BAYESUVIUS

A VISUAL DICTIONARY OF BAYESIAN NETWORKS AND CAUSAL INFERENCE

ROBERT R. TUCCI

Bayesuvius, a visual dictionary of Bayesian Networks and Causal Inference

Robert R. Tucci
www.ar-tiste.xyz

November 11, 2021

This book is constantly being expanded and improved. To download the latest version, go to `https://github.com/rrtucci/Bayesuvius`

Bayesuvius
by Robert R. Tucci
Copyright ©2020-2021, Robert R. Tucci.

This work is licensed under the Creative Commons Attribution-Noncommercial-No Derivative Works 3.0 United States License. To view a copy of this license, visit the link `https://creativecommons.org/licenses/by-nc-nd/3.0/` or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042.

Figure 1: View of Mount Vesuvius from Pompeii

Figure 2: Mount Vesuvius and Bay of Naples

Contents

Foreword **13**

Navigating the ocean of Judea Pearl's Books **14**

CI-2-3 track **15**

Notational Conventions and Preliminaries **17**

0.1 Some abbreviations frequently used throughout this book 17
0.2 $\mathcal{N}(!a)$ 17
0.3 One hot 17
0.4 Special sets 17
0.5 Kronecker delta function 18
0.6 Dirac delta function 18
0.7 Indicator function (aka Truth function) 18
0.8 Majority function 18
0.9 Underlined letters indicate random variables 18
0.10 Probability distributions 19
0.11 Discretization of continuous probability distributions 19
0.12 Samples, i.i.d. variables 19
0.13 Expected Value and Variance 20
0.14 Conditional Expected Value 21
0.15 Notation for covariances 21
0.16 Conditional Covariance 22
0.17 Law of Total Variance 22
0.18 Normal Distribution 23
0.19 Uniform Distribution 24
0.20 Softmax function (aka Boltzmann Distribution) 24
0.21 Sigmoid and log-odds functions 25
0.22 Estimand, Estimator, Bias 26
0.23 Maximum Likelihood Estimate, Likelihood Ratio Test 26
0.24 Mean Square Error (MSE) 27
0.25 Bayes Rule, Bayesian Updating And Conjugate Priors 30
0.26 Linear regression, Ordinary Least Squares (OLS) 30

0.26.1 LR, assuming x_σ are non-random . . . 32
Derivation of LR From Minimization of Error . . . 32
Geometry of LR with non-random x_σ. . . . 33
LR Goodness of Fit, R^2 . . . 34
0.26.2 LR, assuming x_σ are random . . . 36
Transforming expressions from non-random to random x_σ . . 37
Geometry of LR with random x_σ . . . 38
Regression interpreted as differentiation of y . . . 40
0.27 Entropy, Kullback-Liebler divergence, Cross-Entropy . . . 41
0.28 Definition of various entropies used in Shannon Information Theory . 42
0.29 Pearson Chi-Squared Test . . . 43
0.30 Demystifying Population and Sample Variances . . . 44
0.31 Independence of $\hat{\mu}$ and $\hat{\sigma^2}$ . . . 45
0.32 Chi-square distribution . . . 47
0.33 Student's t-distribution . . . 47
0.34 Hypothesis testing and 3 classic test statistics (Likelihood, Score, Wald) 50
0.35 Error Bars . . . 54
0.36 Short Summary of Boolean Algebra . . . 56

Definition of a Bayesian Network **58**

Bayesian Networks and the Passage of Time **61**

1 AdaBoost **62**
1.1 AdaBoost for general ensemble of w-classifiers . . . 62
1.2 AdaBoost for ensemble of tree stumps . . . 66

2 ARACNE structure learning **68**

3 Backdoor Adjustment Formula **70**
3.1 Examples . . . 71

4 Back Propagation (Automatic Differentiation) **74**
4.1 General Theory . . . 74
4.1.1 Jacobians . . . 74
4.1.2 Bnets for function composition, forward propagation and back propagation . . . 75
4.2 Application to Neural Networks . . . 76
4.2.1 Absorbing b_i^λ into $w_{i|j}$. . . . 76
4.2.2 Bnets for function composition, forward propagation and back propagation for NN . . . 78
4.3 General bnets instead of Markov chains induced by layered structure of NNs . . . 81

