Contents

List of Figures xi
List of Tables xiii
Preface xv
Acknowledgments xix

1 Introduction **1**
Ambitions of the twentieth century . . . 2
Pattern classification . . . 4
Prediction and action . . . 7
Chapter notes . . . 9

2 Fundamentals of Prediction **11**
Modeling knowledge . . . 13
Prediction via optimization . . . 16
Types of errors and successes . . . 20
The Neyman-Pearson Lemma . . . 22
Decisions that discriminate . . . 26
Chapter notes . . . 30

3 Supervised Learning **33**
Sample versus population . . . 33
Supervised learning . . . 34
A first learning algorithm: The perceptron . . . 37
Connection to empirical risk minimization . . . 38
Formal guarantees for the perceptron . . . 41
Chapter notes . . . 46

4 Representations and Features **49**
Measurement . . . 50
Quantization . . . 51
Template matching . . . 52
Summarization and histograms . . . 53
Nonlinear predictors . . . 54
Chapter notes . . . 65

5 Optimization **69**
Optimization basics . . . 70
Gradient descent . . . 71
Applications to empirical risk minimization . . . 74
Insights from quadratic functions . . . 76
Stochastic gradient descent . . . 78
Analysis of the stochastic gradient method . . . 84
Implicit convexity . . . 87
Regularization . . . 90
Squared loss methods and other optimization tools . . . 94
Chapter notes . . . 96

6 Generalization **99**
Generalization gap . . . 99
Overparameterization: Empirical phenomena . . . 100
Theories of generalization . . . 105
Algorithmic stability . . . 109
Model complexity and uniform convergence . . . 115
Generalization from algorithms . . . 118
Looking ahead . . . 123
Chapter notes . . . 123

7 Deep Learning **125**
Deep models and feature representation . . . 126
Optimization of deep nets . . . 128
Vanishing gradients . . . 134
Generalization in deep learning . . . 137
Chapter notes . . . 141

8 Datasets **143**
The scientific basis of machine learning benchmarks . . . 144
A tour of datasets in different domains . . . 145
Longevity of benchmarks . . . 156
Harms associated with data . . . 164
Toward better data practices . . . 169
Limits of data and prediction . . . 172
Chapter notes . . . 173

9 Causality **175**
The limitations of observation . . . 176
Causal models . . . 178
Causal graphs . . . 182
Interventions and causal effects . . . 184

Confounding . . . 186
Experimentation, randomization, potential outcomes . . . 189
Counterfactuals . . . 192
Chapter notes . . . 197

10 Causal Inference in Practice **199**
Design and inference . . . 200
The observational basics: Adjustment and controls . . . 201
Reductions to model fitting . . . 202
Quasi-experiments . . . 206
Limitations of causal inference in practice . . . 209
Chapter notes . . . 211

11 Sequential Decision Making and Dynamic Programming **213**
From predictions to actions . . . 214
Dynamical systems . . . 214
Optimal sequential decision making . . . 216
Dynamic programming . . . 217
Computation . . . 220
Partial observation and the separation heuristic . . . 225
Chapter notes . . . 230

12 Reinforcement Learning **231**
Exploration-exploitation trade-offs: Regret and PAC-error . . . 232
Unknown models and approximate dynamic programming . . . 241
Certainty equivalence is often optimal . . . 248
The limits of learning in feedback loops . . . 253
Chapter notes . . . 259

13 Epilogue **261**
Beyond pattern classification? . . . 264

14 Mathematical Background **265**
Common notation . . . 265
Multivariable calculus and linear algebra . . . 265
Probability . . . 267
Estimation . . . 272

Bibliography *275*
Index *295*

Chapter 1

Introduction

"Reflections on life and death of those who in Breslau lived and died" is the title of a manuscript that Protestant pastor Caspar Neumann sent to mathematician Gottfried Wilhelm Leibniz in the late seventeenth century. Neumann had spent years keeping track of births and deaths in his Polish hometown, now called Wrocław. Unlike sprawling cities like London and Paris, Breslau had a rather small and stable population with limited migration in and out. The parishes in town took due record of the newly born and deceased.

