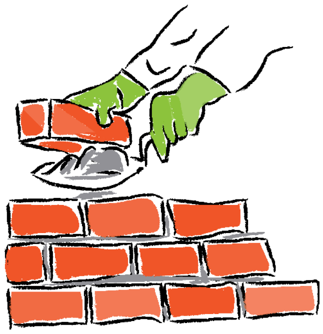

Chapter 13
Supervised Learning

A frequent question in biological and biomedical applications is whether a property of interest (say, disease type, cell type, the prognosis of a patient) can be "predicted", given one or more other properties, called the **predictors**. Often we are motivated by a situation in which the property to be predicted is unknown (it lies in the future, or is hard to measure), while the predictors are known. The crucial point is that we **learn** the prediction rule from a set of training data in which the property of interest is also known. Once we have the rule, we can either apply it to new data, and make actual predictions of unknown outcomes; or we can dissect the rule with the aim of better understanding the underlying biology.

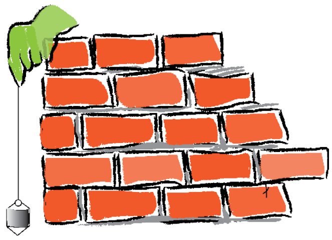

Figure 13.1: In a supervised learning setting, we have a yardstick or plumbline to judge how well we are doing: the response itself.

Compared to unsupervised learning and what we have seen in Chapters 5, 7 and 9, where we do not know what we are looking for or how to decide whether our result is "right", we are on much more solid ground with supervised learning: the objective is clearly stated, and there are straightforward criteria to measure how well we are doing.

The central issue in **supervised learning**[1] is **overfitting** and **generalisability**: did we just learn the training data "by heart" by constructing a rule that has 100% accuracy on the training data, but would perform poorly on any new data? Or did our rule indeed pick up some of the pertinent patterns in the system being studied, which will also apply to yet unseen new data?

13.1 Goals for this chapter

In this chapter we will

- see exemplary applications that motivate the use of supervised learning methods
- learn what discriminant analysis does,
- define measures of performance,
- encounter the curse of dimensionality and see what overfitting is,
- find out about regularisation and understand the concepts of generisability and model complexity,

[1] Sometimes the term **statistical learning** is used, more or less exchangeably.

- see how to use cross-validation to tune parameters of the algorithms,
- get to see a unified framework for machine learning algorithms in R that allows you to use hundreds of methods in a consistent manner,
- discuss method hacking.

13.2 What are the data?

The basic data structure for both supervised and unsupervised learning is (at least conceptually) a dataframe, where each row corresponds to an object and the columns are different features[2] of the objects. While in unsupervised learning we aim to find (dis)similarity relationships between the objects based on their feature values (e. g., by clustering or ordination), in supervised learning we aim to find a mathematical function (or a computational algorithm) that predicts the value of one of the features from the other features. Many implementations require that there are no missing values, whereas other methods can be generalized to work with some amount of missing data.

The feature that we select over all the others with the aim of predicting is called the **objective** or the **response**. Sometimes the choice is natural, but sometimes it is also instructive to reverse the roles, especially if we are interested in dissecting the prediction function for the purpose of biological understanding, or in disentangling correlations from causation.

The framework for supervised learning covers both continuous and categorical response variables. In the continuous case we also call it **regression**, in the categorical case, **classification**. It turns out that this distinction is not a detail, as it has quite far-reaching consequences for the choice of loss function (Section 13.5) and thus the choice of algorithm (Friedman, 1997).

The first question to consider in any supervised learning task is how the number of objects compares to the number of predictors. The more data, the better, and much of the hard work in supervised learning has to do with overcoming the limitations of having a finite (and typically, too small) training set.

▶ Question **13.2.1.** Give examples where we have encountered instances of supervised learning with a categorical response in this book.

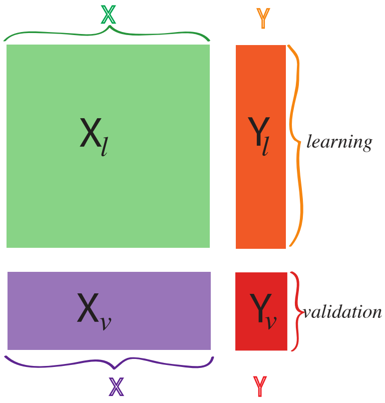

Figure 13.2: In supervised learning, we assign two different roles to our variables. We have labeled the explanatory variables X and the response variable(s) Y. There are also two different sets of observations: the training set X_ℓ and Y_ℓ and the validation set X_v and Y_v.

13.2.1 Motivating examples

Predicting diabetes type

The `diabetes` dataset (Reaven and Miller, 1979) presents three different groups of diabetes patients and five clinical variables measured on them.

```
library("ggplot2")
library("readr")
```

[2] Features are usually numerical scalars or categorical variables, although some methods can be generalized to work with other data types.

```
library("magrittr")
diabetes = read_csv("../data/diabetes.csv", col_names = TRUE)
diabetes

## # A tibble: 144 x 7
##       id relwt glufast glutest steady insulin group
##    <int> <dbl>   <int>   <int>  <int>   <int> <int>
## 1      1  0.81      80     356    124      55     3
## 2      3  0.94     105     319    143     105     3
## 3      5  1.00      90     323    240     143     3
## 4      7  0.91     100     350    221     119     3
## 5      9  0.99      97     379    142      98     3
## 6     11  0.90      91     353    221      53     3
## 7     13  0.96      78     290    136     142     3
## 8     15  0.74      86     312    208      68     3
## 9     17  1.10      90     364    152      76     3
## 10    19  0.83      85     296    116      60     3
## # ... with 134 more rows

diabetes$group %<>% factor
```

We used the forward-backward pipe operator %<>% to convert the group column into a factor.

```
library("reshape2")
ggplot(melt(diabetes, id.vars = c("id", "group")),
       aes(x = value, col = group)) +
  geom_density() + facet_wrap( ~variable, ncol = 2, scales = "free")
```

The plot is shown in Figure 13.3.

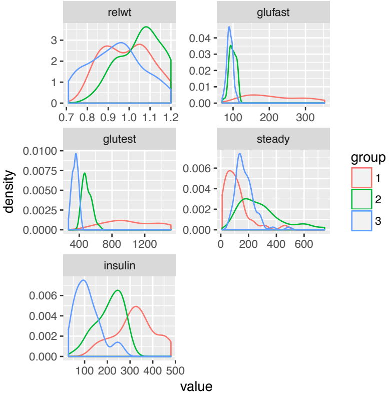

Figure 13.3: We see already from the one-dimensional distributions that some of the individual variables could potentially predict which group a patient is more likely to belong to. Our goal will be to combine variables to improve these one dimensional predictions.

Predicting cellular phenotypes

Neumann et al. (2010) observed human cancer cells using live-cell imaging. The cells were genetically engineered so that their histones were tagged with a green fluorescent protein (GFP). A genome-wide RNAi library was applied to the cells, and for each siRNA perturbation, movies of a few hundred cells were recorded for about two days, to see what effect the depletion of each gene had on cell cycle, nuclear morphology and cell proliferation. Their paper reports the use of an automated image classification algorithm that quantified the visual appearance of each cell's nucleus and enabled the prediction of normal mitosis states or aberrant nuclei. The algorithm was trained on the data from around 3000 cells that were annotated by a human expert. It was then applied to almost 2 billions images of nuclei (Figure 13.4). Using automated image classification provided scalablity (annotating 2 billion images manually would take a long time) and objectivity.

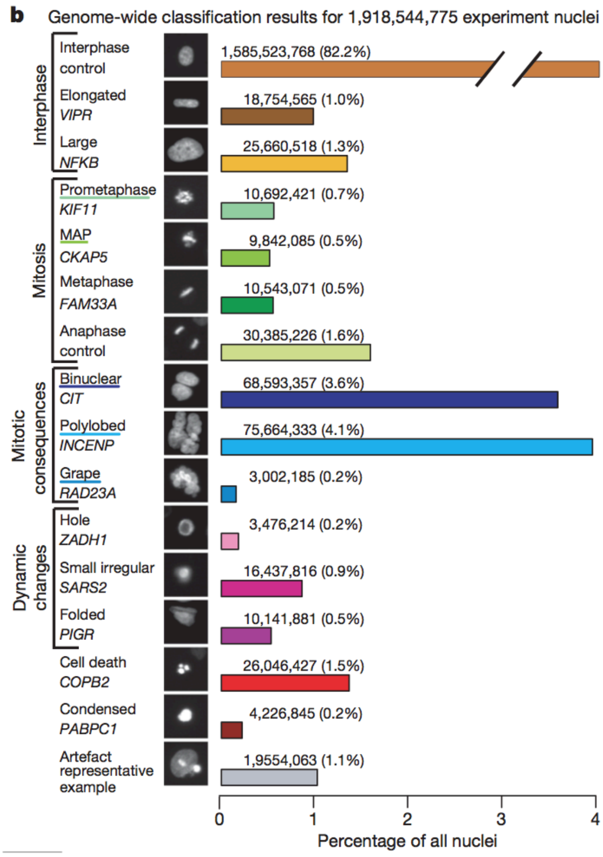

Figure 13.4: The data were images of $2x10^9$ nuclei from movies. The images were segmented to identify the nuclei, and numeric features were computed for each nucleus, corresponding to size, shape, brightness and lots of other more or less abstract quantitative summaries of the joint distribution of pixel intensities. From the features, the cells were classified into 16 different nuclei morphology classes, represented by the rows of the barplot. Representative images for each class are shown in black and white in the center column. The class frequencies, which are very unbalanced, are shown by the lengths of the bars.

Predicting embryonic cell states

We will revisit the mouse embryo data (Ohnishi et al., 2014), which we have already seen in Chapters 3, 5 and 7, and show how we can predict the developmental state (Embryonic Days) from the gene expression measurements.

13.3 Linear discrimination

We start with one of the simplest possible discrimination problems[3], where we have objects described by two continuous features (so the objects can be thought of as points in the 2D plane) and falling into three groups. Our aim is to define class boundaries, which are lines in the 2D space.

13.3.1 Diabetes data

Let's see whether we can predict the feature `group` from the features `insulin` and `glutest` variables in the `diabetes` data. It's always a good idea to first visualise the data (Figure 13.5).

[3] Arguably the simplest possible problem is a single continuous feature, two classes, and the task of finding a single threshold to discriminate between the two groups.

```
ggdb = ggplot(mapping = aes(x = insulin, y = glutest)) +
  geom_point(aes(colour = group), data = diabetes)
ggdb
```

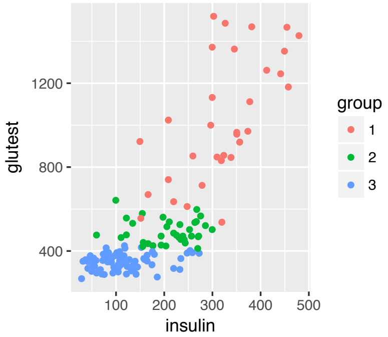

Figure 13.5: Scatterplot of two of the variables in the `diabetes` data. Each point is a sample, and the color indicates the diabetes type as encoded in the `group` variable.

We'll start with a method called **linear discriminant analysis** (**LDA**). This method is a foundation stone of classification, many of the more complicated (and sometimes more powerful) algorithms are really just generalisations of LDA.

```
library("MASS")
diabetes_lda = lda(group ~ insulin + glutest, data = diabetes)
diabetes_lda

## Call:
## lda(group ~ insulin + glutest, data = diabetes)
##
## Prior probabilities of groups:
##         1         2         3
## 0.2222222 0.2500000 0.5277778
##
## Group means:
##    insulin   glutest
## 1 320.9375 1027.3750
## 2 208.9722  493.9444
## 3 114.0000  349.9737
##
## Coefficients of linear discriminants:
##                  LD1         LD2
## insulin -0.004463900 -0.01591192
## glutest -0.005784238  0.00480830
##
## Proportion of trace:
##    LD1    LD2
## 0.9677 0.0323

ghat = predict(diabetes_lda)$class
table(ghat, diabetes$group)

##
## ghat  1  2  3
##    1 25  0  0
##    2  6 24  6
##    3  1 12 70

mean(ghat != diabetes$group)

## [1] 0.1736111
```

▶ Question 13.3.1. What do the different parts of the above output mean?

Now, let's visualise the LDA result[4]. We are going to plot the prediction regions for each of the three groups. We do this by creating a grid of points and using our prediction rule on each of them. We'll then also dig a bit deeper into the mechanics of LDA and plot the class centers (`diabetes_lda$means`) and ellipses that correspond to the fitted covariance matrix (`diabetes_lda$scaling`). Assembling this

[4] Note how we first visualised the data, in Figure 13.5, and are now going to visualise the fitted model (Figure 13.6). The prediction regions can, in principle, be shown for any classification method, including a "black box" method. On the other hand, the cluster centers and ellipses in Figure 13.6 are a method-specific visualisation.

visualization requires us to write a bit of code.

```
make1Dgrid = function(x) {
  rg  = range(x)
  wid = diff(rg)
  rg  = rg + wid * 0.05 * c(-1, 1)
  seq(from = rg[1], to = rg[2], length.out = 100)
}
```

Set up the points for prediction, a 100 × 100 grid that covers the data range.

```
diabetes_grid = with(diabetes,
  expand.grid(insulin = make1Dgrid(insulin),
              glutest = make1Dgrid(glutest)))
```

Do the predictions.

```
diabetes_grid$ghat =
  predict(diabetes_lda, newdata = diabetes_grid)$class
```

The group centers.

```
centers = diabetes_lda$means
```

Compute a unit circle and an affine transformation of the circle into the ellipse we want to plot.

```
unitcircle = exp(1i * seq(0, 2*pi, length.out = 90)) %>%
          (function(x) cbind(Re(x), Im(x)))
ellipse = unitcircle %*% solve(diabetes_lda$scaling)
```

All three ellipses, one for each group center.

```
ellipses = lapply(seq_len(nrow(centers)), function(i) {
  (ellipse +
   matrix(centers[i, ], byrow = TRUE,
          ncol = ncol(centers), nrow = nrow(ellipse))) %>%
     cbind(group = i)
}) %>% do.call(rbind, .) %>% data.frame
ellipses$group %<>% factor
```

Now we are ready to plot (Figure 13.6).

```
ggdb + geom_raster(aes(fill = ghat),
            data = diabetes_grid, alpha = 0.4, interpolate = TRUE) +
    geom_point(data = as_data_frame(centers), pch = "+", size = 8) +
    geom_path(aes(colour = group), data = ellipses) +
    scale_x_continuous(expand = c(0, 0)) +
    scale_y_continuous(expand = c(0, 0))
```

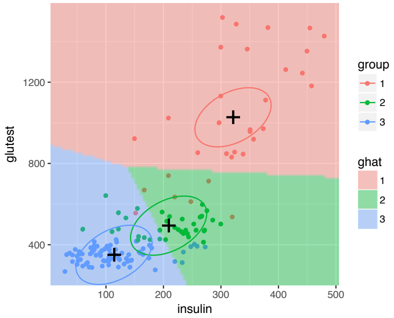

Figure 13.6: As Figure 13.5, with the classification regions from the LDA model shown.