5 Basic Curve Fitting Using Gradient Descent **82**

6 Bell and Clauser-Horne Inequalities in Quantum Mechanics **84**

7 Berkson's Paradox **85**

8 Binary Decision Diagrams **87**

9 Chow-Liu Trees and Tree Augmented Naive Bayes (TAN) **91**
9.1 Chow-Liu Trees 91
9.2 Tree Augmented Naive Bayes (TAN) 95

10 Counterfactual Reasoning **97**
10.1 The 3 Rungs of Causal AI 97
10.2 Do operator 98
10.3 Imagine operator 98

11 Cross-Validation **101**

12 Decision Trees **104**
12.1 Transforming a dtree into a bnet 106
12.2 Structure Learning for Dtrees 107
12.2.1 Information Gain, Gini 108
12.3 Information Gain Ratio 111
12.3.1 Pseudo-code 111

13 Difference-in-Differences **114**
13.1 John Snow, DID and a cholera transmission pathway 114
13.2 PO analysis 116
13.3 Linear Regression 118

14 Digital Circuits **121**
14.1 Mapping any dcircuit to a bnet 122
14.1.1 Option A of Fig.14.2 122
14.1.2 Option B of Fig.14.2 122

15 Do Calculus **123**
15.1 Parent Adjustment Formula 126
15.2 3 Rules of Do Calculus 127
15.3 Backdoor Adjustment Formula 128
15.4 Frontdoor Adjustment Formula 130
15.5 Do operator for DEN diagrams 131

16 Do Calculus proofs **134**

17 D-Separation **148**

18 D-Separation and Do Calculus Rules for LDEN: COMING SOON **151**

19 D-Separation in Quantum Mechanics **152**

20 Dynamic Bayesian Networks **153**

21 Expectation Maximization **155**
21.1 The EM algorithm: . . . 156
21.1.1 Motivation . . . 157
21.2 Minorize-Maximize (MM) algorithms . . . 158
21.3 Examples . . . 159
21.3.1 Gaussian mixture . . . 159
21.3.2 Blood Genotypes and Phenotypes . . . 160
21.3.3 Missing Data/Imputation . . . 162

22 Frisch-Waugh-Lovell (FWL) theorem **163**
22.1 FWL, assuming x^σ are non-random . . . 163
22.2 FWL, assuming x^σ are random . . . 164

23 Frontdoor Adjustment Formula **167**
23.1 Examples . . . 168

24 Gaussian Nodes with Linear Dependence on Parents **169**

25 Generative Adversarial Networks (GANs) **172**

26 Goodness of Causal Fit **177**
26.1 Abstract . . . 177
26.2 Introduction . . . 177
26.3 Goodness of Fit . . . 178
26.4 GCF example 1 . . . 179
26.5 GCF example 2 . . . 181
26.6 GCF in general . . . 182

27 Granger Causality **183**

28 Hidden Markov Model **186**
28.1 Calculating $P(x_t, v^n)$ and $P(x_t, x_{t+1}, v^n)$ . . . 188
28.2 Calculating $\mathcal{F}_t$ and $\overline{\mathcal{F}}_t$ . . . 189
28.3 Calculating $P(x^n|v^n)$ . . . 190
28.4 Calculating $P(v^n|A, B, \pi)$ . . . 191
28.5 Calculating $\hat{x}^n$ (Viterbi algorithm) . . . 192

28.6 Calculating $\hat{A}$, $\hat{B}$, $\hat{\pi}$ (Baum-Welch algorithm) 193

29 Influence Diagrams & Utility Nodes **196**

30 Instrumental Inequality and beyond **198**
30.1 I-inequality 198
30.1.1 I-inequality for binary z,d,y 200
30.2 Bounds on Effect of IV on treatment outcome y 201

31 Instrumental Variables **204**
31.1 δ with unmeasured confounder 204
31.2 δ (with unmeasured confounder) can be inferred via IV 205
31.3 More general bnets with IVs 206
31.4 Instrumental Inequality 207

32 Jackknife Resampling **208**
32.1 Case $A = A^n(\vec{x}) = \frac{1}{n}\sum_\sigma x^\sigma$ 210

33 Junction Tree Algorithm **212**

34 Kalman Filter **213**
34.1 Prediction Problem 214
34.2 Solution 215
34.3 Simple Example 216
34.4 Invariants 217
34.5 Derivation of Solution 217

35 Linear and Logistic Regression **219**
35.1 Generalization to x with multiple components (features) 221
35.2 Alternative $V(b, m)$ for logistic regression 221