Neumann's goal was to find patterns in the occurrence of births and deaths. He thereby sought to dispel a persisting superstition that ascribed critical importance to certain climacteric years of age. Some believed it was at age 63, others that it was either at the 49th or the 81st year, that particularly critical events threatened to end the journey of life. Neumann recognized that his data defied the existence of such climacteric years.

Leibniz must have informed the Royal Society of Neumann's work. In turn, the Society invited Neumann in 1691 to provide the Society with the data he had collected. It was through the Royal Society that British astronomer Edmund Halley became aware of Neumann's work. A friend of Isaac Newton's, Halley had spent years predicting the trajectories of celestial bodies, but not those of human lives.

After a few weeks of processing the raw data through smoothing and interpolation, it was in the spring of 1693 that Halley arrived at what became known as Halley's life table.

At the outset, Halley's table displayed, for each year of age, the number of people of that age alive in Breslau at the time. Halley estimated that a total of approximately 34,000 people were alive, of which approximately 1,000 were between the ages zero and one, 855 were between ages one and two, and so forth.

Halley saw multiple applications of his table. One of them was to estimate the proportion of men in a population that could bear arms. To estimate this

Age. Curt.	Per- sons.	Age. Curt.	Per- sons	Age. Curt.	Per- sons	Age. Curt.	Per- sons	Age. Curt.	Per- sons	Age. Curt.	Per- sons
1	1000	8	680	15	628	22	586	29	539	36	481
2	855	9	670	16	622	23	579	30	531	37	472
3	798	10	661	17	616	24	573	31	523	38	463
4	760	11	653	18	610	25	567	32	515	39	454
5	732	12	646	19	604	26	560	33	507	40	445
6	710	13	640	20	598	27	553	34	499	41	436
7	692	14	634	21	592	28	546	35	490	42	427
Age. Curt	Per- sons.	Age. Curt.	Per- sons	Age. Curt.	Per- sons	Age Curt	Per- sons	Age. Curt.	Per- sons	Age. Curt.	Per- sons
43	417	50	346	57	272	64	202	71	131	78	58
44	407	51	335	58	262	65	192	72	120	79	49
45	397	52	324	59	252	66	182	73	109	80	41
46	387	53	313	60	242	67	172	74	98	81	34
47	377	54	302	61	232	68	162	75	88	82	28
48	367	55	292	62	222	69	152	76	78	83	23
49	357	56	282	63	212	70	142	77	68	84	20

Age.	Persons.
7	5547
14	4584
21	4270
28	3564
35	3604
42	3178
49	2709
56	2194
63	1694
70	1204
77	692
84	253
100	107
	34000
	Sum Total.

Figure 1.1: Halley's life table

proportion he computed the number of people between age 18 and 56, and divided by 2. The result suggested that 26% of the population were men neither too old nor too young to go to war.

At the same time, King William III of England needed to raise money for his country's continued involvement in the Nine Years War, raging from 1688 to 1697. In 1692, William turned to a financial innovation imported from Holland, the public sale of life annuities. A life annuity is a financial product that pays out a predetermined annual amount of money while the purchaser of the annuity is alive. The king had offered annuities at fourteen times the annual payout, a price too low for the young and too high for the old.

Halley recognized that his table could be used to estimate the odds that a person of a certain age would die within the next year. Based on this observation, he described a formula for pricing an annuity that, expressed in modern language, computes the sum of expected discounted payouts over the course of a person's life starting from their current age.

Ambitions of the twentieth century

Halley had stumbled upon the fact that prediction requires no physics. Unknown outcomes, be they future or unobserved, often follow patterns found in past observations. This empirical law would become the basis of consequential decision making for centuries to come.