▶ Question **13.3.2.** Why is the boundary between the prediction regions for groups 1 and 2 not perpendicular to the line between the cluster centers?

▶ Question **13.3.3.** How confident would you be about the predictions in those areas of the 2D plane that are far from all of the cluster centers?

▶ Question **13.3.4.** Why is the boundary between the prediction regions for groups 2 and 3 not half-way between the centers, but shifted in favor of class 3? (Hint: have a look at the `prior` argument of `lda`.) Try again with uniform prior.

▶ Answer **13.3.1.** The result of the following code chunk is shown in Figure 13.7.

```
diabetes_up = lda(group ~ insulin + glutest, data = diabetes,
  prior = with(diabetes, rep(1/nlevels(group), nlevels(group))))

diabetes_grid$ghup =
  predict(diabetes_up, newdata = diabetes_grid)$class

stopifnot(all.equal(diabetes_up$means, diabetes_lda$means))

ellipse_up  = unitcircle %*% solve(diabetes_up$scaling)
ellipses_up = lapply(seq_len(nrow(centers)), function(i) {
  (ellipse_up +
   matrix(centers[i, ], byrow = TRUE,
          ncol = ncol(centers), nrow = nrow(ellipse_up))) %>%
     cbind(group = i)
}) %>% do.call(rbind, .) %>% data.frame
ellipses_up$group %<>% factor

ggdb + geom_raster(aes(fill = ghup),
            data = diabetes_grid, alpha = 0.4, interpolate = TRUE) +
    geom_point(data = data.frame(centers), pch = "+", size = 8) +
    geom_path(aes(colour = group), data = ellipses_up) +
    scale_x_continuous(expand = c(0, 0)) +
    scale_y_continuous(expand = c(0, 0))
```

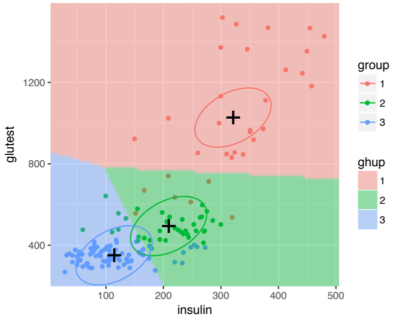

Figure 13.7: As Figure 13.6, but with uniform class priors.

The `stopifnot` line confirms that the class centers are the same –they are independent of the prior–, but the joint covariance is not.

▶ Question **13.3.5.** What is the difference in the prediction accuracy if we use all 5 variables instead of just `insulin` and `glufast`?

▶ Answer **13.3.2.**

```
diabetes_lda5 = lda(group ~ relwt + glufast + glutest +
           steady + insulin, data = diabetes)
diabetes_lda5

## Call:
## lda(group ~ relwt + glufast + glutest + steady + insulin, data = diabetes)
##
## Prior probabilities of groups:
##         1         2         3
## 0.2222222 0.2500000 0.5277778
##
## Group means:
##       relwt   glufast   glutest   steady  insulin
## 1 0.9915625 213.65625 1027.3750 108.8438 320.9375
## 2 1.0558333  99.30556  493.9444 288.0000 208.9722
## 3 0.9372368  91.18421  349.9737 172.6447 114.0000
##
## Coefficients of linear discriminants:
##                   LD1           LD2
## relwt   -1.339546e+00 -3.7950612048
## glufast  3.301944e-02  0.0373202882
## glutest -1.263978e-02 -0.0068947755
## steady   1.240248e-05 -0.0059924778
## insulin -3.895587e-03  0.0005754322
##
## Proportion of trace:
##    LD1    LD2
## 0.8784 0.1216
```

```
ghat5 = predict(diabetes_lda5)$class
table(ghat5, diabetes$group)
```

```
##
## ghat5  1  2  3
##     1 26  0  0
##     2  5 31  3
##     3  1  5 73
```

```
mean(ghat5 != diabetes$group)
```

```
## [1] 0.09722222
```

▶ Question **13.3.6.** Instead of approximating the prediction regions by classification from a grid of points, compute the separating lines explicitly from the linear determinant coefficients.

▶ Answer **13.3.3.** See Section 4.3, Equation (4.10) in (Hastie et al., 2008).

13.3.2 Predicting embryonic cell state from gene expression

Assume that we already know that the four genes *FN1*, *TIMD2*, *GATA4* and *SOX7* are relevant to the classification task[5]. We want to build a classifier that predict the developmental time (embryonic days, E3.25, E3.5, E4.5). We load the data and select four corresponding probes.

```
library("Hiiragi2013")
data("x")
probes = c("1426642_at","1418765_at","1418864_at","1416564_at")
embryoCells = as_data_frame(t(exprs(x)[probes, ])) %>%
  mutate(Embryonic.day = x$Embryonic.day) %>%
  filter(x$genotype == "WT")
```

We can use the Bioconductor annotation package associated with the microarray to verify that the probes correspond to the intended genes,

```
annotation(x)
```

```
## [1] "mouse4302"
```

```
library("mouse4302.db")
anno = AnnotationDbi::select(mouse4302.db, keys = probes,
         columns = c("SYMBOL", "GENENAME"))
anno
```

[5] Later in this chapter we will see methods that can drop this assumption and screen all available features.

```
##       PROBEID SYMBOL
## 1 1426642_at    Fn1
## 2 1418765_at  Timd2
## 3 1418864_at  Gata4
## 4 1416564_at   Sox7
##                                               GENENAME
## 1                                        fibronectin 1
## 2 T cell immunoglobulin and mucin domain containing 2
## 3                               GATA binding protein 4
## 4                SRY (sex determining region Y)-box 7

mt = match(anno$PROBEID, colnames(embryoCells))
colnames(embryoCells)[mt] = anno$SYMBOL
```

and produce a pairs plot (Figure 13.8).

```
library("GGally")
ggpairs(embryoCells, mapping = aes(col = Embryonic.day),
  columns = anno$SYMBOL, upper = list(continuous = "points"))
```

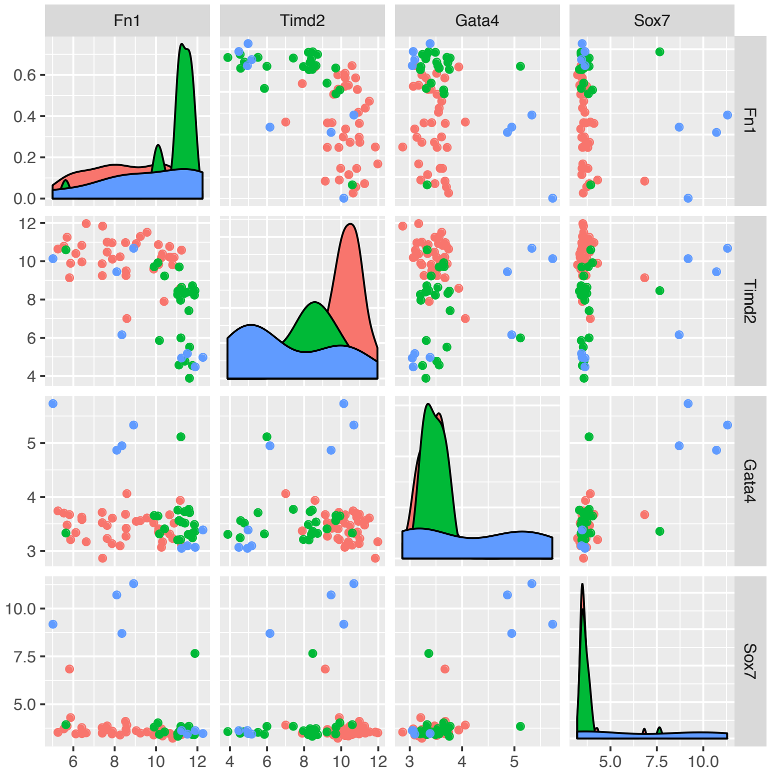

Figure 13.8: Expression values of the discriminating genes, with the prediction target Embryonic.day shown by color.

We can now call `lda` on these data. The linear combinations LD1 and LD2 that serve as discriminating variables are given in the slot `ed_lda$scaling` of the output from `lda`.

```
ec_lda = lda(Embryonic.day ~ Fn1 + Timd2 + Gata4 + Sox7,
             data = embryoCells)
```

```
round(ec_lda$scaling, 1)

##         LD1  LD2
## Fn1   -0.2 -0.4
## Timd2  0.5  0.0
## Gata4 -0.1 -0.6
## Sox7  -0.7  0.5
```

For the visualisation of the learned model in Figure 13.9, we need to build the prediction regions and their boundaries by expanding the grid in the space of the two new coordinates LD1 and LD2.

```
ec_rot = predict(ec_lda)$x %>% as_data_frame %>%
           mutate(ed = embryoCells$Embryonic.day)

ec_lda2 = lda(ec_rot[, 1:2], predict(ec_lda)$class)

ec_grid = with(ec_rot, expand.grid(
  LD1 = make1Dgrid(LD1),
  LD2 = make1Dgrid(LD2)))

ec_grid$edhat = predict(ec_lda2, newdata = ec_grid)$class

ggplot() +
  geom_point(aes(x = LD1, y = LD2, colour = ed), data = ec_rot) +
  geom_raster(aes(x = LD1, y = LD2, fill = edhat),
            data = ec_grid, alpha = 0.4, interpolate = TRUE) +
  scale_x_continuous(expand = c(0, 0)) +
  scale_y_continuous(expand = c(0, 0)) +
  coord_fixed()
```

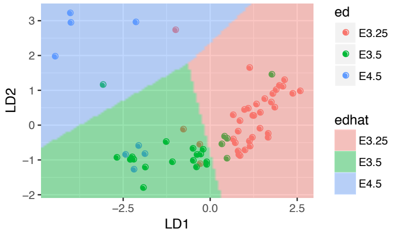

Figure 13.9: LDA classification regions for Embryonic.day.

▶ Question 13.3.7. Repeat these analyses using quadratic discriminant analysis (qda). What difference do you see in the shape of the boundaries?

▶ Answer 13.3.4. See Figure 13.10.

```
library("gridExtra")

ec_qda = qda(Embryonic.day ~ Fn1 + Timd2 + Gata4 + Sox7,
             data = embryoCells)

variables = colnames(ec_qda$means)
pairs = combn(variables, 2)
lapply(seq_len(ncol(pairs)), function(i) {
  grid = with(embryoCells,
    expand.grid(x = make1Dgrid(get(pairs[1, i])),
                y = make1Dgrid(get(pairs[2, i])))) %>%
    ‘colnames<-‘(pairs[, i])

  for (v in setdiff(variables, pairs[, i]))
    grid[[v]] = median(embryoCells[[v]])

  grid$edhat = predict(ec_qda, newdata = grid)$class
  ggplot() + geom_point(
      aes_string(x = pairs[1, i], y = pairs[2, i],
      colour = "Embryonic.day"), data = embryoCells) +
    geom_raster(
      aes_string(x = pairs[1, i], y = pairs[2, i], fill = "edhat"),
      data = grid, alpha = 0.4, interpolate = TRUE) +
    scale_x_continuous(expand = c(0, 0)) +
    scale_y_continuous(expand = c(0, 0)) +
    coord_fixed() +
    if (i != ncol(pairs)) theme(legend.position = "none")
}) %>% grid.arrange(grobs = ., ncol = 3)
```

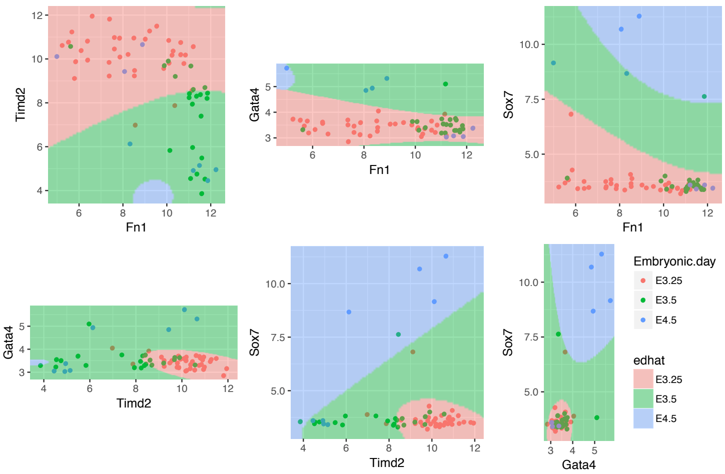

Figure 13.10: QDA for the mouse cell data, all pairwise plots of the four features.

▶ Question 13.3.8. What happens if you call lda or qda with a lot more genes, say the first 1000, in the Hiiragi dataset?

▶ Answer 13.3.5.

```
lda(t(exprs(x))[, 1:1000], x$Embryonic.day)

## Warning in lda.default(x, grouping, ...): variables are collinear

qda(t(exprs(x))[, 1:1000], x$Embryonic.day)

## Error in qda.default(x, grouping, ...): some group is too small for
'qda'
```

13.4 Machine learning vs rote learning

Computers are really good at memorizing facts. In the worst case, a machine learning algorithm is a roundabout way of doing this[6]. The central question in statistical learning is whether the algorithm was able to generalize, i. e., interpolate and extrapolate. Let's look at the following example. We generate random data (rnorm) for n objects, with different numbers of features (given by p). We train a LDA on these data and compute the **misclassification rate**, i. e., the fraction of times the prediction is wrong (pred != resp).

[6] The not so roundabout way is database technologies.

```
library("dplyr")
p = 2:21
n = 20

mcl = lapply(p, function(k) {
  replicate(100, {
    xmat = matrix(rnorm(n * k), nrow = n)
    resp = sample(c("apple", "orange"), n, replace = TRUE)
    fit  = lda(xmat[, 1:k], resp)
    pred = predict(fit)$class
    mean(pred != resp)
  }) %>% mean %>% tibble(mcl = .)
}) %>% bind_rows %>% cbind(., p = p)

ggplot(mcl, aes(x = p, y = mcl)) + geom_line() + geom_point() +
  ylab("Misclassification rate")
```

▶ Question **13.4.1.** What is the purpose of the `replicate` loop in the above code? What happens if you omit it (or replace the 100 by 1)?

▶ Answer **13.4.1.** Averaging the misclassification rate over 100 replicates makes the estimate more stable, and since we are working with simulated data, we are at liberty to do so. For each single replicate, the curve is a noisier version of Figure 13.11.

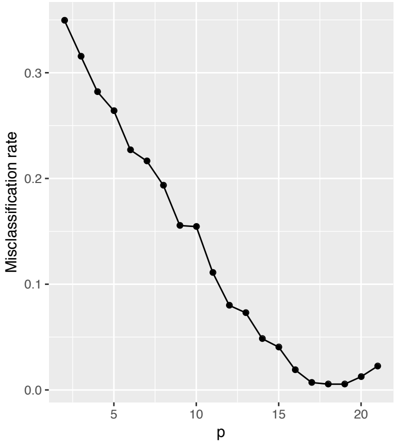

Figure 13.11: Misclassification rate of LDA applied to random data. With increasing number of features (`p`), the misclassification rate becomes almost zero as `p` approaches `n`, the number of objects. (As `p` becomes even larger, the "performance" degrades again, apparently due to numerical properties of the `lda` implementation used here.)

