathrm{tr}}(\beta) = \frac{1}{N}\sum_1^N (y_i - \beta^{\mathrm{T}}x_i)^2$ and $R_{\mathrm{te}}(\beta) = \frac{1}{M}\sum_1^M (\tilde{y}_i - \beta^{\mathrm{T}}\tilde{x}_i)^2$, prove that

$$\mathrm{E}\left[R_{\mathrm{tr}}(\hat{\beta})\right] \leq \mathrm{E}\left[R_{\mathrm{te}}(\hat{\beta})\right],$$

where the expectations are over all that is random in each expression. [This exercise was brought to our attention by Ryan Tibshirani, from a homework assignment given by Andrew Ng.]

Solution

First assume that $N = M$. Note that $\mathrm{E}(R_{\mathrm{tr}}(\hat{\beta}))$ is the expected minimum value of $\frac{1}{N}(\mathbf{y} - \mathbf{X}\beta)^{\mathrm{T}}(\mathbf{y} - \mathbf{X}\beta)$ over all β. Conversely, $\mathrm{E}(R_{\mathrm{te}}(\hat{\beta}))$ is the expected value of the same expression for a value of β trained on a different data set and so must be larger. But, the test data is i.i.d, so $\mathrm{E}(R_{\mathrm{tr}}(\hat{\beta})) = \mathrm{E}\left((y - \hat{\beta}x)^2\right)$ is independent of M. The conclusion follows.