On the heels of Halley and his contemporaries, the eighteenth century saw the steady growth of the life insurance industry. The industrial revolution fueled other forms of insurance sold to a population seeking safety in tumultuous

times. Corporations and governments developed risk models of increasing complexity with varying degrees of rigor. Actuarial science and financial risk assessment became major fields of study built on the empirical law.

Modern statistics and decision theory emerged in the late nineteenth and early twentieth century. Statisticians recognized that the scope of the empirical law extended far beyond insurance pricing, that it could be a method for both scientific discovery and decision making writ large.

Emboldened by advances in probability theory, statisticians modeled populations as probability distributions. Attention turned to what a scientist could say about a population by looking at a random draw from its probability distribution. From this perspective, it made sense to study how to decide between one of two plausible probability models for a population in light of available data. The resulting concepts, such as true positive and false positive, as well as the resulting technical repertoire, are in broad use today as the basis of hypothesis testing and binary classification.

As statistics flourished, two other developments around the middle of the twentieth century turned out to be transformational. The works of Turing, Gödel, and von Neumann, alongside dramatic improvements in hardware, marked the beginning of the computing revolution. Computer science emerged as a scientific discipline. General-purpose programmable computers promised a new era of automation with untold possibilities.

World War II spending fueled massive research and development programs on radar, electronics, and servomechanisms. Established in 1940, the United States National Defense Research Committee included a division devoted to control systems. The division developed a broad range of control systems, including gun directors, target predictors, and radar-controlled devices. The agency also funded theoretical work by mathematician Norbert Wiener, including plans for an ambitious anti-aircraft missile system that used statistical methods for predicting the motion of enemy aircraft.

In 1948, Wiener published his influential book *Cybernetics* around the same time as Shannon published *A Mathematical Theory of Communication*. Both proposed theories of information and communication, but their goals were different. Wiener's ambition was to create a new science, called cybernetics, that unified communications and control in one conceptual framework. Wiener believed that there was a close analogy between the human nervous system and digital computers. He argued that the principles of control, communication, and feedback could be a way not only to create mind-like machines, but also to understand the interaction of machines and humans. Wiener even went so far as to posit that the dynamics of entire social systems and civilizations could be understood and steered through the organizing principles of cybernetics.

The zeitgeist that animated cybernetics also drove ambitions to create artificial neural networks, capable of carrying out basic cognitive tasks. Cognitive concepts such as learning and intelligence had entered research conversations

about computing machines and with them came the quest for machines that learn from experience.

The 1940s were a decade of active research on artificial neural networks, often called connectionism. A 1943 paper by McCulloch and Pitts formalized artificial neurons and provided theoretical results about the universality of artificial neural networks as computing devices. A 1949 book by Donald Hebb pursued the central idea that neural networks might learn by constructing internal representations of concepts.

Pattern classification

Around the mid 1950s, it seemed that progress on connectionism had started to slow and would have perhaps tapered off had psychologist Frank Rosenblatt not made a striking discovery.

Rosenblatt had devised a machine for image classification. Equipped with 400 photosensors, the machine could read an image composed of 20 by 20 pixels and sort it into one of two possible classes. Mathematically, the perceptron computes a linear function of its input pixels. If the value of the linear function applied to the input image is positive, the perceptron decides that its input belongs to class 1, otherwise class -1. What made the perceptron so successful was the way it could learn from examples. Whenever it misclassified an image, it would adjust the coefficients of its linear function via a local correction.

Rosenblatt observed in experiments what would soon be a theorem. If a sequence of images could at all be perfectly classified by a linear function, the perceptron would only make so many mistakes on the sequence before it correctly classified all images it encountered.