Figure 13.11 seems to imply that we can perfectly predict random labels from random data, if we only fit a complex enough model, i.e., one with many parameters. How can we overcome such an absurd conclusion? The problem with the above code is that the model performance is evaluated on the same data on which it was trained. This generally leads to positive bias, as you see in this crass example. How can we overcome this problem? The key idea is to assess model performance on different data than those on which the model was trained.

13.4.1 Cross-validation

A naive approach might be to split the data in two halves, and use the first half for learning ("training"), the second half for assessment ("testing"). It turns out that this is needlessly variable and needlessly inefficient. Needlessly variable, since by splitting the data only once, our results can be quite affected by how the splitting happens to fall. It seems better to do the splitting many times, and average. This will give us more stable results. Needlessly inefficient, since the performance of machine learning algorithms depends on the number of samples, and the performance measured on half the data is likely[7] to be worse than what it is with all the data. For this reason, it is better to use unequal sizes of training and test data. In the extreme case, we'll use as much as $n-1$ samples for training, and the remaining one for testing. After we've done this likewise for all samples, we can average our performance metric. This is called **leave-one-out cross-validation**. An alternative is k-**fold cross-validation**, where the samples are repeatedly split into a training set of size of around $n(k-1)/k$ and a test set of size of around n/k. Both alternatives have pros and contras, and there is not a universally best choice. An advantage of leave-one-out is that the amount of data used for training is close to the maximally available data; this is

[7] Unless we have such an excess of data that it doesn't matter.

especially important if the sample size is limiting and "every little matters" for the algorithm. A drawback of leave-one-out is that the training sets are all very similar, so they may not sufficiently model the kind of sampling changes to be expected if a new dataset came along. For large n, leave-one-out cross-validation can be needlessly time-consuming[8].

```
estimate_mcl_loocv = function(x, resp) {
  vapply(seq_len(nrow(x)), function(i) {
    fit  = lda(x[-i, ], resp[-i])
    ptrn = predict(fit, newdata = x[-i,, drop = FALSE])$class
    ptst = predict(fit, newdata = x[ i,, drop = FALSE])$class
    c(train = mean(ptrn != resp[-i]), test = (ptst != resp[i]))
  }, FUN.VALUE = c(0,0)) %>% rowMeans %>% t %>% as_data_frame
}

xmat = matrix(rnorm(n * last(p)), nrow = n)
resp = sample(c("apple", "orange"), n, replace = TRUE)

mcl = lapply(p, function(k) {
  estimate_mcl_loocv(xmat[, 1:k], resp)
}) %>% bind_rows %>% data.frame(p) %>% melt(id.var = "p")

ggplot(mcl, aes(x = p, y = value, col = variable)) + geom_line() +
  geom_point() + ylab("Misclassification rate")
```

The result is show in Figure 13.12.

▶ Question 13.4.2. Why are the curves in Figure 13.12 more variable ("wiggly") than in Figure 13.11? How can you overcome this?

▶ Answer 13.4.2. Only one dataset (`xmat`, `resp`) was used to calculate Figure 13.12, whereas for Figure 13.11, we had the data generated within a `replicate` loop. You could similarly extend the above code to average the misclassification rate curves over many replicate datasets.

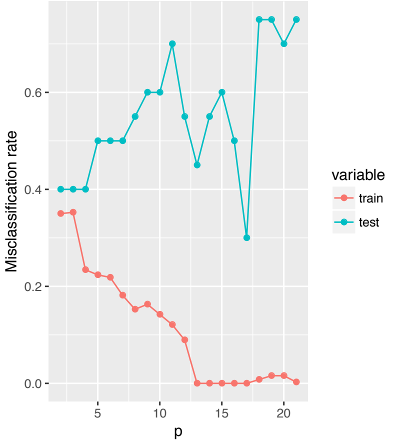

Figure 13.12: Cross-validation: the misclassification rate of LDA applied to random data, when evaluated on test data that were not used for learning, hovers around 0.5 independent of p. The misclassification rate on the training data is also shown. It behaves similar to what we already saw in Figure 13.11.

13.4.2 The curse of dimensionality

In Section 13.4.1 we have seen overfitting and cross-validation on random data, but how does it look if there is in fact a relevant class separation?

```
p   = 2:20
mcl = replicate(100, {
  xmat = matrix(rnorm(n * last(p)), nrow = n)
  resp = sample(c("apple", "orange"), n, replace = TRUE)
  xmat[, 1:6] = xmat[, 1:6] + as.integer(factor(resp))

  lapply(p, function(k) {
    estimate_mcl_loocv(xmat[, 1:k], resp)
  }) %>% bind_rows %>% cbind(p = p) %>% melt(id.var = "p")
}, simplify = FALSE)

mcl = bind_rows(mcl) %>% group_by(p, variable) %>%
```

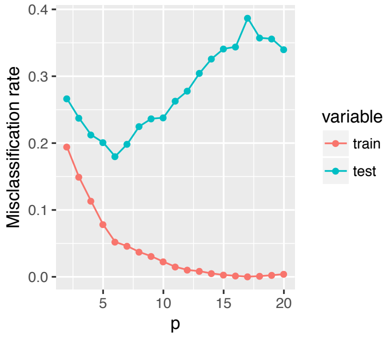

Figure 13.13: As we increase the number of features included in the model, the misclassification rate initially improves; as we start including more and more irrelevant features, it increases again, as we are fitting noise.

[8] See Chapter *Model Assessment and Selection* in the book by Hastie et al. (2008) for further discussion on these trade-offs.

```
    summarise(value = mean(value))

ggplot(mcl, aes(x = p, y = value, col = variable)) + geom_line() +
    geom_point() + ylab("Misclassification rate")
```

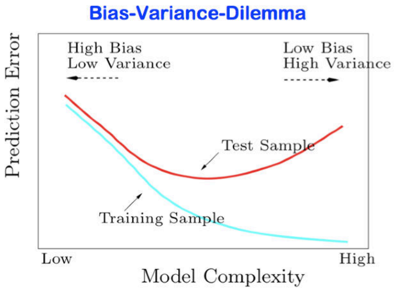

Figure 13.14: Idealized version of Figure 13.13, from Hastie et al. (2008). A recurrent goal in machine learning is finding the sweet spot in the variance–bias trade-off.

The result is shown in Figure 13.13. The group centers are the vectors (in $\mathbb{R}^{20}$) given by the coordinates $(1, 1, 1, 1, 1, 1, 0, 0, 0, \ldots)$ (apples) and $(2, 2, 2, 2, 2, 2, 0, 0, 0, \ldots)$ (oranges), and the optimal decision boundary is the hyperplane orthogonal to the line between them. For k smaller than 6, the decision rule cannot reach this hyperplane – it is biased. As a result, the misclassification rate is suboptimal, and it decreases with k. But what happens for k larger than 6? The algorithm is, in principle, able to model the optimal hyperplane, and it should not be distracted by the additional features. The problem is that it is. The more additional features enter the dataset, the higher the probability that one or more of them happen to fall in a way that they *look like* good, discriminating features in the training data – only to mislead the classifier and degrade its performance in the test data. Shortly we'll see how to use penalization to (try to) control this problem.

The term **curse of dimensionality** was coined by Bellman (1961). It refers to the fact that high-dimensional spaces are very hard to sample. Our intuitions about distances between points in a high-dimensionsal space, and the relationship between its volume and surface, break down.

▶ Question **13.4.3.** Assume you have a dataset with 1 000 000 data points in p dimensions. The data are uniformly distributed in the unit hybercube (i. e., all features lie in the interval $[0, 1]$). What's the side length of a hybercube that can be expected to contain 10 points, as a function of p?

▶ Answer **13.4.3.** See Figure 13.15.

```
sideLength = function(p, pointDensity = 1e6, pointsNeeded = 10)
    (pointsNeeded / pointDensity) ^ (1 / p)
ggplot(tibble(p = 1:750, sideLength = sideLength(p)),
         aes(x = p, y = sideLength)) +
  geom_line(col = "red") + geom_hline(aes(yintercept = 1), linetype = 2)
```

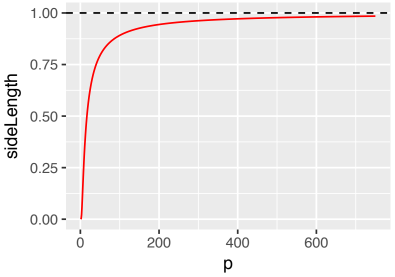

Figure 13.15: Side length of a hybercube expected to contain 10 points out of 1 million uniformly distributed ones, as a function of its dimension p. While for $p = 1$, this length is $10/10^6 = 10^{-5}$, for larger p it approaches 1, i. e., becomes the same as the range of each the features. In genomics, we often aim to fit models to data with thousands of features.

Generally, prediction at the boundaries of feature space is more difficult than in its interior, as it tends to involve extrapolation, rather than interpolation.

▶ Question **13.4.4.** What fraction of a unit cube's total volume is closer than 0.01 to any of its surfaces, as a function of the dimension?

▶ Answer **13.4.4.** See Figure 13.16.

```
p = 1:750
volOuterCube = 1 ^ p
volInnerCube = 0.98 ^ p
ggplot(tibble(p = p, ‘V(shell)‘ = volOuterCube - volInnerCube),
         aes(x = p, y =‘V(shell)‘)) + geom_line(col = "blue")
```

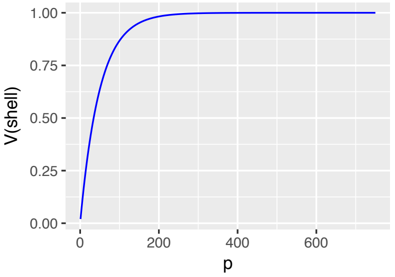

Figure 13.16: Fraction of a unit cube's total volume that is in its "shell" (here operationalised as those points that are closer than 0.01 to its surface) as a function of the dimension p.

▶ Question **13.4.5.** What is the coefficient of variation (ratio of standard deviation over average) of the distance between two randomly picked points in the unit hypercube, as a function of the dimension?

▶ Answer **13.4.5.** We solve this one by simulation. We generate n pairs of random points in the hypercube (x1, x2) and compute their Euclidean distances. See Figure 13.17. This result can also be predicted from the central limit theorem.

```
n = 1000
df = tibble(
  p = round(10 ^ seq(0, 4, by = 0.25)),
  cv = vapply(p, function(k) {
    x1 = matrix(runif(k * n), nrow = n)
    x2 = matrix(runif(k * n), nrow = n)
    d = sqrt(rowSums((x1 - x2)^2))
    sd(d) / mean(d)
  }, FUN.VALUE = NA_real_))
ggplot(df, aes(x = log10(p), y = cv)) + geom_line(col = "orange") +
  geom_point()
```

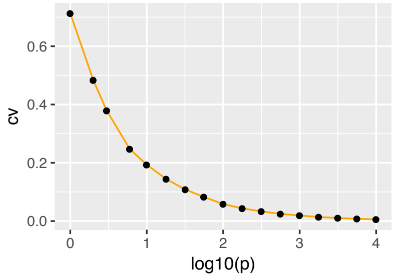

Figure 13.17: Coefficient of variation (CV) of the distance between randomly picked points in the unit hypercube, as a function of the dimension. As the dimension increases, everybody is equally far away from everyone else: there is almost no variation in the distances any more.

13.5 Objective functions

We've already seen the **misclassification rate** (MCR) used to assess our classification performance in Figures 13.11–13.13. Its population version is defined as

$$\text{MCR} = \text{E}\left[\mathbb{1}_{\hat{y}\neq y}\right], \tag{13.1}$$

and for a finite sample

$$\widehat{\text{MCR}} = \frac{1}{n}\sum_{i=1}^{n}\mathbb{1}_{\hat{y}_i\neq y_i}. \tag{13.2}$$

This is not the only choice we could make. Perhaps we care more about the misclassification of apples as oranges than vice versa, and we can reflect this by introducing weights that depend on the type of error made into the sum of Equation (13.2) (or the integral of Equation (13.1)). This can get even more elaborate if we have more than two classes. Often we do not only want to see a single numeric summary, but the whole **confusion table**, which in *R* we can get via expressions like

```
table(truth, response)
```

An important special case is binary classification with asymmetric costs – think about, say, a medical test. Here, the **sensitivity** (a.k.a. **true positive rate** or **recall**) is related to the misclassification of non-sick as sick, and the **specificity** (or **true negative rate**) depends on the probability of misclassification of sick as non-sick. Often, there is a single parameter (e. g., a threshold) that can be moved up and down, allowing a trade-off between sensitivity and specificity (and thus, equivalently, between the two types of misclassification). In those cases, we usually are not content to know the classifier performance at one single choice of threshold, but at many (or all) of them. This leads to **receiver operating characteristic** (ROC) or **precision-recall** curves.

▶ Question **13.5.1.** What are the exact relationships between the per-class misclassification rates and sensitivity and specificity?

► Answer **13.5.1.** The sensitivity or true positive rate is

$$\text{TPR} = \frac{\text{TP}}{\text{P}},$$

where TP is the number of true positives and P the number of all positives. The specificity or true negative rate is

$$\text{SPC} = \frac{\text{TN}}{\text{N}},$$

where TN is the number of true negatives and N the number of all negatives. See also `https://en.wikipedia.org/wiki/Sensitivity_and_specificity`

Another cost function can be computed from the **Jaccard index**, which we already saw in Chapter 5.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}, \tag{13.3}$$

where A is the set of samples for which the true class is 1 ($A = \{i \,|\, y_i = 1\}$) and B is the set of samples for which the predicted class is 1. J is a number between 0 and 1, and a high value of J indicates high overlap of the two sets. Note that J does not depend on the number of samples for which both true and predicted class is 0 – so it is particularly suitable for measuring the performance of methods that try to find rare events.

We can also consider probabilistic class predictions, which come in the form $\hat{P}(Y \,|\, X)$. In this case, a possible risk function would be obtained by looking at distances between the true probability distribution and the estimated probability distributions. For two classes, the finite sample version of the log loss is

$$\text{log loss} = -\frac{1}{n}\sum_{i=1}^{n} y_i \log(\hat{p}_i) + (1-y_i)\log(1-\hat{p}_i), \tag{13.4}$$

where $\hat{p}_i \in [0,1]$ is the prediction, and $y_i \in \{0,1\}$ is the truth[9].

For continuous continuous response variables (regression), a natural choice is the **mean squared error** (MSE). It is the average squared error,

$$\widehat{\text{MSE}} = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2. \tag{13.5}$$

The population version is defined analogously, by turning the summation into an integral as in Equations (13.1) and (13.2).

Statisticians call functions like Equations (13.1–13.5) variously (and depending on context and predisposition) **risk function**, **cost function**, **objective function**[10].

[9] Note that the log loss will be infinite if a prediction is totally confident ($\hat{p}_i$ is exactly 0 or 1) but wrong.

[10] There is even an *R* package dedicated to evaluation of statistical learners called *metrics*.

13.6 Variance–bias trade-off

An important fact that helps us understand the tradeoffs when picking a statistical learning model is that the MSE is the sum of two terms, and often the choices we can make are such that one of those terms goes down while the other one goes up. The bias measures how different the average of all the different estimates is from the truth, and variance, how much an individual one might scatter from the average value (Figure 13.18). In applications, we often only get one shot, therefore being reliably almost on target can beat being right on the long term average but really off today. The decomposition

$$\mathrm{MSE} = \underbrace{\mathrm{Var}(\hat{Y})}_{\text{variance}} + \underbrace{\mathbb{E}[\hat{Y} - Y]^2}_{\text{bias}} \tag{13.6}$$

follows by straightforward algebra.