36 Linear Deterministic Bnets with External Noise **223**
36.1 Example of LDEN diagram 223
36.2 Fully Connected LDEN diagrams 224
36.2.1 Fully connected LDEN diagram with $nx = 2$ 225
36.2.2 Fully connected LDEN diagram with $nx = 3$ 225
36.2.3 Fully connected LDEN diagram with arbitrary nx 226
36.3 Non-linear DEN diagrams 228

37 Markov Blankets **229**

38 Markov Chain Monte Carlo (MCMC) 231
38.1 Inverse Cumulative Sampling . . . 231
38.2 Rejection Sampling . . . 233
38.3 Metropolis-Hastings Sampling . . . 234
38.4 Gibbs Sampling . . . 237
38.5 Importance Sampling . . . 238

39 Markov Chains 240

40 Mediation Analysis 241

41 Message Passing (Belief Propagation) 247
41.1 Distributed Soldier Counting . . . 247
41.2 Spring Systems . . . 249
41.3 BP for Markov Chains . . . 249
41.4 BP Algorithm for Polytrees . . . 256
41.4.1 How BP algo for polytrees reduces to the BP algo for Markov chains . . . 260
41.5 Derivation of BP Algorithm for Polytrees . . . 261
41.6 Example of BP algo for a Tree . . . 264
41.7 Bipartite bnets . . . 268
41.8 BP for bipartite bnets (BP-BB) . . . 271
41.8.1 BP-BB and general BP agree on Markov chains . . . 272
41.8.2 BP-BB and general BP agree on tree bnets. . . . 274
41.9 BP-BB and sum-product decomposition . . . 276

42 Message Passing (Belief Propagation) in Quantum Mechanics 277

43 Meta-learners for estimating ATE 278

44 Missing Data, Imputation 282
44.1 Imputation via EM . . . 283
44.2 Imputation via MCMC . . . 286
44.3 Multiple Imputations . . . 287

45 Monty Hall Problem 288

46 Multi-armed Bandits 290
46.1 Bnet for MAB . . . 291
46.2 Reward functions . . . 293
46.3 Regret functions . . . 295
46.4 Strategies with random exploration . . . 295
46.4.1 ϵ-greedy algorithm . . . 296
46.4.2 ϵ_t-greedy algorithm . . . 297

46.5 Strategies with nonrandom exploration . . . 297
46.5.1 Upper Confidence Bounds (UCB) algorithms . . . 297
Frequentist UCB (UCB1) algorithm . . . 297
Bayesian UCB algorithm . . . 298
46.5.2 Thompson Sampling MAB (TS-MAB) algorithm . . . 299
Bnet for general TS-MAB algorithm . . . 299
TS-MAB algorithm with Beta agent and Bernoulli environment 301
TS-MAB algorithm, skeletal reprise . . . 302
46.5.3 Grad-MAB algorithm . . . 303

47 Naive Bayes **306**

48 Neural Networks **307**
48.1 Activation Functions $\mathcal{A}_i^\lambda : \mathbb{R} \to \mathbb{R}$ . . . 308
48.2 Weight optimization via supervised training and gradient descent . . 309
48.3 Non-dense layers . . . 311
48.4 Autoencoder NN . . . 313

49 Noisy-OR gate **314**
49.1 3 ways to interpret the parameters π_i . . . 315

50 Non-negative Matrix Factorization **318**
50.1 Bnet interpretation . . . 318
50.2 Simplest recursive algorithm . . . 319

51 Observationally Equivalent DAGs **320**
51.1 Examples . . . 320

52 Potential Outcomes **323**
52.1 G and G_{den} bnets, the starting point bnets . . . 324
52.2 G bnet with nodes $y^\sigma(0), y^\sigma(1)$ added to it. . . . 326
52.3 Expected Values of treatment outcome y^σ . . . 328
52.4 Translation Dictionary . . . 328
52.5 $\mathcal{Y}_{|d,x} = \mathcal{Y}_{d|d,x}$ (SUTVA) . . . 329
52.6 Conditional Independence Assumption (CIA) . . . 330
52.7 Treatment Effects . . . 330
52.8 Insights into what makes treatment effects equal and $\mathcal{Y}_{1|0} = \mathcal{Y}_1$ . . . 333
52.9 G_{do+} bnet . . . 334
52.10 $ACE = ATE$ . . . 335
52.11 Good, Bad Controls . . . 336
52.12 PO Confounder Sensitivity Analysis . . . 337
52.13 (SDO, ATE) space . . . 338
52.14 Strata-Matching . . . 340

