Symbols

* operator
  asterisk regex, 129–130, 134
  multiplication, 2, 43, 45, 50
  replication, 34–35
  unpacking, 38

** (power) operator, 2

*? (nongreedy asterisk) regex operator, 130–131, 134

\ (escape) prefix, 138, 139, 141, 145
\n (newline) character, 4, 22, 23, 130
\s (whitespace) character, 4, 145–148
\t (tab) character, 4

^ (not) regex operator, 140, 145–147
{} (instances) regex operator, 134, 135, 142

- operator
  negation, 2
  subtraction, 2, 43, 45

. (dot) regex operator, 129, 133–134

" (double quote), 4

''' (triple quote), 4

() (group) regex operator, 133–134, 135, 137, 138

% (modulo) operator, 2, 167

| operator
  or regex, 135, 144
  union, 164–165

+ operator
  addition, 2, 43, 45
  at-least-one regex, 134
  concatenation, 164–165

? (zero-or-one) regex operator, 130, 134, 139

?! (negative lookahead) regex operator, 149

?P (named group) regex operator, 145–147

' (single quote), 4

'' (triple quote), 4

/ (division) operator, 2, 43

// (integer division) operator, 2

[] operator
  character class regex, 138, 140–141
  indexing, 46
  list creation, 6

_ (throwaway) parameter, 175

_ (trailing underscore) character, 98

A

abs() function, 2, 72

absolute values, 72

activation functions, 107

addition (+) operator, 2, 43, 45

advanced indexing, 56, 67

Air Quality Index (AQI) outliers
  example, 53, 54–56

algorithms. See also classification algorithms

anagram detection, 152–154

binary search, 176–180

clustering algorithms, 94–97

Fibonacci series, 174–176

Levenshtein distance, 159–162

linear regression, 83–89

obfuscation, 165–168

outlier detection, 70, 73–74

palindrome detection, 154–156

permutations calculation, 156–159

powerset creation, 162–165

prime number generation, 168–174

and programming mastery, 151–152

Quicksort, 180–182

recursive, 157–159

runtime complexity, 154, 169, 177

all() function, 76

anagram detection example, 152–154

and keyword, 3–4

any() function, 36–37

append() list method, 7, 9, 22–23, 176

argsort() function, 88

arange() function, 64–65, 66–67

arithmetic operations, 2

arrays. See NumPy arrays
association analysis, 74–79
asterisk (*) regex operator, 129–130, 134
astype() function, 59, 69
at-least-one (+) regex operator, 134
autocorrection applications, 159
average() function, 44, 62–63, 117–119

B
bestseller books filtering example, 68–69
bestseller bundle association example, 77–79
bias-variance trade-off, 113–114
binary search algorithm, 176–180
Boolean data
array operations, 54, 58–59, 72–73, 76
as NumPy array data type, 50
values and evaluation, 2–4, 56, 143, 160–161, 176
Boolean indexing, 57–59, 69
bounce rates, 70
boundary cases, 123
brackets ([]),
character class regex operator, 138, 140–148
indexing operator, 46
list creation operator, 6
break keyword, 14
broadcasting
definition, 50
examples, 52–53, 54–56, 59, 61

C
Caesar’s cipher, 165
cardiac health cyclic data example, 33–35
categorical output, 90
centroids, 95
character class ([ ]) regex operator, 138, 140–141
character extraction example, 137–140
Christmas quote example, 135
classification algorithms
concepts, 120
and curse of dimensionality, 119
decision trees, 111–113
K-Nearest Neighbors, 100–104
logistic regression, 89–94
problem description, 89
support-vector machines, 119, 121–123
classifiers, 120
class labels, 93
close() (file) command, 23
cluster_centers_attribute, 97–99
clustering algorithms, 94–97
coefficients, 83–86
collaborative filtering, 74–79
collection data types, 9–10
column vectors, 88
compilation, 133–134
compile() method, 133
concatenation
+ operator, 164–165
list, 7, 33–35, 164–165
string, 4
conditional execution, 13
counter data structures, 6–12
dictionaries, 10–11
lists, 6–8
operations, 11–12
sets, 9–10
stacks, 8–9
cost, in list comprehension, 12, 18–20, 24
continue statement, 14
control flow, 12
if, else, and elif, 13
loops, 13–14
convergence, 109
copurchases association examples, 74–79
corrupted list correction example, 31–33
cyclic data generation example, 33–35