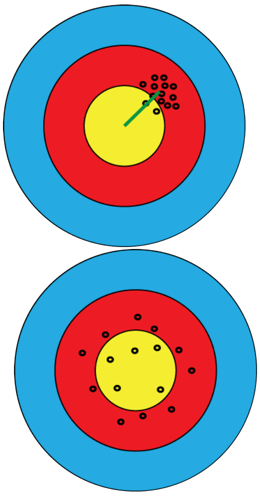

Figure 13.18: In the upper bull's eye, the estimates are systematically off target, but in a quite reproducible manner. The green segment represents the bias. In the lower bull's eye, the estimates are not biased, as they are centered in the right place, however they have high variance. We can distinguish the two scenarios since we see the result from many shots. If we only had one shot and missed the bull's eye, we could not easily know whether that's because of bias or variance.

When trying to minimize the MSE, it is important to remember that sometimes we can pay the price of some bias to obtain a much smaller variance and thus an overall estimator of lower MSE. In classification (with categorical response variables), different objective functions than the MSE are used, and there is usually no such straightforward decomposition as in Equation (13.6). In general, we can go much further in classification applications than in regression with trading biases for variance, since the discreteness of the response neutralizes certain biases (Friedman, 1997).

13.6.1 Penalization

In high-dimensional statistics, we are constantly plagued by variance: there is just not enough data to fit all the possible parameters. One of the most fruitful ideas in high-dimensional statistics is **penalization**: a tool to actively control and exploit the variance-bias tradeoff.

Although generalisation of LDA to high-dimensional settings is possible (Clemmensen et al., 2011; Witten and Tibshirani, 2011), it turns out that logistic regression is a more general approach[11], and therefore we'll now switch to that, using the *glmnet* package.

Multinomial[12] logistic regression models the posterior log-odds between k classes and can be written in the form[13]

$$\log \frac{P(Y = i \,|\, X = x)}{P(Y = k \,|\, X = x)} = \beta_i^0 + \beta_i x, \tag{13.7}$$

where $i = 1, \ldots, k-1$; x is the $n \times p$ data matrix (n: number of samples, p: number of features), and β_i is a p-dimensional vector that determines how the classification odds for class i versus class k depend on x. The numbers β_i^0 are intercepts and depend, among other things, on the classes' prior probabilities. Instead of the log odds (13.7) (i. e., ratios of class probabilities), we can also write down an equivalent model for the class probabilities themselves, and the fact that we here used the k-th

[11] It fits into the framework of generalized linear models.

[12] Or, for the special case of two classes, binomial logistic regression.

[13] See (Hastie et al., 2008) for a complete presentation.

class as a reference is an arbitrary choice, as the model estimates are equivariant under this choice (Hastie et al., 2008). The model is fit by maximising the log-likelihood $\ell(\beta, \beta^0; x)$, where $\beta = (\beta_1, \dots, \beta_{k-1})$ and analogously for β^0.

So far, so good. But as p gets larger, there is an increasing chance that some of the estimates go wildly off the mark, due to random sampling happenstances in the data. This is true even if for each individual coordinate of the vector β_i, the error distribution is bounded: the probabilty of there being one coordinate that is in the far tails increases the more coordiates there are, i.e., the larger p is.

A related problem can also occur, not in (13.7), but in other, non-linear models, as the model dimension p increases while the sample size n remains the same: the likelihood landscape around its maximum becomes increasingly flat, and the maximum-likelihood estimate of the model parameters becomes more and more variable. Eventually, the maximum is no longer a point, but a submanifold, and the maximum likelihood estimate is unidentifiable.

Both of these limitations can be overcome with a modification of the objective: instead of maximising the bare log-likelihood, we maximise a penalized version of it,

$$\hat{\beta} = \arg\max_{\beta} \ell(\beta, \beta^0; x) + \lambda \operatorname{pen}(\beta), \qquad (13.8)$$

where $\lambda \geq 0$ is a real number, and pen is a convex function, called the **penalty function**. Popular choices are $\operatorname{pen}(\beta) = |\beta|^2$ (**ridge regression**) and $\operatorname{pen}(\beta) = |\beta|^1$ (**lasso**)[14]. In the **elastic net**, ridge and lasso are hybridized by using the penalty function $\operatorname{pen}(\beta) = (1-\alpha)|\beta|^1 + \alpha|\beta|^2$ with some further parameter $\alpha \in [0, 1]$. The crux is, of course, how to choose the right λ, and we will discuss that in the following.

13.6.2 Example: predicting colon cancer from stool microbiome composition

Zeller et al. (2014) studied metagenome sequencing data from fecal samples of 156 humans that included colorectal cancer patients and tumor-free controls. Their aim was to see whether they could identify biomarkers (presence or abundance of certain taxa) that could help with early tumor detection. The data are available from Bioconductor through its **ExperimentHub** service under the identifier EH359.

```r
library("ExperimentHub")
eh = ExperimentHub()
zeller = eh[["EH361"]]
```

```r
zeller$disease %>% table

## .
##        cancer large_adenoma             n small_adenoma
##            53            15            61            27
```

▶ Question **13.6.1.** Explore the `eh` object to see what other datasets there are.
▶ Answer **13.6.1.**

[14] Here, $|\beta|^{\nu} = \sum_i \beta_i^{\nu}$ is the L_{ν}-norm of the vector β. Variations are possible, for instead we could include in this summation only some but not all of the elements of β; or we could scale different elements differently, for instance based on some prior belief of their scale and importance.

```
eh
```

For the following, let's focus on the normal and cancer samples and set the adenomas aside.

```
zellerNC = zeller[, zeller$disease %in% c("n", "cancer")]
```

Before jumping into model fitting, it is always a good idea to do some exploration of the data. First, let's look at the sample annotations for some of the samples. We pick them randomly, since this can be more representative of the whole dataset than only looking at the first or last ones.

```
pData(zellerNC)[ sample(ncol(zellerNC), 3), ]

##                     subjectID age gender bmi country disease
## CCIS71578391ST-4-0     FR-187  70   male  25  france       n
## CCIS50003399ST-4-0     FR-194  66 female  28  france       n
## CCIS38765456ST-20-0    FR-723  79 female  22  france  cancer
##                     tnm_stage ajcc_stage localization     fobt
## CCIS71578391ST-4-0       <NA>       <NA>         <NA> negative
## CCIS50003399ST-4-0       <NA>       <NA>         <NA> negative
## CCIS38765456ST-20-0    t4n1m1         iv           lc positive
##                     wif-1_gene_methylation_test   group bodysite
## CCIS71578391ST-4-0                    negative control    stool
## CCIS50003399ST-4-0                    negative control    stool
## CCIS38765456ST-20-0                   positive     crc    stool
##                     ethnicity number_reads
## CCIS71578391ST-4-0      white     74021867
## CCIS50003399ST-4-0      white     63416533
## CCIS38765456ST-20-0     white     81682982
```

Next, let's explore the feature names[15].

```
formatfn = function(x)
   gsub("|", "| ", x, fixed = TRUE) %>% lapply(strwrap)

rownames(zellerNC)[1:4]

## [1] "k__Bacteria"                  "k__Viruses"
## [3] "k__Bacteria|p__Firmicutes"    "k__Bacteria|p__Bacteroidetes"

rownames(zellerNC)[nrow(zellerNC) + (-2:0)] %>% formatfn

## [[1]]
## [1] "k__Bacteria| p__Proteobacteria| c__Deltaproteobacteria|"
## [2] "o__Desulfovibrionales| f__Desulfovibrionaceae|"
## [3] "g__Desulfovibrio| s__Desulfovibrio_termitidis"
##
## [[2]]
## [1] "k__Viruses| p__Viruses_noname| c__Viruses_noname|"
## [2] "o__Viruses_noname| f__Baculoviridae| g__Alphabaculovirus|"
## [3] "s__Bombyx_mori_nucleopolyhedrovirus|"
## [4] "t__Bombyx_mori_nucleopolyhedrovirus_unclassified"
##
## [[3]]
```

[15] We define the helper function `formatfn` to line wrap these long character strings for the available space here.

```
## [1] "k__Bacteria| p__Proteobacteria| c__Deltaproteobacteria|"
## [2] "o__Desulfovibrionales| f__Desulfovibrionaceae|"
## [3] "g__Desulfovibrio| s__Desulfovibrio_termitidis|"
## [4] "t__GCF_000504305"
```

As you can see, the features are a mixture of abundance quantifications at different taxonomic levels, from **k**ingdom over **phylum** to **s**pecies. We could select only some of these, but here we continue with all of them. Next, let's look at the distribution of some of the features. Here, we show two; in practice, it is helpful to scroll through many such plots quickly to get an impression.

```
ggplot(melt(exprs(zellerNC)[c(510, 527), ]), aes(x = value)) +
    geom_histogram(bins = 25) +
    facet_wrap( ~ Var1, ncol = 1, scales = "free")
```

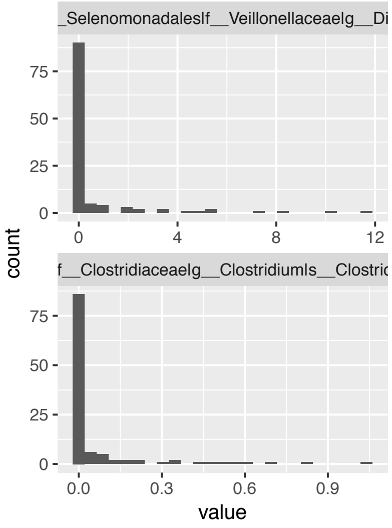

Figure 13.19: Histograms of the distributions for two randomly selected features. The distributions are highly skewed, with many zero values and a thin, long tail of non-zero values.

In the simplest case, we fit model (13.7) as follows.

```
library("glmnet")
glmfit = glmnet(x = t(exprs(zellerNC)),
                y = factor(zellerNC$disease),
                family = "binomial")
```

A remarkable feature of the `glmnet` function is that it fits (13.7) not only for one choice of λ, but for all possible λs at once. For now, let's look at the prediction performance for, say, $\lambda = 0.04$. The name of the function parameter is `s`:

```
predTrsf = predict(glmfit, newx = t(exprs(zellerNC)),
                   type = "class", s = 0.04)
table(predTrsf, zellerNC$disease)

##
## predTrsf cancer  n
##    cancer     51  0
##    n           2 61
```

Not bad[16]. Let's have a closer look at `glmfit`. The *glmnet* package offers a a diagnostic plot that is worth looking at (Figure 13.20).

```
plot(glmfit, col = brewer.pal(12, "Set3"), lwd = sqrt(3))
```

▶ Question **13.6.2.** What is the x-axis in Figure 13.20? What are the different lines?

▶ Answer **13.6.2.** Consult the manual page of the function `plot.glmnet` in the *glmnet* package.

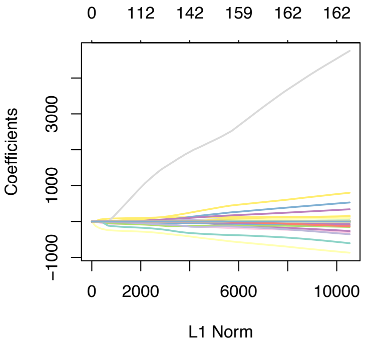

Figure 13.20: Regularization paths for `glmfit`.

Let's get back to the question of how to choose the parameter λ. We could try many different choices –and indeed, all possible choices– of λ, assess classification performance in each case using cross-validation, and then choose the best λ[17]. We could do so by writing a loop as we did in the `estimate_mcl_loocv` function in Section 13.4.1. It turns out that the *glmnet* package already has built-in functionality for that, with the function `cv.glmnet`, which we can use instead.

```
cvglmfit = cv.glmnet(x = t(exprs(zellerNC)),
                     y = factor(zellerNC$disease),
                     family = "binomial")
plot(cvglmfit)
```

[16] But remember that this is on the training data, without cross-validation.

[17] You'll already realize from the description of this strategy that if we optimize λ in this way, the resulting apparent classification performance will likely be exaggerated. We need a truly independent dataset, or at least another, outer cross-validation loop to get a more realistic impression of the generalizability. We will get back to this question at the end of the chapter.

The diagnostic plot is shown in Figure 13.21. We can access the optimal value with

```
cvglmfit$lambda.min

## [1] 0.08830775
```

As this value results from finding a minimum in an estimated curve, it turns out that it is often too small, i. e., that the implied penalization is too weak. A heuristic recommended by the authors of the *glmnet* package is to use a somewhat larger value instead, namely the largest value of λ such that the performance measure is within 1 standard error of the minimum.

```
cvglmfit$lambda.1se

## [1] 0.1015325
```

▶ Question **13.6.3.** How does the confusion table look like for λ = `lambda.1se`?

▶ Answer **13.6.3.**

```
s0 = cvglmfit$lambda.1se
predict(glmfit, newx = t(exprs(zellerNC)),type = "class", s = s0) %>%
    table(zellerNC$disease)

##
## .         cancer  n
##   cancer      35  7
##   n           18 54
```

▶ Question **13.6.4.** What features drive the classification?

▶ Answer **13.6.4.**

```
coefs = coef(glmfit)[, which.min(abs(glmfit$lambda - s0))]
topthree = order(abs(coefs), decreasing = TRUE)[1:3]
as.vector(coefs[topthree])

## [1] -28.629194  -4.486355  -1.095961

formatfn(names(coefs)[topthree])

## [[1]]
## [1] "k__Bacteria| p__Candidatus_Saccharibacteria|"
## [2] "c__Candidatus_Saccharibacteria_noname|"
## [3] "o__Candidatus_Saccharibacteria_noname|"
## [4] "f__Candidatus_Saccharibacteria_noname|"
## [5] "g__Candidatus_Saccharibacteria_noname|"
## [6] "s__candidate_division_TM7_single_cell_isolate_TM7b"
##
## [[2]]
## [1] "k__Bacteria| p__Firmicutes| c__Clostridia| o__Clostridiales|"
## [2] "f__Ruminococcaceae| g__Subdoligranulum|"
## [3] "s__Subdoligranulum_variabile"
##
## [[3]]
## [1] "k__Bacteria| p__Firmicutes| c__Clostridia| o__Clostridiales|"
## [2] "f__Lachnospiraceae| g__Lachnospiraceae_noname|"
## [3] "s__Lachnospiraceae_bacterium_7_1_58FAA"
```

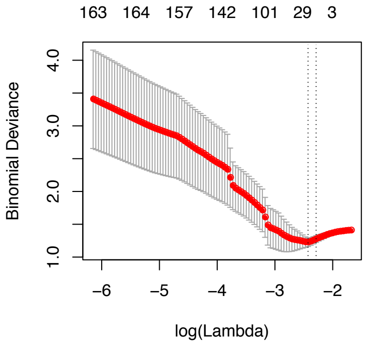

Figure 13.21: Diagnostic plot for `cv.glmnet`: shown is a measure of cross-validated prediction performance, the deviance, as a function of λ. The dashed vertical lines show `lambda.min` and `lambda.1se`.

▶ Question **13.6.5.** How do the results change if we transform the data, say, with the `asinh` transformation as we saw in Chapter 5?