52.14.1 Exact strata-matching . . . 340
Estimates of Treatment Effects . . . 340
Example, estimation of treatment effects . . . 342
52.14.2 Approximate strata-matching . . . 344
52.14.3 Unbiased strata-matching estimators . . . 344
52.15 Propensities . . . 346
52.16 Propensity based estimators of treatment effects . . . 350
52.17 Positivity . . . 351
52.18 Multi-time PO bnets (Panel Data) . . . 352

53 Program evaluation and review technique (PERT) 356
53.1 Example . . . 358

54 Random Forest and Bagging 362
54.1 Bagging (with fully-featured bags) . . . 362
54.2 Bagging (with randomly-shortened bags) . . . 364

55 Recurrent Neural Networks 365
55.1 Language Sequence Modeling . . . 368
55.2 Other types of RNN . . . 368
55.2.1 Long Short Term Memory (LSTM) unit (1997) . . . 370
55.2.2 Gated Recurrence Unit (GRU) (2014) . . . 372

56 Regression Discontinuity Design 374
56.1 PO analysis . . . 374
56.2 Linear Regression . . . 376

57 Regret Based Decision Making: COMING SOON 377

58 Reinforcement Learning (RL) 378
58.1 Exact RL bnet . . . 381
58.2 Actor-Critic RL bnet . . . 383
58.3 Q function learning RL bnet . . . 385

59 Reliability Box Diagrams and Fault Tree Diagrams 387
59.1 Minimal Cut Sets . . . 393

60 Restricted Boltzmann Machines 395

61 ROC curves 397
61.1 Terminology Table Adapted from Wikipedia Ref.[116] . . . 400

62 Scoring the Nodes of a Learned Bnet **402**
62.1 Probability Distributions and Special Functions 403
62.2 Single node with no parents 405
62.3 Multiple nodes with any number of parents 407
62.4 Bayesian Scores 409
62.5 Information Theoretic scores 409

63 Selection Bias Removal: COMING SOON **411**

64 Shapley Explainability **412**
64.0.1 Numerical examples of SHAP 415

65 Simpson's Paradox **418**
65.1 Pearl Causality 420
65.2 Numerical Example 421

66 Structure and Parameter Learning for Bnets **422**
66.1 Overview 422
66.2 Score based SL algorithms 424
66.3 Constraint based SL algorithms 425
66.4 Pseudo-code for some bnet learning algorithms 426

67 Support Vector Machines And Kernel Method **428**
67.1 Learning Algorithm for SVM Classifier 429
67.2 Linear (dot-product) Kernel 430
67.3 Alternatives to Linear Kernel 434
67.4 Random Forest and Kernel Method 434

68 Synthetic Controls **435**
68.1 PO analysis 437

69 Time Series Analysis: ARMA and VAR **439**
69.1 White noise 439
69.2 Backshift operator 440
69.3 Metrics 440
69.4 Definition of $ARMA(p,q)$, $AR(p)$ and $MA(q)$. 442
69.5 Solving $AR(p)$ 444
69.6 Solving $MA(q)$ 445
69.7 Solving $ARMA(p,q)$ 446
69.8 Auto-correlation and partial auto-correlation 446
69.9 Generating function of auto-correlation 450
69.10 Impulse Response 451
69.11 $AR(p)$ and Yule-Walker equations 453
69.12 Forecasting 454

69.13 Model Learning . . . 460
69.14 Differencing and $ARIMA(p, d, q)$ . . . 460
69.15 Parameter Learning . . . 464
69.15.1 PL of $AR(p)$ . . . 464
69.15.2 PL of $MA(q)$ . . . 467
69.15.3 PL of $ARMA(p, q)$ . . . 469
69.16 $VAR(p)$ . . . 470

70 Transfer Learning **471**

71 Transformer Networks **473**

72 Transportability of Causal Knowledge **476**

73 Turbo Codes **480**
73.1 Decoding Algorithm . . . 483
73.2 Message Passing Interpretation of Decoding Algorithm . . . 485

74 Uplift Modelling **486**
74.1 UP types . . . 486
74.2 Some Relevant Technical Formulas from Chapter 52 . . . 487
74.3 UP Analysis . . . 488
74.4 UP Decision Trees . . . 490
74.4.1 Appendix, connection between Δ_c and $\Delta_{c|j}$ . . . 495