Rosenblatt developed the perceptron in 1957 and continued to publish on the topic in the years that followed. The perceptron project was funded by the US Office of Naval Research, which jointly announced the project with Rosenblatt at a press conference in 1958 that led the *New York Times* to exclaim:

> The Navy revealed the embryo of an electronic computer that it expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence.[1]

This development sparked significant interest in perceptrons and reinvigorated neural network research throughout the 1960s. By all accounts, the research in the decade that followed Rosenblatt's work had essentially all the ingredients of what is now called machine learning, specifically, supervised learning.

Practitioners experimented with a range of different features and model architectures, moving from linear functions to perceptrons with multiple layers,

the equivalent of today's deep neural networks. A range of variations of the optimization method and different ways of propagating errors came and went.

Theory followed closely behind. Not long after the invention came a theorem, called mistake bound, that gave an upper bound on the number of mistakes the perceptron would make in the worst case on any sequence of labeled data points that can be fit perfectly with a linear separator.

Today, we recognize the perceptron as an instance of the stochastic gradient method applied to a suitable objective function. The stochastic gradient method remains the optimization workhorse of modern machine learning applications.

Shortly after the well-known mistake bound came a lesser known theorem. The result showed that when the perceptron succeeded in fitting training data, it would also succeed in classifying unseen examples correctly provided that these were drawn from the same distribution as the training data. We call this *generalization*: finding rules consistent with available data that apply to instances we have yet to encounter.

By the late 1960s, these ideas from perceptrons had solidified into a broader subject called *pattern recognition* that encompassed most of the concepts we consider core to machine learning today. In 1939, Wald formalized the basic problem of classification as one of optimal decision making when the data is generated by a known probabilistic model. Researchers soon realized that pattern classification could be achieved using data alone to guide prediction methods such as perceptrons, nearest neighbor classifiers, and density estimators. The connections with mathematical optimization including gradient descent and linear programming also took shape during the 1960s.

Pattern classification—today more popularly known as supervised learning—built on statistical tradition in how it formalized the idea of generalization. We assume observations come from a fixed data generating process, such as samples drawn from a fixed distribution. In a first optimization step, called training, we fit a model to a set of data points labeled by class membership. In a second step, called testing, we judge the model by how well it performs on newly generated data from the very same process.

This notion of generalization as performance on fresh data can seem mundane. After all, it simply requires the classifier to do, in a sense, more of the same. We require consistent success on the same data generating process as encountered during training. Yet the seemingly simple question of what theory underwrites the generalization ability of a model has occupied the machine learning research community for decades.

Pattern classification, once again

Machine learning as a field, however, is not a straightforward evolution of the pattern recognition of the 1960s, at least not culturally and not historically.

After a decade of perceptron research, a group of influential researchers, including McCarthy, Minsky, Newell, and Simon, put forward a research program by the name of artificial intelligence. The goal was to create human-like intelligence in a machine. Although the goal itself was in many ways not far from the ambitions of connectionists, the group around McCarthy fancied entirely different formal techniques. Rejecting the numerical pattern fitting of the connectionist era, the proponents of this new discipline saw the future in symbolic and logical manipulation of knowledge represented in formal languages.

Artificial intelligence became the dominant academic discipline to deal with cognitive capacities of machines within the computer science community. Pattern recognition and neural networks research continued, albeit largely outside artificial intelligence. Indeed, journals on pattern recognition flourished during the 1970s.

During this time, artificial intelligence research led to a revolution in *expert systems*, logic- and rule-based models that had significant industrial impact. Expert systems were hard coded and left little room for adapting to new information. AI researchers interested in such adaptation and improvement—learning, if you will—formed their own subcommunity, beginning in 1981 with the first International Workshop on Machine Learning. The early work from this community reflects the logic-based research that dominated artificial intelligence at the time; the papers read as if of a different field than what we now recognize today as machine learning research. It was not until the late 1980s that machine learning began to look more like pattern recognition, once again.

Personal computers had made their way from research labs into home offices across wealthy nations. Internet access, if slow, made email a popular form of communication among researchers. File transfer over the internet allowed researchers to share code and datasets more easily.