▶ Answer **13.6.5.** See Figure 13.22.

```
cv.glmnet(x = t(asinh(exprs(zellerNC))),
          y = factor(zellerNC$disease),
          family = "binomial") %>% plot
```

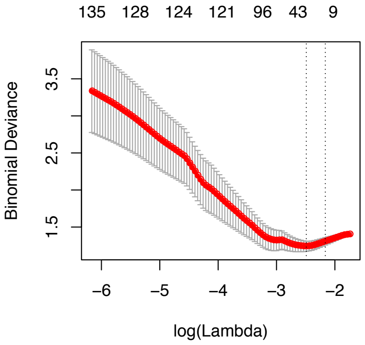

Figure 13.22: like Figure 13.21, but using an asinh transformation of the data.

▶ Question **13.6.6.** Would a good classification performance on these data mean that this assay is ready for screening and early cancer detection?

▶ Answer **13.6.6.** No. The performance here is measured on a set of samples in which the cases have similar prevalence as the controls. This serves well enough to explore the biology. However, in a real-life application, the cases will be much less frequent. To be practically useful, the assay must have a much higher specificity, i. e., not wrongly diagnose disease where there is none. To establish specificity, a much larger set of normal samples need to be tested.

13.6.3 Example: classifying mouse cells from their expression profiles

Figures 13.21 and 13.22 are textbook examples of how we expect the dependence of (cross-validated) classification performance versus model complexity (λ) to look. Now let's get back to the mouse embryo cells data. We'll try to classify the cells from embryonic day `E3.25` with respect to their genotype.

```
sx = x[, x$Embryonic.day == "E3.25"]
embryoCellsClassifier = cv.glmnet(t(exprs(sx)), sx$genotype,
                        family = "binomial", type.measure = "class")
plot(embryoCellsClassifier)
```

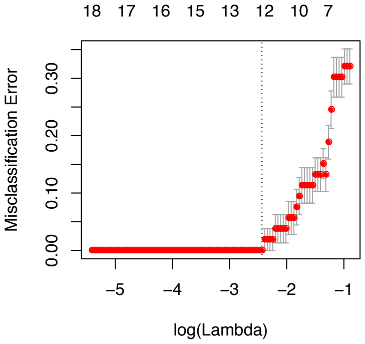

Figure 13.23: Cross-validated misclassification error versus penalty parameter for the mouse cells data.

In Figure 13.23 we see that the misclassification error is (essentially) monotonously increasing with λ, and is smallest for $\lambda \to 0$, i. e., if we apply no penalization at all.

▶ Question **13.6.7.** What is going on with these data?

▶ Answer **13.6.7.** It looks that inclusion of more, and even of all features, does not harm the classification performance. In a way, these data are "too easy". Let's do a t-test for all features:

```
mouse_de = rowttests(sx, "genotype")
ggplot(mouse_de, aes(x = p.value)) +
  geom_histogram(boundary = 0, breaks = seq(0, 1, by = 0.01))
```

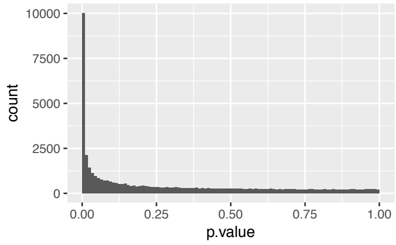

Figure 13.24: Histogram of p-values for the per-feature t-tests between genotypes in the E3.25 samples.

The result, shown in Figure 13.24, shows that large number of genes are differentially expressed, and thus informative for the class distinction. We can also compute the pairwise distances between all samples, using all features.

```
dists = as.matrix(dist(scale(t(exprs(x)))))
diag(dists) = +Inf
```

and then for each sample determine the class of its nearest neighbor

```
nn = sapply(seq_len(ncol(dists)), function(i) which.min(dists[, i]))
table(x$sampleGroup, x$sampleGroup[nn]) %>% `colnames<-`(NULL)
```

```
##
##                  [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
##   E3.25            33    0    0    0    3    0    0    0
##   E3.25 (FGF4-KO)   1   15    0    1    0    0    0    0
##   E3.5 (EPI)        2    0    3    0    6    0    0    0
##   E3.5 (FGF4-KO)    0    0    0    8    0    0    0    0
##   E3.5 (PE)         0    0    0    0   11    0    0    0
##   E4.5 (EPI)        0    0    0    0    2    2    0    0
##   E4.5 (FGF4-KO)    1    0    0    0    0    0    9    0
##   E4.5 (PE)         0    0    0    0    2    0    0    2
```

Using all features, the nearest neighbour classifier is correct in almost all cases, including for the E3.25 wildtype vs FGF4-KO distinction. This means that for these data, there is no apparent benefit in regularisation or feature selection. Limitations of using all features might become apparent with truly new data, but that is out of reach for cross-validation.

13.7 A large choice of methods

We have now seen three classification methods: linear discriminant analysis (`lda`), quadratic discriminant analysis (`qda`) and the elastic net (`glmnet`). In fact, there are hundreds of different learning algorithms[18] available in R and its add-on packages. You can get an overview in the CRAN task view Machine Learning & Statistical Learning. Some examples are:

- Support vector machines: the function `svm` in the package *e1071*; `ksvm` in *kernlab*
- Tree based methods in the packages *rpart*, *tree*, *randomForest*
- Boosting methods: the functions `glmboost` and `gamboost` in package *mboost*
- `PenalizedLDA` in the package *PenalizedLDA*, `dudi.discr` and `dist.pcaiv` in *ade4*).

The complexity and heterogeneity of choices of learning strategies, tuning parameters and evaluation criteria in each of these packages can be confusing. You will already have noted differences in the interfaces of the `lda`, `qda` and `glmnet` functions, i. e., in how they expect their input data to presented and what they return. There is even greater diversity across all the other packages and functions. At the same time, there are common tasks such as cross-validation, parameter tuning and performance assessment that are more or less the same no matter what specific method is used. As you have seen, e. g., in our `estimate_mcl_loocv` function, the looping and data shuffling involved leads to rather verbose code.

So what to do if you want to try out and explore different learning algorithms? Fortunately, there are several projects that provide unified interfaces to the large number of different machine learning interfaces in R, and also try to provide “best practice” implementations of the common tasks such as parameter tuning and performance assessment. The two most well-known ones are the packages *caret* and *mlr*.

[18] For an introduction to the subject that uses R and provides many examples and exercises, we recommend (James et al., 2013).

Here were have a look at *caret*. You can get a list of supported methods through its `getModelInfo` function. There are quite a few, here we just show the first 8.

```
library("caret")
caretMethods = names(getModelInfo())
head(caretMethods, 8)

## [1] "ada"          "AdaBag"       "AdaBoost.M1" "adaboost"
## [5] "amdai"        "ANFIS"        "avNNet"       "awnb"

length(caretMethods)

## [1] 232
```

We will check out a neural network method, the `nnet` function from the eponymous package. The `parameter` slot informs us on the the available tuning parameters[19].

```
getModelInfo("nnet", regex = FALSE)[[1]]$parameter

##   parameter   class         label
## 1      size numeric #Hidden Units
## 2     decay numeric  Weight Decay
```

Let’s try it out.

```
trnCtrl = trainControl(
  method = "repeatedcv",
  repeats = 3,
  classProbs = TRUE)

tuneGrid = expand.grid(
  size = c(2, 4, 8),
  decay = c(0, 1e-2, 1e-1))

nnfit = train(
  Embryonic.day ~ Fn1 + Timd2 + Gata4 + Sox7,
  data = embryoCells,
  method = "nnet",
  tuneGrid  = tuneGrid,
  trControl = trnCtrl,
  metric = "Accuracy")
```

That’s quite a mouthful, but the nice thing is that this syntax is standardized and applies across many different methods. All you need to do specify the name of the method and the grid of tuning parameters that should be explored via the `tuneGrid` argument.

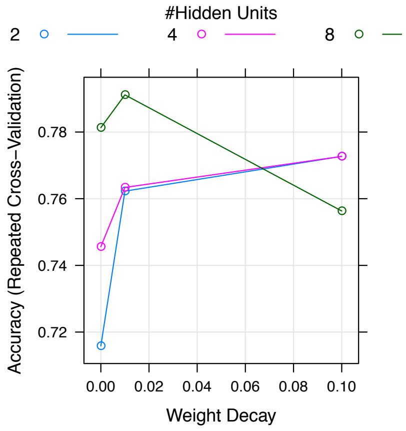

Figure 13.25: Parameter tuning of the neural net by cross-validation.

Now we can have a look at the output (Figure 13.25).

```
nnfit

## Neural Network
##
## 66 samples
##   4 predictor
```

[19] They are described in the manual of the `nnet` function.

```
##  3 classes: 'E3.25', 'E3.5', 'E4.5'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 59, 59, 59, 59, 60, 60, ...
## Resampling results across tuning parameters:
##
##   size  decay  Accuracy   Kappa
##   2     0.00   0.7158333  0.4270723
##   2     0.01   0.7622619  0.5710980
##   2     0.10   0.7726984  0.5926065
##   4     0.00   0.7455952  0.5284067
##   4     0.01   0.7633730  0.5902138
##   4     0.10   0.7726984  0.5919238
##   8     0.00   0.7813095  0.6169179
##   8     0.01   0.7911508  0.6257162
##   8     0.10   0.7563095  0.5595771
##
## Accuracy was used to select the optimal model using  the
##  largest value.
## The final values used for the model were size = 8 and decay = 0.01.
```

```
plot(nnfit)
predict(nnfit) %>% head(10)
```

```
##  [1] E3.25 E3.25 E3.25 E3.25 E3.25 E3.25 E3.25 E3.25 E3.25 E3.25
## Levels: E3.25 E3.5 E4.5
```

▶ Question **13.7.1.** Will the accuracy that we obtained above for the optimal tuning parameters generalize to a new dataset? What could you do to address that?

▶ Answer **13.7.1.** No, it is likely to be too optimistic, as we have picked the optimum. To get a somewhat more realistic estimate of prediction performance when generalized, we could formalize (into computer code) all our data preprocessing choices and the above parameter tuning procedure, and embed this in another, outer cross-validation loop (Ambroise and McLachlan, 2002). However, this is likely still not enough, as we discuss in the next section.

13.7.1 Method hacking

In Chapter 6 we encountered **p-value hacking**. A similar phenomenon exists in statistical learning: given a dataset, we explore various different methods of preprocessing (such as normalization, outlier detection, transformation, feature selection), try out different machine learning algorithms and tune their parameters until we are content with the result. The measured accuracy is likely to be too optimistic, i. e., will not generalize to a new dataset. Embedding as many of our methodical choices into a computational formalism and having an outer cross-validation loop (not to be confused with the inner loop that does the parameter tuning) will ameliorate the problem. But is unlikely to address it completely, since not all our choices can be formalized.

The gold standard remains validation on truly unseen data. In addition, it is never a bad thing if the classifier is not a black box but can be interpreted in terms of domain knowledge. Finally, report not just summary statistics, such as misclassification rates, but lay open the complete computational workflow, so that anyone (including your future self) can convince themselves of the robustness of the result or of the influence of the preprocessing, model selection and tuning choices (Holmes, 2016).

Exercises

▶ Exercise **13.1.** Apply a kernel support vector machine, available in the *kernlab* package, to the `zeller` microbiome data. What kernel function is best?

▶ Exercise **13.2.** It has been quipped that all classification methods are just refinements of two archetypal ideas: discriminant analysis and k nearest neighbors. In what sense might that be a useful classification?

▶ Answer **13.1.** In linear discriminant analysis, we consider our objects as elements of $\mathbb{R}^p$, and the learning task is to define regions in this space, or boundary hyperplanes between them, which we use to predict the class membership of new objects. This is archetypal for **classification by partition**. Generalizations of linear discriminant analysis permit more general spaces and more general boundary shapes.

In k nearest neighbors, no embedding into a coordinate space is needed, but instead we require a distance (or dissimilarity) measure that can be computed between each pair of objects, and the classification decision for a new object depends on its distances to the training objects and their classes. This is archetypal for **kernel-based** methods.

▶ Exercise **13.3.** Use `glmnet` for a prediction of a continous variable, i.e., for regression. Explore the effects of using ridge versus lasso penalty.

▶ Answer **13.2.** There are infinitely many possibilities here. For instance, you could explore the prostate cancer data as in Chapter 3 of (Hastie et al., 2008); the data are available in the CRAN package *ElemStatLearn*.

▶ Exercise **13.4.** Consider smoothing as a regression and model selection problem. What is the equivalent quantity to the penalization parameter λ in Equation (13.8)? How do you choose it?

▶ Answer **13.3.** We refer to Chapter 5 of (Hastie et al., 2008)

▶ Exercise **13.5.** **Scale invariance**. Consider a rescaling of one of the features in the (generalized) linear model (13.7). For instance, denote the ν-th column of x by $x_{\cdot\nu}$, and suppose that $p \geq 2$ and that we rescale $x_{\cdot\nu} \mapsto s\,x_{\cdot\nu}$ with some number $s \neq 0$. What will happen to the estimate $\hat{\beta}$ from Equation (13.8) in (a) the unpenalized case ($\lambda = 0$) and (b) the penalized case ($\lambda > 0$)?

▶ Answer **13.4.** In the unpenalized case, the estimates will be scaled by $1/s$, so that the resulting model is, in effect, the same. In the penalized case, the penalty from the ν-th component of β will be different. If $|s| > 1$, the amplitude of the feature is increased, smaller β-components are required for it to have the same effect in the prediction, and therefore the feature is more likely to receive a non-zero and/or larger estimate, possibly on the cost of the other features; conversely for $|s| < 1$.

Bibliography

Christophe Ambroise and Geoffrey J. McLachlan. Selection bias in gene extraction on the basis of microarray gene-expression data. **PNAS**, 99(10):6562–6566, 2002.

S. Anders, A. Reyes, and W. Huber. Detecting differential usage of exons from RNA-seq data. **Genome Research**, 22(10):2008–2017, 2012.

Simon Anders and Wolfgang Huber. Differential expression analysis for sequence count data. **Genome Biology**, 11:R106, 2010. URL `http://genomebiology.com/2010/11/10/R106`.

Francis J Anscombe. The transformation of poisson, binomial and negative-binomial data. **Biometrika**, pages 246–254, 1948.

Paul L Auer and RW Doerge. Statistical design and analysis of RNA sequencing data. **Genetics**, 185(2):405–416, 2010.

Rhonda Bacher and Christina Kendziorski. Design and computational analysis of single-cell rna-sequencing experiments. **Genome Biology**, 17(1):1, 2016.

A. Baddeley, J. Moller, and R. Waagepetersen. Non- and semiparametric estimation of interaction in inhomogeneous point patterns. **Statistica Neerlandica**, 54:329–350, 2000.

A.J. Baddeley. Spatial sampling and censoring. In O.E. Barndorff-Nielsen, W.S. Kendall, and M.N.M. van Lieshout, editors, **Stochastic Geometry: Likelihood and Computation**, pages 37–78. Chapman and Hall, 1998.

Daniela Beisser, Gunnar W Klau, Thomas Dandekar, Tobias Müller, and Marcus T Dittrich. BioNet: an R-package for the functional analysis of biological networks. **Bioinformatics**, 26(8):1129–1130, 2010.