75 Variational Bayesian Approximation **496**
75.1 Free Energy $\mathcal{F}(\vec{x})$ . . . 498

76 XGBoost **501**
76.1 Divergences . . . 501
76.2 Minimizing Cost function for single tree . . . 503
76.3 Leaf Splitting . . . 506
76.4 Pruning . . . 506
76.5 Feature Binning . . . 508
76.6 Final estimate of target attribute . . . 508
76.7 Bnet for XGBoost . . . 509

77 Zero Information Transmission (Graphoid Axioms) **511**
77.1 Consequences of Eq.(77.2) . . . 511

Bibliography **514**

Foreword

Welcome to Bayesuvius! a proto-book uploaded to github.

A different Bayesian network is discussed in each chapter. Each chapter title is the name of a Bnet. Chapter titles are in alphabetical order.

This is a volcano in its early stages. First version uploaded to a github repo called Bayesuvius on June 24, 2020. First version only covers 2 Bnets (Linear Regression and GAN). I will add more chapters periodically. Remember, this is a moonlighting effort so I can't do it all at once.

For any questions about notation, please go to Notational Conventions section.

Requests and advice are welcomed.

Thanks for reading this
Robert R. Tucci
www.ar-tiste.xyz

ADDENDA

- **August 15, 2021:** At this point in time, the book has grown to 67 Chapters and 433 pages. Today, I am self-publishing it as an ebook at Amazon and similar outlets. It will still be free.

Navigating the ocean of Judea Pearl's Books

Many of the greatest ideas in the bnet field were invented by Judea Pearl and his collaborators. Thus, this book is heavily indebted to those great scientists. Those ideas have had no clearer and more generous expositor than Judea Pearl himself.

Pearl has written 4 books that I have used in writing Bayesuvius. His 1988 book Ref.[35] dates back to his pre-causal period. That book I used to learn about topics such as d-separation, belief propagation, Markov-blankets, and noisy-ORs. 3 other books that he wrote later, in his causal period, are:

1. In 2000 (1st ed.), and 2013 (2nd ed.), Pearl published what is so far his most technical and exhaustive book on the subject of causality, Ref.[37].

2. In 2016, he released together with Glymour and Jewell, a less advanced "primer" on causality, Ref.[40].

3. In 2018, he released together with Mackenzie his lovely "The Book of Why", Ref.[41].

Those 3 books I used to learn about causality topics such as Do Calculus, backdoor and frontdoor adjustment formulae, linear deterministic bnets with exogenous noise, and counterfactuals.

A micro poem written by me to celebrate Judea Pearl and his work:

I, Robot

Let other robots `talk()`,
while I,
`talk()`, `do()` and `imagine()`.

CI-2-3 track

Figure 3: CI meme

As discussed in Chapter 10, Judea Pearl has proposed 3 rungs of Causal Inference (CI). This book covers all 3 rungs. Some people (for instance, economists)[1] are mostly interested in the highest 2 rungs; i.e, rung 2 (do operations) and rung 3 (imagining/counterfactual thinking). Here is a subset of chapters that I call the CI-2-3 track, that are devoted mostly to rungs 2 and 3.

[1]This is evinced by the excellent books Refs.[6], and [8] written by economists.

1. Backdoor Adjustment Formula
2. Berkson's Paradox
3. Counterfactual Reasoning
4. Difference-in-Differences
5. Do Calculus
6. Do Calculus proofs
7. D-Separation
8. D-Separation and Do Calculus Rules for LDEN: COMING SOON
9. Frisch-Waugh-Lovell (FWL) theorem
10. Frontdoor Adjustment Formula
11. Goodness of Causal Fit
12. Granger Causality
13. Instrumental Inequality and beyond
14. Instrumental Variables
15. Mediation Analysis
16. Meta-learners for estimating ATE
17. Potential Outcomes
18. Regression Discontinuity Design
19. Regret Based Decision Making: COMING SOON
20. Simpson's Paradox
21. Synthetic Controls
22. Transportability of Causal Knowledge
23. Uplift Modelling

Notational Conventions and Preliminaries

0.1 Some abbreviations frequently used throughout this book

- bnet= Bnet= Bayesian Network
- CPT = Conditional Probabilities Table, same as TPM
- DAG = Directed Acyclic Graph
- i.i.d.= independent identically distributed.
- RCT= Randomized Controlled Trial, aka A/B testing.
- TPM= Transition Probability Matrix, same as CPT

0.2 $\mathcal{N}(!a)$

$\mathcal{N}(!a)$ will denote a normalization constant that does not depend on a. For example, $P(x) = \mathcal{N}(!x)e^{-x}$ where $\int_0^\infty dx\ P(x) = 1$.