database formatting example, 37–39
data cleaning example, 60–64
data structures. See container data structures; data types; NumPy arrays
data types
Boolean, 2–4
None keyword, 4, 5–6
numerical, 2
and NumPy arrays, 50–51, 53, 59
strings, 4–5
dead code, 14
DecisionTreeClassifier module, 112–113
decision trees, 111–113, 123–126
def keyword, 14–15
dictionaries
  data structure, 10–11
  in employee data examples, 20, 36–37, 39
dimensionality
  curse of, 119
  and NumPy arrays, 42–43, 48–50
Divide and Conquer algorithms, 180
division (/) operator, 2, 43
dot (.) regex, 129, 133–134
double quote ("), 4
dtype property, 51, 53
duplicate character detection example, 145–147

e
edit distance, 159
element-wise operations, 43
e1f keyword, 13
e1se keyword, 13
employee data examples
  arithmetic, 45
  clustering, 97–99
  dictionary, 18, 20, 35–37
encryption, 165–166
endswith() string method, 5
ensemble learning, 123–126
error minimization, 85–86, 88
escape (\) prefix, 138, 139, 141, 145
expression, in list comprehension, 12, 18–20
extend() list method, 7

f
factorial calculation example, 156–159
false positives, 132
False value. See also Boolean data
  of Python objects, 160–161
  and while loops, 14
features and predictions, 82–83
Fibonacci series algorithm, 174–176
FIFO (first-in, first-out) structures, 8–9
file reading example, 22–24
filtering. See also association analysis, 68–69, 73–74
findall() function, 129–131, 135–137, 138, 142, 146–147
find() string method, 5, 28–29
Finxter ratings, 104–105, 109–110
fit() function
  and decision trees, 112–113
  and K-Nearest Neighbors (KNN) algorithm, 101–103
  and linear regression, 87–88
  and logistic regression, 92–93
  and neural network analysis, 108–109
  and random forests, 124–125
  and support-vector machines, 122
float data type and operations, 2, 50
float() function, 2
for loops, 12, 13–14, 18–20
fullmatch() function, 142–143, 144
functions. See also lambda functions;
  individual function names
    defined, 14–15
    throwaway parameter (_), 175
functools library, 163

g
generator expressions, 36–37
greedy pattern matching, 130–131
group () regex operator, 133–134, 135, 137, 138

h
Hadamard product, 45
hashable data types, 9–10, 12
hash() function, 9, 12
histogramming, 154
home price prediction example, 100–103
hyperlink analysis example, 136–137

i
if keyword, 12, 13, 19
income calculation example, 45–46
incrementor functions, 16
indexes
  [] operator, 46
  advanced indexing, 56, 67
  and argsort() function, 64–65
  as arguments, 27
  and Boolean arrays, 57–59, 69
index() list method, 8
inference phase, 83
initializer argument, 163–164
in keyword, 5, 11, 25
insert() list method, 7
Instagram influencer filtering example, 57–59
instances ({}) regex operator, 134, 135, 142
integer data type and operations, 2, 50
integer division (//) operator, 2
int() function, 2
investment portfolio risk example, 114–116
is keyword, 6
items() dictionary method, 11, 20
iterable arguments, 34
iterable (reduce()) argument, 163–164, 175

J
join() string method, 5, 166

K
(key, value) pairs, 10–11
keys() function, 11
K-Means algorithm, 95–99
KMeans module, 97–99
K-Nearest Neighbors (KNN) algorithm, 100–104
KNeighborsClassifier module, 103
KNeighborsRegressor module, 101–103