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. **Neural computation**, 15(6):1373–1396, 2003.

Richard Ernest Bellman. **Adaptive control processes: a guided tour**. Princeton University Press, 1961.

Sean C Bendall, Garry P Nolan, Mario Roederer, and Pratip K Chattopadhyay. A deep profiler's guide to cytometry. **Trends in immunology**, 33(7):323–332, 2012.

Yoshua Bengio, Jean-François Paiement, Pascal Vincent, Olivier Delalleau, Nicolas Le Roux, and Marie Ouimet. Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering. **Advances in neural information processing systems**, 16:177–184, 2004.

Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. **Journal of the Royal Statistical Society B**, 57:289–300, 1995.

Yoav Benjamini and Marina Bogomolov. Selective inference on multiple families of hypotheses. **Journal of the Royal Statistical Society: Series B (Statistical Methodology)**, 76(1):297–318, 2014.

Yoav Benjamini and Daniel Yekutieli. Hierarchical fdr testing of trees of hypotheses. Technical report, Technical report, Department of Statistics and Operations Research, Tel Aviv University, 2003.

M. V. Boland and R. F. Murphy. A neural network classifier capable of recognizing the patterns of all major subcellular structures in fluorescence microscope images of HeLa cells. **Bioinformatics**, 17(12):1213–1223, 2001.

Remco Bouckaert, Joseph Heled, Denise Kühnert, Tim Vaughan, Chieh-Hsi Wu, Dong Xie, Marc A Suchard, Andrew Rambaut, and Alexei J Drummond. BEAST 2: a software platform for bayesian evolutionary analysis. **PLoS Comput Biol**, 10(4):e1003537, 2014.

Richard Bourgon, Robert Gentleman, and Wolfgang Huber. Independent filtering increases detection power for high-throughput experiments. **PNAS**, 107(21):9546–9551, 2010. URL `http://www.pnas.org/content/107/21/9546.long`.

George EP Box, William G Hunter, and J Stuart Hunter. **Statistics for experimenters: an introduction to design, data analysis, and model building**. John Wiley & Sons, 1978.

Eoin L Brodie, Todd Z DeSantis, Dominique C Joyner, Seung M Baek, Joern T Larsen, Gary L Andersen, Terry C Hazen, Paul M Richardson, Donald J Herman, TK Tokunaga, JM Wan, and MK Firestone. Application of a high-density oligonucleotide microarray approach to study bacterial population dynamics during uranium reduction and reoxidation. **Applied and Environmental Microbiology**, 72(9):6288–6298, 2006.

A. N. Brooks, L. Yang, M. O. Duff, K. D. Hansen, J. W. Park, S. Dudoit, S. E. Brenner, and B. R. Graveley. Conservation of an RNA regulatory map between Drosophila and mammals. **Genome Research**, pages 193–202, 2011. ISSN 1088-9051. doi: 10.1101/gr.108662.110. URL `http://genome.cshlp.org/cgi/doi/10.1101/gr.108662.110`.

Michael George Bulmer. **Francis Galton: pioneer of heredity and biometry**. JHU Press, 2003.

Benjamin Callahan, Diana Proctor, David Relman, Julia Fukuyama, and Susan Holmes. Reproducible research workflow in R for the analysis of personalized human microbiome data. In **Pacific Symposium on Biocomputing**, volume 21, page 183. NIH Public Access, 2016a.

Benjamin J Callahan, Paul J McMurdie, Michael J Rosen, Andrew W Han, Amy J Johnson, and Susan P Holmes. DADA2: High resolution sample inference from amplicon data. **Nature Methods**, pages 1–4, 2016b.

C Cannings and AWF Edwards. Natural selection and the de finetti diagram. **Annals of human genetics**, 31(4):421–428, 1968.

J.G. Caporaso, J. Kuczynski, J. Stombaugh, K. Bittinger, F.D. Bushman, E.K. Costello, N. Fierer, A.G. Peña, J.K. Goodrich, J.I. Gordon, and R. Knight. Qiime allows analysis of high-throughput community sequencing data. **Nature methods**, 7(5):335–336, 2010.

J.G. Caporaso, C.L. Lauber, W.A. Walters, D. Berg-Lyons, C.A. Lozupone, P.J. Turnbaugh, N. Fierer, and R. Knight. Global patterns of 16s rrna diversity at a depth of millions of sequences per sample. **PNAS**, 108(Supplement 1):4516–4522, 2011.

A.E. Carpenter, T.R. Jones, M.R. Lamprecht, C. Clarke, I.H. Kang, O. Friman, D.A. Guertin, J.H. Chang, R.A. Lindquist, J. Moffat, et al. CellProfiler: image analysis software for identifying and quantifying cell phenotypes. **Genome Biology**, 7:R100, 2006.

Daniel B Carr, Richard J Littlefield, WL Nicholson, and JS Littlefield. Scatterplot matrix techniques for large N. **Journal of the American Statistical Association**, 82(398):424–436, 1987.

Ron Caspi, Tomer Altman, Joseph M Dale, Kate Dreher, Carol A Fulcher, Fred Gilham, Pallavi Kaipa, Athikkattuvalasu S Karthikeyan, Anamika Kothari, Markus Krummenacker, et al. The metacyc database of metabolic pathways and enzymes and the biocyc collection of pathway/genome databases. **Nucleic acids research**, 38(suppl 1):D473–D479, 2010.

John Chakerian and Susan Holmes. Computational tools for evaluating phylogenetic and hierarchical clustering trees. **Journal of Computational and Graphical Statistics**, 21(3):581–599, 2012.

Min Chen, Yang Xie, and Michael Story. An exponential-gamma convolution model for background correction of illumina beadarray data. **Communications in Statistics-Theory and Methods**, 40(17):3055–3069, 2011.

Sung Nok Chiu, Dietrich Stoyan, Wilfrid S. Kendall, and Joseph Mecke. **Stochastic geometry and its applications**. Springer, 2013.

Line Clemmensen, Trevor Hastie, Daniela Witten, and Bjarne Ersbøll. Sparse discriminant analysis. **Technometrics**, 53:406–413, 2011.

W. S. Cleveland, M. E. McGill, and R. McGill. The shape parameter of a two-variable graph. **Journal of the American Statistical Association**, 83:289–300, 1988.

William S Cleveland. **The Collected Works of John W. Tukey: Graphics 1965-1985**, volume 5. CRC Press, 1988.

J.R. Cole, Q. Wang, E. Cardenas, J. Fish, B. Chai, R.J. Farris, A.S. Kulam-Syed-Mohideen, D.M. McGarrell, T. Marsh, G.M. Garrity, and J.M. Tiedje. The ribosomal database project: improved alignments and new tools for rrna analysis. **Nucleic acids research**, 37 (Supplement 1):D141–D145, 2009.

1000 Genomes Project Consortium. An integrated map of genetic variation from 1,092 human genomes. **Nature**, 491(7422):56–65, 2012.

R. Dennis Cook. Detection of Influential Observation in Linear Regression. **Technometrics**, February 1977.

N.A.C. Cressie. **Statistics for spatial data**. John Wiley and Sons, 1991.

Fabrice de Chaumont, Stéphane Dallongeville, Nicolas Chenouard, Nicolas Hervé, Sorin Pop, Thomas Provoost, Vannary Meas-Yedid, Praveen Pankajakshan, Timothé Lecomte, Yoann Le Montagner, Thibault Lagache, Alexandre Dufour, and Jean-Christophe Olivo-Marin. Icy: an open bioimage informatics platform for extended reproducible research. **Nature Methods**, 9:690–696, 2012.

Bruno DeFinetti. Considerazioni matematiche sull'ereditarieta mendeliana. **Metron**, 6:3–41, 1926.

T. Z. DeSantis, P. Hugenholtz, N. Larsen, M. Rojas, E. L. Brodie, K. Keller, T. Huber, D. Dalevi, P. Hu, and G. L. Andersen. Greengenes, a Chimera-Checked 16S rRNA Gene Database and Workbench Compatible with ARB. **Appl. Environ. Microbiol.**, 72(7):5069–5072, 2006. doi: 10.1128/AEM.03006-05. URL `http://aem.asm.org/cgi/content/abstract/72/7/5069`.

Persi Diaconis and David Freedman. Finite exchangeable sequences. **The Annals of Probability**, pages 745–764, 1980.

Persi Diaconis and Susan Holmes. Gray codes for randomization procedures. **Statistics and Computing**, 4(4):287–302, 1994.

Persi Diaconis, Susan Holmes, and Richard Montgomery. Dynamical bias in the coin toss. **SIAM review**, 49(2):211–235, 2007.

Edwin Diday and M Paula Brito. Symbolic cluster analysis. In **Conceptual and Numerical Analysis of Data**, pages 45–84. Springer, 1989.

P.J. Diggle. **Statistical analysis of spatial point patterns**. Academic Press, 1983.

Daniel B. DiGiulioa, Benjamin J. Callahan, Paul J. McMurdie, Elizabeth K. Costello, Deirdre J. Lyelle, Anna Robaczewskaa, Christine L. Sun, Daniela S. Aliaga-Goltsman, Ronald J. Wongand Gary M. Shaw, David K. Stevenson, Susan P. Holmes, and David A. Relman. Temporal and spatial variation of the human microbiota during pregnancy. **PNAS**, 2015.

Murat Dundar, Ferit Akova, Halid Z. Yerebakan, and Bartek Rajwa. A non-parametric bayesian model for joint cell clustering and cluster matching: identification of anomalous sample phenotypes with random effects. **BMC Bioinformatics**, 15(1):1–15, 2014. ISSN 1471-2105. doi: 10.1186/1471-2105-15-314. URL `http://dx.doi.org/10.1186/1471-2105-15-314`.

Richard Durbin, Sean Eddy, Anders Krogh, and Graeme Mitchison. **Biological Sequence Analysis**. Cambridge University Press, 1998.

R.C. Edgar and H. Flyvbjerg. Error filtering, pair assembly and error correction for next-generation sequencing reads. **Bioinformatics**, 31(21):3476–3482, 2015.

Bradley Efron. **Large-scale inference: empirical Bayes methods for estimation, testing, and prediction**, volume 1. Cambridge University Press, 2010.

Bradley Efron and Robert J Tibshirani. **An introduction to the bootstrap**. CRC press, 1994.

D Elson and E Chargaff. On the desoxyribonucleic acid content of sea urchin gametes. **Experientia**, 8(4):143–145, 1952.

Steven N Evans and Frederick A Matsen. The phylogenetic Kantorovich–Rubinstein metric for environmental sequence samples. **Journal of the Royal Statistical Society: Series B (Statistical Methodology)**, 74(3):569–592, 2012.

Ronald Aylmer Fisher. **The design of experiments**. Oliver & Boyd, 1935.

Bernard Flury. **A first course in multivariate statistics**. Springer, 1997.

David A Freedman. Statistical models and shoe leather. **Sociological methodology**, 21(2):291–313, 1991.

Jerome H Friedman. On bias, variance, 0/1âĂŤloss, and the curse-of-dimensionality. **Data Mining and Knowledge Discovery**, 1: 55–77, 1997.

Jerome H Friedman and Lawrence C Rafsky. Multivariate generalizations of the wald-wolfowitz and smirnov two-sample tests. **The Annals of Statistics**, pages 697–717, 1979.

Julia Fukuyama, Paul J McMurdie, Les Dethlefsen, David A Relman, and Susan Holmes. Comparisons of distance methods for combining covariates and abundances in microbiome studies. In **Pac Symp Biocomput**. World Scientific, 2012.

Robert C Gentleman, Vincent J Carey, Douglas M Bates, Ben Bolstad, Marcel Dettling, Sandrine Dudoit, Byron Ellis, Laurent Gautier, Yongchao Ge, Jeff Gentry, Kurt Hornik, Torsten Hothorn, Wolfgang Huber, Stefano Iacus, Rafael Irizarry, Friedrich Leisch, Cheng Li, Martin Maechler, Anthony J Rossini, Gunther Sawitzki, Colin Smith, Gordon Smyth, Luke Tierney, Jean Y H Yang, and Jianhua Zhang. Bioconductor: open software development for computational biology and bioinformatics. **Genome Biology**, 5(10):R80, Jan 2004. doi: 10.1186/gb-2004-5-10-r80. URL `http://genomebiology.com/2004/5/10/R80`.

Yoav Gilad and Orna Mizrahi-Man. A reanalysis of mouse encode comparative gene expression data. **F1000Research**, 4, 2015.

David J Glass. **Experimental design for biologists**. Cold Spring Harbor Laboratory Press, 2007.

Anastassia Gorvitovskaia, Susan P Holmes, and Susan M Huse. Interpreting prevotella and bacteroides as biomarkers of diet and lifestyle. **Microbiome**, 4(1):1, 2016.

R_ Grantham, Christian Gautier, Manolo Gouy, M Jacobzone, and R Mercier. Codon catalog usage is a genome strategy modulated for gene expressivity. **Nucleic acids research**, 9(1):213–213, 1981.

M.J. Greenacre. **Correspondence analysis in practice**. Chapman & Hall, 2007.

Bettina Grun, Theresa Scharl, and Friedrich Leisch. Modelling time course gene expression data with finite mixtures of linear additive models. **Bioinformatics**, 28(2):222–228, 2012. doi: 10.1093/bioinformatics/btr653. URL `http://bioinformatics.oxfordjournals.org/content/28/2/222.abstract`.

Alon Halevy, Peter Norvig, and Fernando Pereira. The unreasonable effectiveness of data. **Intelligent Systems, IEEE**, 24(2):8–12, 2009.

Robin M Hallett, Anna Dvorkin-Gheva, Anita Bane, and John A Hassell. A gene signature for predicting outcome in patients with basal-like breast cancer. **Scientific reports**, 2, 2012.

Trevor Hastie and Werner Stuetzle. Principal curves. **Journal of the American Statistical Association**, 84(406):502–516, 1989.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. **The Elements of Statistical Learning**. Springer, 2^{nd} edition, 2008.

Megan L Head, Luke Holman, Rob Lanfear, Andrew T Kahn, and Michael D Jennions. The extent and consequences of p-hacking in science. **PLoS Biol**, 13(3):e1002106, 2015.

M. Held, M.H.A. Schmitz, B. Fischer, T. Walter, B. Neumann, M.H. Olma, M. Peter, J. Ellenberg, and D.W. Gerlich. CellCognition: time-resolved phenotype annotation in high-throughput live cell imaging. **Nature Methods**, 7:747, 2010.

F. Henderson. Software Engineering at Google. **ArXiv e-prints**, 2017.

Jennifer A Hoeting, David Madigan, Adrian E Raftery, and Chris T Volinsky. Bayesian model averaging: a tutorial. **Statistical science**, pages 382–401, 1999.

S. Holmes. Statistics for phylogenetic trees. **Theoretical population biology**, 63(1):17–32, 2003. ISSN 0040-5809.

Susan Holmes. Phylogenetic trees: an overview. In **Statistics and Genetics**, number 112, pages 81–118. Springer, IMA, New York, 1999.

Susan Holmes. Multivariate analysis: The French way. In D. Nolan and T. P. Speed, editors, **Probability and Statistics: Essays in Honor of David A. Freedman**, volume 56 of **IMS Lecture Notes–Monograph Series**. IMS, Beachwood, OH, 2006. URL `http://www.imstat.org/publications/lecnotes.htm`.