0.3 One hot

A **one hot** vector of zeros and ones is a vector with all entries zero with the exception of a single entry which is one. A **one cold** vector has all entries equal to one with the exception of a single entry which is zero. For example, if $x^n = (x_0, x_1, \ldots, x_{n-1})$ and $x_i = \delta(i, 0)$ then x^n is one hot.

0.4 Special sets

Define $\mathbb{Z}, \mathbb{R}, \mathbb{C}$ to be the integers, real numbers and complex numbers, respectively.

For $a < b$, define $\mathbb{Z}_I$ to be the integers in the interval I, where $I = [a, b], [a, b), (a, b], (a, b)$ (i.e, I can be closed or open on either side).

$A_{>0} = \{k \in A : k > 0\}$ for $A = \mathbb{Z}, \mathbb{R}$.

0.5 Kronecker delta function

For x, y in discrete set S,

$$\delta(x, y) = \begin{cases} 1 \text{ if } x = y \\ 0 \text{ if } x \neq y \end{cases} \tag{1}$$

0.6 Dirac delta function

For $x, y \in \mathbb{R}$,

$$\int_{-\infty}^{+\infty} dx\ \delta(x - y) f(x) = f(y) \tag{2}$$

0.7 Indicator function (aka Truth function)

$$\mathbb{1}(\mathcal{S}) = \begin{cases} 1 \text{ if } \mathcal{S} \text{ is true} \\ 0 \text{ if } \mathcal{S} \text{ is false} \end{cases} \tag{3}$$

For example, $\delta(x, y) = \mathbb{1}(x = y)$.

0.8 Majority function

The **majority function** is defined as follows.

$$\texttt{majority}(L) = \begin{array}{c} \text{most common element of list } L \\ \text{(ties resolved by chance)} \end{array} \tag{4}$$

Note that the majority function acts on lists, not sets. By definition, all elements of a set appear only once in the set. $\texttt{majority}(L)$ is usually used when the elements of L are categorical (i.e., not real numbers). When they are real numbers, it makes more sense to use, instead of $\texttt{majority}(L)$, a simple average of the elements of L.

0.9 Underlined letters indicate random variables

Random variables will be indicated by underlined letters and their values by non-underlined letters. Each node of a bnet will be labelled by a random variable. Thus, $\underline{x} = x$ means that node $\underline{x}$ is in state x.

It is more conventional to use an upper case letter to indicate a random variable and a lower case letter for its state. Thus, $X = x$ means that random variable X is

in state x. However, we have opted in this book to avoid that notation, because we often want to define certain lower case letters to be random variables or, conversely, define certain upper case letters to be non-random variables.

0.10 Probability distributions

$P_{\underline{x}}(x) = P(\underline{x} = x) = P(x)$ is the probability that random variable $\underline{x}$ equals $x \in S_{\underline{x}}$. $S_{\underline{x}}$ is the set of states (i.e., values) that $\underline{x}$ can assume and $n_{\underline{x}} = |S_{\underline{x}}|$ is the size (aka cardinality) of that set. Hence,

$$\sum_{x \in S_{\underline{x}}} P_{\underline{x}}(x) = 1 \tag{5}$$

$$P_{\underline{x},\underline{y}}(x, y) = P(\underline{x} = x, \underline{y} = y) = P(x, y) \tag{6}$$

$$P_{\underline{x}|\underline{y}}(x|y) = P(\underline{x} = x|\underline{y} = y) = P(x|y) = \frac{P(x, y)}{P(y)} \tag{7}$$