L
labeled vs. unlabeled data, 94–95
lambda functions
defining, 15–16, 24–26
recursive, 158–159, 160–162
lambda keyword, 15
len() function, 6
len() string method, 5
Levenshtein distance algorithm, 159–162
linear classifiers, 120
linear regression, 83–89
coding, 86–89
concepts and formulas, 83–86
LinearRegression module, 87
list comprehensionexamples, 22–24, 115, 139
formula, 12, 18–20
and generator expressions, 36
nested, 21–22
with slicing, 29–30
lists. See also list comprehensionconcatenation, 7, 33–35, 162–165
defining, 6
membership testing, 11
vs. NumPy arrays, 42, 43
operations on, 6–8
logical_and() function, 72–74
logistic regression, 89–94
LogisticRegression module, 92–93
loops, 13–14
lower() string method, 4
lung cancer logistic regression example, 90–94

M
machine learningbias-variance trade-off, 113–114
classification concepts, 120
decision trees, 111–113
ensemble learning, 123–126
K-Means clustering algorithm, 94–99
K-Nearest Neighbors algorithm, 100–104
linear regression algorithm, 83–89
logistic regression algorithm, 89–94
model parameters, 83
neural network analysis, 104–110
overview, 81, 126
supervised, 82–83
support-vector machines, 119, 121–123
unsupervised, 94–95
machine learning models
decision trees, 111–113
K-Means clustering algorithm, 94–99
K-Nearest Neighbors algorithm, 100–104
linear regression function, 83–89
logistic regression function, 89–94
neural networks, 104–110
parameters, 83
random forests, 123–126
support-vector machines, 119, 121–123
map() function, 25–26
margin of error, 121
margin of safety, 123
mark non-prime numbers example, 169–174
mark string example, 25–26
mask index arrays, 59
match() function, 133–134, 135–136
Matplotlib library, 34, 71–72
max() function, 44–45, 46, 79
maximum likelihood models, 91–92
max_iter() argument, 109
mean, 70–71, 73–74
mean() function, 73
meta-predictions, 123
min() function, 44, 115
minimum wage test example, 35–37
MLPRegressor module, 108–110
modulo (%) operator, 2, 167
multilayer perceptron (MLP), 104–110
multiline strings, 4, 130, 137, 140–141, 149–150
multinomial classification, 90
multiplication of arrays, 45, 50, 73
multiplication (*) operator, 2, 43, 45, 50
multiset data structures, 10
mutability, 6–7

N
named groups, 145–147
n_clusters argument, 98
ndim attribute, 48–49
negation (-) operator, 2
negative lookahead, 149–150
negative lookahead (!?) regex operator, 149
n_estimators parameter, 124–125
neural network analysis
coding, 108–110
concepts of artificial, 106–107
example, 104–105
newline (\n) character, 4, 22, 23, 130
None keyword, 4, 5–6
nongreedy asterisk (*) regex operator, 130–131, 134
nongreedy pattern matching, 130–131, 134, 137
nonlinear classifiers, 120
nonsecure URL search example, 140–141
nonzero() function, 54–56
normal distribution data, 70–71
normal() function, 71
not keyword, 3–4
not (^) regex operator, 140, 145–147
null value. See None keyword
numerical data types and operations, 2
NumPy arrays
arithmetic operations on, 43–46, 72
axes and dimensionality, 48–50
axis argument, 61–63, 65–66, 76
Boolean operations, 54–56
broadcasting, 50, 52–53, 54–56
creating, 42–43
and data types, 50–51, 53, 59
filtering, 68–69
indexing, 46, 57–59
logical and operation, 72–73
minimum variance calculation, 114–116
reshaping, 61, 62–63
slice assignments, 60–61, 62–63
slicing, 46–48, 51–52, 58–59, 75–76, 78
sorting in, 64–67
statistics calculations, 116–119
NumPy library, 41, 43

O
obfuscation algorithm, 165–168
one-liners
resources, xxiii
use and misuse, 183–184
value of learning, xix–xxii
or keyword, 3–4
order of execution
in Boolean operations, 3–4
in regular expressions, 135
or () regex operator, 135, 144
outlier detection, 53–57, 70, 73–74