Machine learning researchers recognized that in order for the discipline to thrive it needed a way to more rigorously evaluate progress on concrete tasks. Whereas in the 1950s it had seemed miraculous enough if training errors decreased over time on any nontrivial task, it was clear now that machine learning needed better benchmarks.

In the late 1980s, the first widely used benchmarks emerged. Then graduate student David Aha created the UCI machine learning repository that made several datasets widely available via FTP. Aiming to better quantify the performance of AI systems, the Defense Advanced Research Projects Agency (DARPA) funded a research program on speech recognition that led to the creation of the influential TIMIT speech recognition benchmark.

These benchmarks had the data split into two parts, one called training data, one called testing data. This split elicits the promise that the learning algorithm will only access the training data when it fits the model. The testing

data is reserved for evaluating the trained model. The research community can then rank learning algorithms by how well the trained models perform on the testing data.

Splitting data into training and testing sets was an old practice, but the idea of reusing such datasets as benchmarks was novel and transformed machine learning. The *dataset-as-benchmark paradigm* caught on and became core to applied machine learning research for decades to come. Indeed, machine learning benchmarks were at the center of the most recent wave of progress on deep learning. Chief among them was ImageNet, a large repository of images, labeled by nouns of objects displayed in the images. A subset of roughly 1 million images belonging to 1,000 different object classes was the basis of the ImageNet Large Scale Visual Recognition Challenge. Organized from 2010 until 2017, the competition became a striking showcase for performance of deep learning methods for image classification.

Increases in computing power and volume of available data were key driving factors for progress in the field. But machine learning benchmarks did more than provide data. Benchmarks gave researchers a way to compare results, share ideas, and organize communities. They implicitly specified a problem description and a minimal interface contract for code. Benchmarks also became a means of knowledge transfer between industry and academia.

The most recent wave of machine learning as pattern classification was so successful, in fact, that it became the new artificial intelligence in the public narrative of popular media. The technology reached entirely new levels of commercial significance with companies competing fiercely over advances in the space.

This new artificial intelligence had done away with the symbolic reasoning of the McCarthy era. Instead, the central drivers of progress were widely regarded as growing datasets, increasing compute resources, and more benchmarks along with publicly available code to start from. Are those then the only ingredients needed to secure the sustained success of machine learning in the real world?

Prediction and action

Unknown outcomes often follow patterns found in past observations. But what do we do with the patterns we find and the predictions we make? Like Halley proposing his life table for annuity pricing, predictions only become useful when they are acted upon. But going from patterns and predictions to successful actions is a delicate task. How can we even anticipate the effect of a hypothetical action when our actions now influence the data we observe and value we accrue in the future?

One way to determine the effect of an action is experimentation: try it out

and see what happens. But there's a lot more we can do if we can model the situation more carefully. A model of the environment specifies how an action changes the state of the world, and how in turn this state results in a gain or loss of utility. We include some aspects of the environment explicitly as variables in our model. Others we declare *exogenous* and model as noise in our system.

The solution of how to take such models and turn them into plans of action that maximize expected utility is a mathematical achievement of the twentieth century. By and large, such problems can be solved by *dynamic programming*. Initially formulated by Bellman in 1954, dynamic programming poses optimization problems where at every time step, we observe data, take an action, and pay a cost. By chaining these steps together in time, elaborate plans can be made that remain optimal under considerable stochastic uncertainty. These ideas revolutionized aerospace in the 1960s, and are still deployed in infrastructure planning, supply chain management, and the landing of SpaceX rockets. Dynamic programming remains one of the most important algorithmic building blocks in the computer science toolkit.