0.11 Discretization of continuous probability distributions

The TPM of a node of a bnet can be either a discrete or a continuous probability distribution. To go from continuous to discrete, one replaces integrals over states of a node by sums over new states, and Dirac delta functions by Kronecker delta functions. More precisely, consider a function $f : [a, b] \to \mathbb{R}$. Express $[a, b]$ as a union of small, disjoint (except for one point) closed sub-intervals (bins) of length Δx. Name one point in each bin to be the representative of that bin, and let $S_{\underline{x}}$ be the set of all the bin representatives. This is called discretization or binning. Then

$$\frac{1}{(b-a)} \int_{[a,b]} dx\ f(x) \to \frac{\Delta x}{(b-a)} \sum_{x \in S_{\underline{x}}} f(x) = \frac{1}{n_{\underline{x}}} \sum_{x \in S_{\underline{x}}} f(x)\ . \tag{8}$$

Both sides of last equation are 1 when $f(x) = 1$. Furthermore, if $y \in S_{\underline{x}}$, then

$$\int_{[a,b]} dx\ \delta(x - y) f(x) = f(y) \to \sum_{x \in S_{\underline{x}}} \delta(x, y) f(x) = f(y)\ . \tag{9}$$

0.12 Samples, i.i.d. variables

$$\vec{x} = (x[0], x[1], x[2] \ldots, x[nsam(\vec{x}) - 1]) = x[:] \tag{10}$$

$nsam(\vec{x})$ is the number of samples of $\vec{x}$. $\underline{x}[\sigma] \in S_{\underline{x}}$ are i.i.d. (independent identically distributed) samples with

$$x[\sigma] \sim P_{\underline{x}} \text{ (i.e. } P_{\underline{x[\sigma]}} = P_{\underline{x}}) \tag{11}$$

$$P(\underline{x} = x) = \frac{1}{nsam(\vec{x})} \sum_{\sigma} \mathbb{1}(x[\sigma] = x) \tag{12}$$

Hence, for any $f : S_{\underline{x}} \to \mathbb{R}$,

$$\sum_{x} P(\underline{x} = x) f(x) = \frac{1}{nsam(\vec{x})} \sum_{\sigma} f(x[\sigma]) \tag{13}$$

If we use two sampled variables, say $\vec{x}$ and $\vec{y}$, in a given bnet, their number of samples $nsam(\vec{x})$ and $nsam(\vec{y})$ need not be equal.

$$P(\vec{x}) = \prod_{\sigma} P(x[\sigma]) \tag{14}$$

$$\sum_{\vec{x}} = \prod_{\sigma} \sum_{x[\sigma]} \tag{15}$$

$$\partial_{\vec{x}} = [\partial_{x[0]}, \partial_{x[1]}, \partial_{x[2]}, \ldots, \partial_{x[nsam(\vec{x})-1]}] \tag{16}$$

$$\begin{aligned} P(\vec{x}) &\approx [\prod_{x} P(x)^{P(x)}]^{nsam(\vec{x})} && (17) \\ &= e^{nsam(\vec{x}) \sum_x P(x) \ln P(x)} && (18) \\ &= e^{-nsam(\vec{x}) H(P_{\underline{x}})} && (19) \end{aligned}$$

0.13 Expected Value and Variance

Given a random variable $\underline{x}$ with states $S_{\underline{x}}$ and a function $f : S_{\underline{x}} \to \mathbb{R}$, define

$$E_{\underline{x}}[f(\underline{x})] = E_{x \sim P(x)}[f(x)] = \sum_{x} P(x) f(x) \tag{20}$$

$$\begin{aligned} Var_{\underline{x}}[f(\underline{x})] &= E_{\underline{x}} \left[(f(\underline{x}) - E_{\underline{x}}[f(\underline{x})])^2 \right] && (21) \\ &= E_{\underline{x}}[f(\underline{x})^2] - (E_{\underline{x}}[f(\underline{x})])^2 && (22) \end{aligned}$$

$$E[\underline{x}] = E_{\underline{x}}[\underline{x}] \tag{23}$$

$$Var[\underline{x}] = Var_{\underline{x}}[\underline{x}] \tag{24}$$

0.14 Conditional Expected Value

Given a random variable $\underline{x}$ with states $S_{\underline{x}}$, a random variable $\underline{y}$ with states $S_{\underline{y}}$, and a function $f : S_{\underline{x}} \times S_{\underline{y}} \to \mathbb{R}$, define

$$E_{\underline{x}|\underline{y}}[f(\underline{x}, \underline{y})] = \sum_x P(x|\underline{y}) f(x, \underline{y}) \ , \tag{25}$$

$$E_{\underline{x}|\underline{y}=y}[f(\underline{x}, y)] = E_{\underline{x}|y}[f(\underline{x}, y)] = \sum_x P(x|y) f(x, y) \ . \tag{26}$$

Note that

$$\begin{aligned} E_{\underline{y}}[E_{\underline{x}|\underline{y}}[f(\underline{x}, \underline{y})]] &= \sum_{x,y} P(x|y) P(y) f(x, y) && (27) \\ &= \sum_{x,y} P(x, y) f(x, y) && (28) \\ &= E_{\underline{x},\underline{y}}[f(\underline{x}, \underline{y})] \ . && (29) \end{aligned}$$

0.15 Notation for covariances

Consider two random variables $\underline{x}, \underline{y}$.