P
palindrome detection example, 154–156
pattern matching. See regular expressions
permutations calculation example, 156–159
Peters, Tim, *The Zen of Python*, xxi–xxii
pivot element, 180–183
plot() function, 34–35
pop() list method, 9
power (**) operator, 2
powersets, 162–165
predict() function, 88, 108–110, 122, 125
predictions and features, 82–83
predict_proba() function, 93–94
prime numbers
detection example, 168–169
generator example, 169–174
probability, a priori, 157
programming skills
and algorithm mastery, 151–152
development and practice, xix–xxii, 116, 126, 183–184
problem solving strategies, 143
productivity, 39–40, 87, 127
in rating example, 104–105, 109–110
pruning, 112
Python
code readability, xxi–xxii, 24, 116
libraries, xix–xx, 26, 41, 71, 86, 87, 163
naming conventions, 98
object truth values, 160–161
resources, xxiii
skills rating example, 104–105, 109–110
regex. See regular expressions
regex functions, 135, 137, 142–143, 149
regression problems
vs. classification problems, 89
and K-Nearest Neighbors algorithm, 100–101
and linear regression algorithm, 83
regular expressions. See also regex
characters; regex functions
for character substitution, 149–150
compiled patterns, 133–134
for duplicate character detection, 145–147
false positives removal, 132–134
greedy and non-greedy pattern matching, 130
groups and named groups, 138–139, 145–146
negative lookahead, 149–150
special characters, 138
for user input validation, 141–145
for word repetition detection, 147–148
re module, 129–131
remove() list method, 7–8
replace() string method, 5
replication (*) operator, 34–35
reshape() function, 62–63, 88, 92–93, 101–103
return expressions, 15, 24–25
return keyword, 15
return values, 6, 24
reverse() list method, 8
ROTI3 algorithm, 165–168
salary increase calculation example, 51–53
SAT score analysis example, 66–67
scikit-learn library, 86, 97–98
search() function, 135, 147
sequence aggregator examples, 164–165, 175
set comprehension, 12
sets
data structure, 9–10, 56
membership testing, 11–12
powerset construction example, 162–165
Index

shape attribute, 49–50, 76
Sieve of Eratosthenes, 169–174
sigmoid function, 90–92
 single quote (‘), 4
sklearn package, 98
slice assignments, 31–33, 60–61
slicing
 with list comprehension, 29–30
 multidimensional, 46–48
 with negative step size, 66, 67, 155–156
 syntax and examples, 26–29
softmax function, 90
sorted (Python) function, 65, 66, 153–154
sort() (NumPy) function, 64–66, 67
sorting, 64–67, 153–154, 180–182
sort() list method, 8
split() function, 21–22
Stack Overflow, 170
stacks, 8–9
standard deviation, 70–71, 73–74, 117
start argument, 27, 155
startswith() string method, 5
statistics calculations, 116–119
std() function, 73, 117–119
step argument, 27
stock price examples
 calculations, 61–62
 linear regression, 84–89
stop argument, 27, 155
strings. See also multiline strings;
 regular expressions
 data type, 4
 selected methods, 4–5
strip() string method, 4, 22–24
str() string method, 4
sub() regex function, 149–150
subtraction (-) operator, 2, 43, 45
sum() function, 76, 77, 78
supervised machine learning, 82–83, 94
support-vector classification (SVC), 122
support-vector machines (SVMs), 119, 121–123
SVC module, 122

tab (\t) character, 4
team rankings example, 156–157
throwaway (_ ) parameter, 175
time format validation examples, 141–145
trailing underscore ( _ ) character, 98
training data, 82–83, 100
tree module, 112–113
trees. See decision trees
triple quote ("'"), 4
True value. See also Boolean data
do Python objects, 160–161
 and while loops, 14

U
union (|) operator, 164–165
unlabeled vs. labeled data, 94–95
unpacking (*) operator, 94–95
unsupervised machine learning, 94–95
upper() string method, 5
urllib.request module, 132
urlopen() method, 132
URL search example, 140–141
user input validation examples, 141–145

V
values() function, 11, 36–37
van Rossum, Guido, 36
var() function, 115, 117–119
variance, 113–116, 126

W
web scraper example, 132–134
where() function, 116
while loops, 13–14
whitespace (\s) character, 4, 145–148
word repetition detection example, 147–148

X
xkcd() function, 71–72

Z
Zen of Python, The (Peters), xxi–xxii
zero-or-one (?) regex operator, 130, 134, 139
zip() function, 37–39