Planning actions under uncertainty has also always been core to artificial intelligence research, though initial proposals for sequential decision making in AI were more inspired by neuroscience than operations research. In 1950-era AI, the main motivating concept was one of *reinforcement learning*, which posited that one should encourage taking actions that were successful in the past. This reinforcement strategy led to impressive game-playing algorithms like Samuel's Checkers Agent circa 1959. Surprisingly, it wasn't until the 1990s that researchers realized that reinforcement learning methods were approximation schemes for dynamic programming. Powered by this connection, a mix of researchers from AI and operations research applied neural nets and function approximation to simplify the approximate solution of dynamic programming problems. The subsequent 30 years have led to impressive advances in reinforcement learning and approximate dynamic programming techniques for playing games, such as Go, and in powering dexterous manipulation in robotic systems.

Central to the reinforcement learning paradigm is understanding how to balance learning about an environment and acting on it. This balance is a nontrivial problem even in the case where actions do not lead to a change in state. In the context of machine learning, experimentation in the form of taking an action and observing its effect often goes by the name *exploration*. Exploration reveals the payoff of an action, but it comes at the expense of not taking an action that we already knew had a decent payoff. Thus, there is an inherent trade-off between exploration and *exploitation* of previous actions. Though in theory the optimal balance can be computed by dynamic programming, it is more common to employ techniques from *bandit optimization* that are simple and effective strategies to balance exploration and exploitation.

Not limited to experimentation, causality is a comprehensive conceptual framework to reason about the effect of actions. Causal inference, in principle, allows us to estimate the effect of hypothetical actions from observational data. A growing technical repertoire of causal inference is taking various sciences by storm, as witnessed in epidemiology, political science, policy, climate, and development economics.

There are good reasons that many see causality as a promising avenue for making machine learning methods more robust and reliable. Current state-of-the-art predictive models remain surprisingly fragile to changes in the data. Even small natural variations in a data-generating process can significantly deteriorate performance. There is hope that tools from causality could lead to machine learning methods that perform better under changing conditions.

However, causal inference is no panacea. There are no causal insights without making substantive judgments about the problem that are not verifiable from data alone. The reliance on hard earned substantive domain knowledge stands in contrast with the nature of recent advances in machine learning that largely did without it—and that was the point.

Chapter notes

Halley's life table has been studied and discussed extensively; for an entry point, see recent articles by Bellhouse[2] and Ciecka,[3] or the article by Pearson and Pearson.[4]

Halley was not the first to create a life table. In fact, what Halley created is more accurately called a population table. Instead, John Grount deserves credit for the first life table in 1662 based on mortality records from London. Considered to be the founder of demography and an early epidemiologist, Grount's work was in many ways more detailed than Halley's fleeting engagement with Breslau's population. However, to Grount's disadvantage the mortality records released in London at the time did not include the age of the deceased, thus complicating the work significantly.

Mathematician de Moivre picked up Halley's life table in 1725 and sharpened the mathematical rigor of Halley's idea. A few years earlier, de Moivre had published the first textbook on probability theory called *The Doctrine of Chances: A Method of Calculating the Probability of Events in Play*. Although de Moivre lacked the notion of a probability distribution, his book introduced an expression resembling the normal distribution as an approximation to the binomial distribution, what was in effect the first central limit theorem. The time of Halley coincides with the emergence of probability. Hacking's book provides much additional context, particularly relevant are Chapter 12 and 13.[5]

For the history of feedback, control, and computing before cybernetics, see the excellent text by Mindell.[6] For more on the cybernetics era itself, see the

books by Kline[7] and Heims.[8] See Beniger[9] for how the concepts of control and communication and the technology from that era led to the modern information society.

The prologue from the 1988 edition of *Perceptrons* by Minsky and Papert presents a helpful historical perspective. The recent 2017 reprint of the same book contains additional context and commentary in a foreword by Léon Bottou.