Susan Holmes. Statistical proof? the problem of irreproducibility. **Bulletin AMS**, ???:???, 2016.

Susan Holmes, Michael He, Tong Xu, and Peter P Lee. Memory t cells have gene expression patterns intermediate between naive and effector. **PNAS**, 102(15):5519–5523, 2005.

Susan Holmes, Alexander Alekseyenko, Alden Timme, Tyrell Nelson, P.J. Pasricha, and Alfred Spormann. Visualization and statistical comparisons of microbial communities using R packages on Phylochip data. In **Pacific Symposium on Biocomputing**, page 142, 2011a.

Susan Holmes, Alexander V Alekseyenko, Alden Timme, Tyrrell Nelson, Pankaj Jay Pasricha, and Alfred Spormann. Visualization and statistical comparisons of microbial communities using r packages on phylochip data. In **Pacific Symposium on Biocomputing**, pages 142–153. World Scientific, 2011b.

Susan Holmes Junca. **Outils informatiques pour l'évaluation de la pertinence d'un résultat en analyse des données**. PhD thesis, Université Montpellier II, France, 1985. Diss.

Kurt Hornik. A CLUE for CLUster Ensembles. **Journal of Statistical Software**, 14(12), 2005.

H Hotelling. Analysis of a complex of statistical variables into principal components. **Journal of Educational Psychology**, 24(6): 417–441, 1933.

Harold Hotelling. Some improvements in weighing and other experimental techniques. **The Annals of Mathematical Statistics**, 15 (3):297–306, 1944.

Peter J. Huber. Robust estimation of a location parameter. **The Annals of Mathematical Statistics**, 35:73–101, 1964.

Wolfgang Huber, Vincent J Carey, Robert Gentleman, Simon Anders, Marc Carlson, Benilton S Carvalho, Hector Corrada Bravo, Sean Davis, Laurent Gatto, Thomas Girke, Raphael Gottardo, Florian Hahne, Kasper D Hansen, Rafael A Irizarry, Michael Lawrence, Michael I Love, James MacDonald, Valerie Obenchain, Andrzej K Oleś, Hervé Pagès, Alejandro Reyes, Paul Shannon, Gordon K Smyth, Dan Tenenbaum, Levi Waldron, and Martin Morgan. Orchestrating high-throughput genomic analysis with bioconductor.

Nature Methods, 12(2):115–121, 2015.

Henry R Hulett, William A Bonner, Janet Barrett, and Leonard A Herzenberg. Cell sorting: automated separation of mammalian cells as a function of intracellular fluorescence. **Science**, 166(3906):747–749, 1969.

Trey Ideker, Owen Ozier, Benno Schwikowski, and Andrew F Siegel. Discovering regulatory and signalling circuits in molecular interaction networks. **Bioinformatics**, 18 Suppl 1:S233–40, Jan 2002. URL `http://bioinformatics.oxfordjournals.org/cgi/reprint/18/suppl_1/S233`.

Nikolaos Ignatiadis, Bernd Klaus, Judith Zaugg, and Wolfgang Huber. Data-driven hypothesis weighting increases detection power in genome-scale multiple testing. **Nature Methods**, 2016.

Ross Ihaka. Color for presentation graphics. In Kurt Hornik and Friedrich Leisch, editors, **Proceedings of the 3rd International Workshop on Distributed Statistical Computing**. ISSN 1609-395X, Vienna, Austria, 2003.

Ross Ihaka and Robert Gentleman. R: A language for data analysis and graphics. **Journal of Computational and Graphical Statistics**, 5(3):299–314, 1996.

R. A. Irizarry, B. Hobbs, F. Collin, Y. D. Beazer-Barclay, K. J. Antonellis, U. Scherf, and T. P. Speed. Exploration, normalization, and summaries of high density oligonucleotide array probe level data. **Biostatistics**, 4(2):249–264, 2003.

Rafael A Irizarry, Hao Wu, and Andrew P Feinberg. A species-generalized probabilistic model-based definition of cpg islands. **Mammalian Genome**, 20(9-10):674–680, 2009.

Alan Julian Izenman. **Nonlinear Dimensionality Reduction and Manifold Learning**, pages 597–632. Springer New York, New York, NY, 2008.

Laurent Jacob, Guillaume Obozinski, and Jean-Philippe Vert. Group lasso with overlap and graph lasso. In **Proceedings of the 26th Annual International Conference on Machine Learning**, pages 433–440. ACM, 2009.

Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. **An introduction to statistical learning**. Springer, 2013.

Pierre Jolicoeur, James E Mosimann, et al. Size and shape variation in the painted turtle. a principal component analysis. **Growth**, 24 (4):339–354, 1960.

Ian Jolliffe. **Principal component analysis**. Wiley Online Library, 2002.

T. Jones, A. Carpenter, and P. Golland. Voronoi-based segmentation of cells on image manifolds. **Computer Vision for Biomedical Image Applications**, page 535, 2005.

Daniel Kahneman. **Thinking, fast and slow**. Macmillan, 2011.

Purna C Kashyap, Angela Marcobal, Luke K Ursell, Samuel A Smits, Erica D Sonnenburg, Elizabeth K Costello, Steven K Higginbottom, Steven E Domino, Susan P Holmes, David A Relman, J.I. Gordon, and J Sonnenburg. Genetically dictated change in host mucus carbohydrate landscape exerts a diet-dependent effect on the gut microbiota. **PNAS**, 110(42):17059–17064, 2013.

Leonard Kaufman and Peter J Rousseeuw. Partitioning around medoids (program pam). **Finding groups in data: an introduction to cluster analysis**, pages 68–125, 1990.

David Kendall. Incidence matrices, interval graphs and seriation in archeology. **Pacific Journal of mathematics**, 28(3):565–570, 1969.

DG Kendall. A mathematical approach to seriation. **Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences**, 269(1193):125–134, 1970.

Marc Kéry and J Andrew Royle. **Applied Hierarchical Modeling in Ecology: Analysis of distribution, abundance and species richness in R and BUGS: Volume 1: Prelude and Static Models**. Academic Press, 2015.

Raffaella Koncan, Aránzazu Valverde, María-Isabel Morosini, María García-Castillo, Rafael Cantón, Giuseppe Cornaglia, Fernando Baquero, and Rosa del Campo. Learning from mistakes: Taq polymerase contaminated with β-lactamase sequences results in false emergence of streptococcus pneumoniae containing tem. **Journal of antimicrobial chemotherapy**, 60(3):702–703, 2007.

James J Kozich, Sarah L Westcott, Nielson T Baxter, Sarah K Highlander, and Patrick D Schloss. Development of a dual-index sequencing strategy and curation pipeline for analyzing amplicon sequence data on the miseq illumina sequencing platform. **Applied and environmental microbiology**, 79(17):5112–5120, 2013.

Erik Kristiansson, Michael Thorsen, Markus J Tamás, and Olle Nerman. Evolutionary forces act on promoter length: identification of enriched cis-regulatory elements. **Molecular biology and evolution**, 26(6):1299–1307, 2009.

Pei Fen Kuan, Dongjun Chung, Guangjin Pan, James A Thomson, Ron Stewart, and Sündüz Keleş. A statistical framework for the analysis of chip-seq data. **Journal of the American Statistical Association**, 106(495):891–903, 2011.

Kenneth Lange. **MM Optimization Algorithms**. SIAM, 2016.

Christina Laufer, Bernd Fischer, Maximilian Billmann, Wolfgang Huber, and Michael Boutros. Mapping genetic interactions in human cancer cells with RNAi and multiparametric phenotyping. **Nature Methods**, 10:427–431, 2013.

Joshua Lederberg and Alexa Mccray. 'Ome Sweet 'Omics– A Genealogical Treasury of Words. **The Scientist**, 17(7), April 2001. URL `http://www.the-scientist.com/article/display/12335/`.

J. T. Leek and J. D. Storey. Capturing heterogeneity in gene expression studies by surrogate variable analysis. **PLoS Genetics**, 3(9): 1724–1735, 2007.

Jeffrey T Leek, Robert B Scharpf, Héctor Corrada Bravo, David Simcha, Benjamin Langmead, W Evan Johnson, Donald Geman, Keith Baggerly, and Rafael A Irizarry. Tackling the widespread and critical impact of batch effects in high-throughput data. **Nature Reviews Genetics**, 11(10):733–739, 2010.

Wen-Hsiung Li. **Molecular evolution.** Sinauer Associates Incorporated, 1997.

Wen-Hsiung Li and Dan Graur. **Fundamentals of molecular evolution**, volume 48. Sinauer Associates Sunderland, MA, 1991.

Arthur Liberzon, Aravind Subramanian, Reid Pinchback, Helga Thorvaldsdóttir, Pablo Tamayo, and Jill P Mesirov. Molecular signatures database (msigdb) 3.0. **Bioinformatics**, 27(12):1739–1740, 2011.

Shin Lin, Yiing Lin, Joseph R Nery, Mark A Urich, Alessandra Breschi, Carrie A Davis, Alexander Dobin, Christopher Zaleski, Michael A Beer, William C Chapman, et al. Comparison of the transcriptional landscapes between human and mouse tissues. **PNAS**, 111(48): 17224–17229, 2014.

Michael I Love, Wolfgang Huber, and Simon Anders. Moderated estimation of fold change and dispersion for rna-seq data with deseq2. **Genome biology**, 15(12):1–21, 2014.

Michael I. Love, Simon Anders, Vladislav Kim, and Wolfgang Huber. Rna-seq workflow: gene-level exploratory analysis and differential expression. **F1000Research**, 4(1070), 2015. doi: 10.12688/f1000research.7035.1.

C.A. Lozupone, M. Hamady, S.T. Kelley, and R. Knight. Quantitative and qualitative {beta} diversity measures lead to different insights into factors that structure microbial communities. **Applied and environmental microbiology**, 73(5):1576, 2007.

Kanti Mardia, John T Kent, and John M Bibby. **Multiariate Analysis**. Academic Press, New York, 1979.

Jean-Michel Marin and Christian Robert. **Bayesian core: a practical approach to computational Bayesian statistics**. Springer Science & Business Media, 2007.

William T McCormick Jr, Paul J Schweitzer, and Thomas W White. Problem decomposition and data reorganization by a clustering technique. **Operations Research**, 20(5):993–1009, 1972.

Geoffrey McLachlan and Thriyambakam Krishnan. **The EM algorithm and extensions**, volume 382. John Wiley & Sons, 2007.

Geoffrey McLachlan and David Peel. **Finite mixture models**. John Wiley & Sons, 2004.

Paul J McMurdie and Susan Holmes. Waste not, want not: Why rarefying microbiome data is inadmissible. **PLoS Computational Biology**, 10(4):e1003531, 2014.

Paul J McMurdie and Susan Holmes. Shiny-phyloseq: Web application for interactive microbiome analysis with provenance tracking. **Bioinformatics**, 31(2):282–283, 2015.

Roger Mead. **The design of experiments: statistical principles for practical applications**. Cambridge University Press, 1990.

Victoria Moignard, Steven Woodhouse, Laleh Haghverdi, Andrew J Lilly, Yosuke Tanaka, Adam C Wilkinson, Florian Buettner, Iain C Macaulay, Wajid Jawaid, Evangelia Diamanti, Shin-Ichi Nishikawa, Nir Piterman, Valerie Kouskoff, Fabian J Theis, Jasmin Fisher, and Berthold Göttgens. Decoding the regulatory network of early blood development from single-cell gene expression measurements. **Nature biotechnology**, 2015.

J. Mollon. Seeing colour. In T. Lamb and J. Bourriau, editors, **Colour: Art and Science**. Cambridge Unversity Press, 1995.

Alexander M Mood. On hotelling's weighing problem. **The Annals of Mathematical Statistics**, pages 432–446, 1946.

AE Mourant, Ada Kopec, and K Domaniewska-Sobczak. The distribution of the human blood groups 2nd edition, 1976.

Daniel Müllner. fastcluster: Fast hierarchical, agglomerative clustering routines for r and python. **Journal of Statistical Software**, 53(9):1–18, 2013.

Serban Nacu, Rebecca Critchley-Thorne, Peter Lee, and Susan Holmes. Gene expression network analysis and applications to immunology. **Bioinformatics**, 23(7):850–8, Apr 2007. doi: 10.1093/bioinformatics/btm019. URL `http://bioinformatics.oxfordjournals.org/cgi/content/abstract/23/7/850`.

TA Nelson, S Holmes, AV Alekseyenko, M Shenoy, T Desantis, CH Wu, GL Andersen, J Winston, J Sonnenburg, PJ Pasricha, and A Spormann. Phylochip microarray analysis reveals altered gastrointestinal microbial communities in a rat model of colonic hypersensitivity. **Neurogastroenterology & Motility**, 23(2):169–e42, 2011.

B. Neumann, T. Walter, J. K. Heriche, J. Bulkescher, H. Erfle, C. Conrad, P. Rogers, I. Poser, M. Held, U. Liebel, C. Cetin, F. Sieckmann, G. Pau, R. Kabbe, A. Wunsche, V. Satagopam, M. H. Schmitz, C. Chapuis, D. W. Gerlich, R. Schneider, R. Eils, W. Huber, J. M. Peters, A. A. Hyman, R. Durbin, R. Pepperkok, and J. Ellenberg. Phenotypic profiling of the human genome by time-lapse microscopy reveals cell division genes. **Nature**, 464(7289):721–727, Apr 2010.

Jerzy Neyman and Egon S Pearson. **Sufficient statistics and uniformly most powerful tests of statistical hypotheses**. University California Press, 1936.

Ann L Oberg and Olga Vitek. Statistical design of quantitative mass spectrometry-based proteomic experiments. **Journal of proteome research**, 8(5):2144–2156, 2009.

Y. Ohnishi, W. Huber, A. Tsumura, M. Kang, P. Xenopoulos, K. Kurimoto, A. K. Oles, M. J. Arauzo-Bravo, M. Saitou, A. K. Hadjantonakis, and T. Hiiragi. Cell-to-cell expression variability followed by signal reinforcement progressively segregates early mouse lineages. **Nature Cell Biology**, 16(1):27–37, 2014.

Kieran O'Neill, Nima Aghaeepour, Josef Špidlen, and Ryan Brinkman. Flow cytometry bioinformatics. **PLoS Comput Biol**, 9(12): e1003365, 2013.

S original by Trevor Hastie R port by Andreas Weingessel <Andreas.Weingessel@ci.tuwien.ac.at>. **princurve: Fits a Principal Curve in Arbitrary Dimension**, 2013. URL `https://CRAN.R-project.org/package=princurve`. R package version 1.1-12.

F. Ozsolak and P. M. Milos. RNA sequencing: advances, challenges and opportunities. **Nature Reviews Genetics**, 12:87–98, 2011.

Emmanuel Paradis. **Analysis of Phylogenetics and Evolution with R**. Springer Science & Business Media, 2011.

Rob Patro, Stephen M Mount, and Carl Kingsford. Sailfish enables alignment-free isoform quantification from RNA-seq reads using lightweight algorithms. **Nature Biotechnology**, 32(5):462–464, 2014.