Much of the first International Workshop on Machine Learning was compiled in an edited volume, which summarizes the motivations and perspectives that seeded the field.[10] Langley's article provides helpful context on the state of evaluation in machine learning in the 1980s and how the desire for better metrics led to a renewed emphasis on pattern recognition.[11] Similar calls for better evaluation motivated the speech transcription program at DARPA, leading to the TIMIT dataset, arguably the first machine learning benchmark dataset.[12,13,14]

It is worth noting that the Parallel Distributed Processing Research Group led by Rummelhart and McLeland actively worked on neural networks during the 1980s and made extensive use of the rediscovered back-propagation algorithm, an efficient algorithm for computing partial derivatives of a circuit.[15]

A recent article by Jordan provides an insightful perspective on how the field came about and what challenges it still faces.[16]

Index

A/B test, 201
adaptive analyst, 157
adaptive tree, 158, 162
adaptivity, 156
adjustment estimator, 201
adjustment formula, 186, 201
admissible variable, 201
algorithmic stability, 109
analyst, 157
ATE, 199
average stability, 110
average treatment effect, 186, 199

backdoor criterion, 187
backpropagation, 130
bag of words, 54
bandits, 233
basis function, 57
Bayes' rule, 18, 270
belief propagation, 228
Berkeley, 176
Berkson's law, 184
Bernoulli, 269
bias, 164
boosting, 117

causal effect, 184, 186
causal forest, 205
causal graph, 182
causal model, 178
central limit theorem, 106
certainty equivalence, 231, 242
chain rule, 129, 132, 267
Chebyshev's inequality, 107
class, 16
classifier, 30
collider, 184
common cause, 183
COMPAS, 29
competition, 144
concentration inequalities, 107
condition number, 77
conditional average treatment effect, 205
conditional probability, 270
confounder, 183
confounding, 183, 186, 187, 199
confusion table, 20
consistency, 197
contextual bandits, 239
continuous control, 230
control, 229
control action, 214
control group, 191
convex function, 71
convexity, 71, 111, 136
convolution, 53
copyright, 166
counterfactual, 192, 194, 195
covariance, 269

data augmentation, 104
data generating process, 181, 200
datasheets, 169
deanonymization, 166

decision, 30
decision making, 26
decision rule, 16, 26, 30
deep reinforcement learning, 248
descent direction, 71
design, 200
detection theory, 30
differences in differences, 206
direct cause, 181
direct effect, 181
discount factor, 219
discrimination, 27
disparate impact, 28
disparate treatment, 28
do-operator, 180, 185
double machine learning, 204
downsampling, 54
dynamic programming, 213, 218
dynamical system, 214

eigenvalue, 266
empirical risk, 35
empirical risk minimization, 35, 36
ensemble, 117
ensembling, 157
estimation, 229, 272
exogenous factors, 181
expectation, 269
explaining away, 184
explicit regularization, 91
explore-then-commit, 235, 240

F1-score, 21
fairness, 28
false discovery rate, 21
false negative rate, 21
false positive rate, 21
feature, 49
filtering, 227
fork, 183
Fourier transform, 53, 64
Fragile Families Challenge, 172

Gaussian, 15, 60, 270
Gaussian kernel, 60, 64
generalization, 36, 273
generalization gap, 36, 99, 144, 159
generative model, 15
global minimizer, 70
gradient, 39, 266
gradient descent, 69
greedy algorithm, 241

harm, 164
heterogeneous treatment effect, 205
hidden Markov model, 228
hinge loss, 39, 75
histogram, 53
Hoeffding's inequality, 107
holdout method, 144
human subjects, 50

i.i.d. assumption, 34, 45
ignorability, 197, 202
ILSVRC, 155
ImageNet, 154
implicit convexity, 70, 87
implicit regularization, 91
indirect cause, 183
individual treatment effect, 189
inference, 200
instrumental variables, 206
interference, 211
intervention, 184