G. Pau, F. Fuchs, O. Sklyar, M. Boutros, and W. Huber. EBImage - an R package for image processing with applications to cellular phenotypes. **Bioinformatics**, 26:979, 2010.

Sandrine Pavoine, Anne-Béatrice Dufour, and Daniel Chessel. From dissimilarities among species to dissimilarities among communities: a double principal coordinate analysis. **Journal of theoretical biology**, 228(4):523–537, 2004.

Guy Perrière and Jean Thioulouse. Use and misuse of correspondence analysis in codon usage studies. **Nucleic Acids Research**, 30 (20):4548–4555, 2002.

Stan Pounds and Stephan W Morris. Estimating the occurrence of false positives and false negatives in microarray studies by approximating and partitioning the empirical distribution of p-values. **Bioinformatics**, 19(10):1236–1242, 2003.

IC Prentice. Non-metric ordination methods in ecology. **The Journal of Ecology**, pages 85–94, 1977.

Elmar Pruesse, Christian Quast, Katrin Knittel, Bernhard M Fuchs, Wolfgang Ludwig, Jörg Peplies, and Frank Oliver Glöckner. Silva: a comprehensive online resource for quality checked and aligned ribosomal rna sequence data compatible with arb. **Nucleic acids research**, 35(21):7188–7196, 2007.

Elizabeth Purdom. Analysis of a data matrix and a graph: Metagenomic data and the phylogenetic tree. **Annals of Applied Statistics**, Jul 2010.

Elizabeth Purdom and Susan P Holmes. Error distribution for gene expression data. **Statistical applications in genetics and molecular biology**, 4(1), 2005.

S. Rajaram, B. Pavie, L. F. Wu, and S. J. Altschuler. PhenoRipper: software for rapidly profiling microscopy images. **Nature Methods**, 9:635–637, 2012.

GM Reaven and RG Miller. An attempt to define the nature of chemical diabetes using a multidimensional analysis. **Diabetologia**, 16 (1):17–24, 1979.

Soo-Yon Rhee, Matthew J Gonzales, Rami Kantor, Bradley J Betts, Jaideep Ravela, and Robert W Shafer. Human immunodeficiency virus reverse transcriptase and protease sequence database. **Nucleic acids research**, 31(1):298–303, 2003.

John Rice. **Mathematical statistics and data analysis**. Cengage Learning, 2006.

B.D. Ripley. **Statistical inference for spatial processes.** Cambridge University Press, 1988.

Davide Risso, John Ngai, Terence P Speed, and Sandrine Dudoit. Normalization of rna-seq data using factor analysis of control genes or samples. **Nature biotechnology**, 32(9):896–902, 2014.

Christian Robert and George Casella. **Introducing Monte Carlo Methods with R**. Springer Science & Business Media, 2009a.

Christian Robert and George Casella. **Introducing Monte Carlo Methods with R**. Springer Science & Business Media, 2009b.

Garry Robins, Tom Snijders, Peng Wang, Mark Handcock, and Philippa Pattison. Recent developments in exponential random graph (p*) models for social networks. **Social networks**, 29(2):192–215, 2007.

David M Rocke and Blythe Durbin. A model for measurement error for gene expression arrays. **Journal of Computational Biology**, 8(6):557–569, 2001.

Fredrik Ronquist, Maxim Teslenko, Paul van der Mark, Daniel L Ayres, Aaron Darling, Sebastian Höhna, Bret Larget, Liang Liu, Marc A Suchard, and John P Huelsenbeck. Mrbayes 3.2: efficient bayesian phylogenetic inference and model choice across a large model space. **Systematic biology**, 61(3):539–542, 2012.

Michael J Rosen, Benjamin J Callahan, Daniel S Fisher, and Susan P Holmes. Denoising pcr-amplified metagenome data. **BMC Bioinformatics**, 13(1):283, 2012.

Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. **Journal of computational and applied mathematics**, 20:53–65, 1987.

Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. **Science**, 290(5500):2323–2326, 2000.

Kris Sankaran and Susan Holmes. structssi: Simultaneous and selective inference for grouped or hierarchically structured data. **Journal of Statistical Software**, 59(1):1–21, 2014.

Mark F Schilling. Multivariate two-sample tests based on nearest neighbors. **Journal of the American Statistical Association**, 81 (395):799–806, 1986.

Johannes Schindelin, Ignacio Arganda-Carreras, Erwin Frise, Verena Kaynig, Mark Longair, Tobias Pietzsch, Stephan Preibisch, Curtis Rueden, Stephan Saalfeld, Benjamin Schmid, Jean-Yves Tinevez, Daniel James White, Volker Hartenstein, Kevin Eliceiri, Pavel Tomancak, and Albert Cardona. Fiji: an open-source platform for biological-image analysis. **Nature Methods**, 9:676–682, 2012.

P D Schloss, S L Westcott, T Ryabin, J R Hall, M Hartmann, E B Hollister, R A Lesniewski, B B Oakley, D H Parks, C J Robinson, J W Sahl, B Stres, G G Thallinger, D J Van Horn, and C F Weber. Introducing mothur: Open-Source, Platform-Independent, Community-Supported Software for Describing and Comparing Microbial Communities. **Appl. and environmental microbiology**, 75(23): 7537–7541, November 2009.

P.D. Schloss, A.M. Schuber, J.P. Zackular, K.D. Iverson, Young V.B., and Petrosino J.F. Stabilization of the murine gut microbiome following weaning. **Gut microbes**, 3(4):383–393, 2012.

Bernhard Schölkopf, Koji Tsuda, and Jean-Philippe Vert. **Kernel methods in computational biology**. MIT press, 2004.

Stephen Senn. Controversies concerning randomization and additivity in clinical trials. **Statistics in medicine**, 23(24):3729–3753, 2004.

Jean Serra. **Image Analysis and Mathematical Morphology**. Academic Press, 1983.

A Francesca Setiadi, Nelson C Ray, Holbrook E Kohrt, Adam Kapelner, Valeria Carcamo-Cavazos, Edina B Levic, Sina Yadegarynia, Chris M Van Der Loos, Erich J Schwartz, Susan Holmes, and PP Lee. Quantitative, architectural analysis of immune cell subsets in tumor-draining lymph nodes from breast cancer patients and healthy lymph nodes. **PLoS One**, 5(8):e12420, 2010.

Cosma Shalizi. **Advanced Data Analysis from an Elementary Point of View**. Cambridge University Press, 2017. URL `https://www.stat.cmu.edu/~cshalizi/ADAfaEPoV/ADAfaEPoV.pdf`.

David Sims, Ian Sudbery, Nicholas E Ilott, Andreas Heger, and Chris P Ponting. Sequencing depth and coverage: key considerations in genomic analyses. **Nature Reviews Genetics**, 15(2):121–132, 2014.

Charlotte Soneson, Michael I. Love, and Mark Robinson. Differential analyses for RNA-seq: transcript-level estimates improve gene-level inferences. **F1000Research**, 4(1521), 2015. doi: 10.12688/f1000research.7563.2.

O. Stegle, L. Parts, R. Durbin, and J. Winn. A Bayesian framework to account for complex non-genetic factors in gene expression levels greatly increases power in eQTL studies. **PLoS Computational Biology**, 6(5):e1000770, 2010.

T. Steijger, J. F. Abril, P. G. Engstrom, F. Kokocinski, T. J. Hubbard, R. Guigo, J. Harrow, P. Bertone, J. F. Abril, M. Akerman, T. Alioto, G. Ambrosini, S. E. Antonarakis, J. Behr, P. Bertone, R. Bohnert, P. Bucher, N. Cloonan, T. Derrien, S. Djebali, J. Du, S. Dudoit, P. Engstrom, M. Gerstein, T. R. Gingeras, D. Gonzalez, S. M. Grimmond, R. Guigo, L. Habegger, J. Harrow, T. J. Hubbard, C. Iseli, G. Jean, A. Kahles, F. Kokocinski, J. Lagarde, J. Leng, G. Lefebvre, S. Lewis, A. Mortazavi, P. Niermann, G. Ratsch, A. Reymond, P. Ribeca, H. Richard, J. Rougemont, J. Rozowsky, M. Sammeth, A. Sboner, M. H. Schulz, S. M. Searle, N. D. Solorzano, V. Solovyev, M. Stanke, T. Steijger, B. J. Stevenson, H. Stockinger, A. Valsesia, D. Weese, S. White, B. J. Wold, J. Wu, T. D. Wu, G. Zeller, D. Zerbino, and M. Q. Zhang. Assessment of transcript reconstruction methods for RNA-seq. **Nature Methods**, 10(12):1177–1184, 2013.

Stephen M Stigler. **The seven pillars of statistical wisdom**. Harvard University Press, 2016.

Gilbert Strang and Wellesley-Cambridge Press. **Introduction to linear algebra**, volume 3. Wellesley-Cambridge Press Wellesley, MA, 1993.

Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. **science**, 290(5500):2319–2323, 2000.

Cajo ter Braak. Correspondence analysis of incidence and abundance data: Properties in terms of a unimodal respose. **Biometrics**, 41, Jan 1985.

Robert Tibshirani. Regression shrinkage and selection via the lasso. **Journal of the Royal Statistical Society. Series B (Methodological)**, pages 267–288, 1996.

Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the number of clusters in a data set via the gap statistic. **Journal of the Royal Statistical Society: Series B (Statistical Methodology)**, 63(2):411–423, 2001.

Michael W Trosset and Carey E Priebe. The out-of-sample problem for classical multidimensional scaling. **Computational statistics & data analysis**, 52(10):4635–4642, 2008.

George C Tseng and Wing H Wong. Tight clustering: A resampling-based approach for identifying stable and tight patterns in data. **Biometrics**, 61(1):10–16, 2005.

John W Tukey. Exploratory data analysis. **Massachusetts: Addison-Wesley**, 1977.

Amos Tversky and Daniel Kahneman. Heuristics and biases: Judgement under uncertainty. **Science**, 185:1124–1130, 1974.

Amos Tversky and Daniel Kahneman. Judgment under uncertainty: Heuristics and biases. In **Utility, probability, and human decision making**, pages 141–162. Springer, 1975.

Pierre-FranÃğois Verhulst. Recherches mathématiques sur la loi d'accroissement de la population. **Nouveaux Mémoires de l'Académie Royale des Sciences et Belles-Lettres de Bruxelles**, 18:1âĂŞ42, 1845.

Martin Vetterli, Jelena Kovačević, and Vivek Goyal. **Foundations of Signal Processing**. Cambridge University Press, 2014.

H. von Helmholtz. **Handbuch der Physiologischen Optik**. Leopold Voss, Leipzig, 1867.

Wencke Walter and Fatima Sanchez-Cabo. **GOplot: Visualization of Functional Analysis Data**, 2015. URL `https://CRAN.`

R-project.org/package=GOplot. R package version 1.0.1.

Q. Wang, G.M. Garrity, J.M. Tiedje, and J.R. Cole. Naive bayesian classifier for rapid assignment of rrna sequences into the new bacterial taxonomy. **Applied and environmental microbiology**, 73(16):5261, 2007.

Ronald L Wasserstein and Nicole A Lazar. The asa's statement on p-values: context, process, and purpose. **The American Statistician**, 2016.

Hadley Wickham. A layered grammar of graphics. **Journal of Computational and Graphical Statistics**, 19(1):3–28, 2010.

Hadley Wickham. Tidy data. **Journal of Statistical Software**, 59(10), 2014.

Hadley Wickham. **ggplot2: Elegant Graphics for Data Analysis**. Springer New York, 2016. ISBN 978-0-387-98140-6. URL http://had.co.nz/ggplot2/book. Second Edition.

Mark A Wiel, Tonje G Lien, Wina Verlaat, Wessel N Wieringen, and Saskia M Wilting. Better prediction by use of co-data: adaptive group-regularized ridge regression. **Statistics in Medicine**, 35(3):368–381, 2016.

Eugene P Wigner. The unreasonable effectiveness of mathematics in the natural sciences. **Communications on pure and applied mathematics**, 13(1):1–14, 1960.

Leland Wilkinson. Dot plots. **The American Statistician**, 53(3):276, 1999.

Leland Wilkinson. **The Grammar of Graphics**. Springer, 2005.

Quin F Wills, Kenneth J Livak, Alex J Tipping, Tariq Enver, Andrew J Goldson, Darren W Sexton, and Chris Holmes. Single-cell gene expression analysis reveals genetic associations masked in whole-tissue experiments. **Nature biotechnology**, 31(8):748–752, 2013.

D Witten, R Tibshirani, S Gross, and B Narasimhan. Pma: Penalized multivariate analysis. **R package version**, 1(5), 2009a.

Daniela M Witten and Robert Tibshirani. Penalized classification using fisher's linear discriminant. **Journal of the Royal Statistical Society: Series B (Statistical Methodology)**, 73(5):753–772, 2011.

Daniela M Witten, Robert Tibshirani, and Trevor Hastie. A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. **Biostatistics**, page kxp008, 2009b.

Svante Wold, Arnold Ruhe, Herman Wold, and WJ Dunn, III. The collinearity problem in linear regression. the partial least squares (pls) approach to generalized inverses. **SIAM Journal on Scientific and Statistical Computing**, 5(3):735–743, 1984.

Erik S Wright. Decipher: harnessing local sequence context to improve protein multiple sequence alignment. **BMC bioinformatics**, 16(1):1, 2015.

CF Jeff Wu and Michael S Hamada. **Experiments: planning, analysis, and optimization**, volume 552. John Wiley & Sons, 2011.

Hongxiang Yu, Diana L Simons, Ilana Segall, Valeria Carcamo-Cavazos, Erich J Schwartz, Ning Yan, Neta S Zuckerman, Frederick M Dirbas, Denise L Johnson, Susan P Holmes, et al. Prc2/eed-ezh2 complex is up-regulated in breast cancer lymph node metastasis compared to primary tumor and correlates with tumor proliferation in situ. **PloS one**, 7(12):e51239, 2012.

Achim Zeileis, Christian Kleiber, and Simon Jackman. Regression models for count data in R. **Journal of Statistical Software**, 27(8), 2008. URL http://www.jstatsoft.org/v27/i08/.

Georg Zeller, Julien Tap, Anita Y Voigt, Shinichi Sunagawa, Jens Roat Kultima, Paul I Costea, Aurélien Amiot, Jürgen Böhm, Francesco Brunetti, Nina Habermann, Rajna Hercog, Moritz Koch, Alain Luciani, Daniel R Mende, Martin A Schneider, Petra Schrotz-King, Christophe Tournigand, Jeanne Tran Van Nhieu, Takuji Yamada, Jürgen Zimmermann, Vladimir Benes, Matthias Kloor, Cornelia M Ulrich, Magnus von Knebel Doeberitz, Iradj Sobhani, and Peer Bork. Potential of fecal microbiota for early-stage detection of colorectal cancer. **Molecular Systems Biology**, 10(11):766, 2014. doi: 10.15252/msb.20145645. URL `http://msb.embopress.org/content/10/11/766.abstract`.

Hui Zou and Trevor Hastie. **elasticnet: Elastic-Net for Sparse Estimation and Sparse PCA**, 2012. URL `https://CRAN.R-project.org/package=elasticnet`. R package version 1.1.

Hui Zou, Trevor Hastie, and Robert Tibshirani. Sparse principal component analysis. **Journal of computational and graphical statistics**, 15(2):265–286, 2006.