Jacobian, 267

Kaggle, 144
Kalman filtering, 228
kernel, 57
kernel function, 58
kernel methods, 93
knowledge, 13, 178, 200, 210

label, 16
Ladder algorithm, 162
leaderboard, 153, 165
leaderboard error, 161

leaderboard principle, 161
least squares, 76, 274
lifts, 56
likelihood function, 15
likelihood ratio, 18
likelihood ratio test, 18
linear function, 54, 222
linear quadratic regulator, 221, 248, 254
linear separator, 37
local minimizer, 70, 71
logistic loss, 75
logistic regression, 75
loss function, 16, 69, 74, 99, 111
LQR, 221, 223, 228, 229, 248

MAP, 20, 76
margin, 37, 38, 42, 92
margin bounds, 117
margin error, 117
margin mistake, 38
Markov decision process, 216
Markov's inequality, 106
maximum a posteriori, 20, 76
MDP, 216
measurement, 50, 169
measurement construct, 50, 170
measurement procedure, 170
mediator, 183
minimizer, 70
minimum error rule, 12, 14
mistake bound, 42
MNIST, 151
model, 13, 178, 213
model parameters, 39
model predictive control, 223
model-error theorem, 250
MTurk, 154
multi-armed bandits, 233

Netflix Prize, 165
neural network, 57, 61, 101, 103
Neyman-Pearson, 22
NIST, 152
nonconvex, 89
nonlinearity, 61

observational methods, 210
observational study, 210
one-hot encoding, 52
optimal control, 215
optimism, 238
overfitting, 101, 145
overlap, 202
overparameterization, 56, 89, 100, 101, 103, 124, 129, 156

PAC, 232
parameter estimation, 272
parent, 181
perceptron, 46
perceptron algorithm, 38
plug-in estimator, 272
policy gradient, 246
policy iteration, 222
policy search, 244
polynomial kernel, 59
polynomials, 55
POMDP, 226–228, 256
pooling, 54
positive semidefinite, 59, 265
positivity, 202
posterior, 18, 20
potential outcomes, 189, 195, 202
pre-training, 155
precision, 21
prediction, 11, 14
predictor, 16, 35
privacy, 165
probability, 267
probability density, 14, 268
probability distribution, 13
probably approximately correct, 232
propensity score, 203
ProPublica, 29

Q-function, 218–220, 222

Q-learning, 243
quadratic function, 55
quantization, 51
quasi-experiments, 206

Rademacher complexity, 116
random features, 63
random search, 247
randomized controlled trial, 191, 210
receiver operating characteristic, 22
rectified linear unit, 61
regression discontinuity, 206
regret, 232, 233
regularization, 40, 90, 104, 113
regularized ERM, 113
REINFORCE, 245
reinforcement learning, 213, 231
reliability, 171
ReLU, 61
representational harm, 164
representer theorem, 93
reward, 214
Riccati Equation, 222
ridge regression, 95
risk, 17, 35
ROC, 22

separation principle, 227
sequential decision making, 213
sigmoidal function, 61
spillover, 211
squared loss, 41, 76, 94
stability, 45, 92
state, 214
statistical model, 13, 14, 213
stochastic gradient method, 39, 78
structural causal model, 181
subgradient, 40
successive elimination, 237
summarization, 53
support vector machine, 75
surrogate loss, 41
SUTVA, 196, 211

tabular MDP, 220, 248
target variable, 170
Taylor series, 59
Taylor's theorem, 266
template matching, 52
tensor, 51
Tikhonov regularization, 113
TIMIT, 146
translation, 53
treatment, 195, 199
treatment group, 191
true negative rate, 21
true positive rate, 21

UCB, 238
UCI, 148
unconfoundedness, 202
underfitting, 101
uniform convergence, 115
uniform stability, 111
unit, 194, 195, 211
universal approximation, 61
unobserved confounding, 187
upper confidence bound, 238

validity, 171, 210
value iteration, 222
VC dimension, 115

weight decay, 40, 113
word embedding, 165
WordNet, 154

zero-one loss, 41, 74