- Mean value of $\underline{x}$

$$\langle \underline{x} \rangle = E_{\underline{x}}[\underline{x}] \tag{30}$$

- Signed distance of $\underline{x}$ to its mean value

$$\Delta \underline{x} = \underline{x} - \langle \underline{x} \rangle \tag{31}$$

- Covariance of $(\underline{x}, \underline{y})$

$$Cov(\underline{x}, \underline{y}) = \left\langle \underline{x}, \underline{y} \right\rangle = \left\langle \Delta \underline{x} \Delta \underline{y} \right\rangle = \left\langle \underline{x}\underline{y} \right\rangle - \langle \underline{x} \rangle \left\langle \underline{y} \right\rangle \tag{32}$$

  $\left\langle \underline{x}, \underline{y} \right\rangle$ is symmetric (i.e., $\left\langle \underline{x}, \underline{y} \right\rangle = \left\langle \underline{y}, \underline{x} \right\rangle$) and bilinear (i.e., $\left\langle \sum_i \alpha_i \underline{x}_i, \underline{y} \right\rangle = \sum_i \alpha_i \left\langle \underline{x}_i, \underline{y} \right\rangle$, where $\alpha_i \in \mathbb{R}$ are non-random scalars and $\underline{x}_i, \underline{y} \in \mathbb{R}$ are real-valued random variables.)

- Variance of $\underline{x}$

$$Var(\underline{x}) = \langle \underline{x}, \underline{x} \rangle \tag{33}$$

- Standard deviation or $\underline{x}$

$$\sigma_{\underline{x}} = \sqrt{\langle \underline{x}, \underline{x} \rangle} \tag{34}$$

- Correlation Coefficient of $(\underline{x}, \underline{y})$

$$\rho_{\underline{x},\underline{y}} = \frac{\langle \underline{x}, \underline{y} \rangle}{\sqrt{\langle \underline{x}, \underline{x} \rangle \langle \underline{y}, \underline{y} \rangle}} \tag{35}$$

0.16 Conditional Covariance

Let $\underline{x}, \underline{y}, \underline{a}$ be random variables. The covariance $Cov(\underline{x}, \underline{y}|\underline{a})$ of $\underline{x}$ and $\underline{y}$ given $\underline{a}$, is defined the same way as $Cov(\underline{x}, \underline{y})$, except that all expected values are conditioned on $\underline{a}$.

$$Cov(\underline{x}, \underline{y}|\underline{a}) = \langle \underline{x}, \underline{y} \rangle_{|\underline{a}} = \left\langle (\underline{x} - \langle \underline{x} \rangle_{|\underline{a}})(\underline{y} - \langle \underline{y} \rangle_{|\underline{a}}) \right\rangle_{|\underline{a}} \tag{36}$$

where

$$\langle \underline{x} \rangle_{|\underline{a}} = E_{\underline{x}|\underline{a}}[\underline{x}] \ . \tag{37}$$

0.17 Law of Total Variance

Claim 1 *Suppose $P : S_{\underline{x}} \times S_{\underline{y}} \to [0, 1]$ is a probability distribution. Suppose $f : S_{\underline{x}} \times S_{\underline{y}} \to \mathbb{R}$ and $f = f(x, y)$. Then*

$$Var_{\underline{x},\underline{y}}(f) = E_{\underline{y}}[Var_{\underline{x}|\underline{y}}(f)] + Var_{\underline{y}}(E_{\underline{x}|\underline{y}}[f]) \ . \tag{38}$$

In particular,

$$Var_{\underline{x}}(x) = E_{\underline{y}}[Var_{\underline{x}|\underline{y}}(x)] + Var_{\underline{y}}(E_{\underline{x}|\underline{y}}[x]) \ . \tag{39}$$

proof:

Let

$$A = \sum_y P(y) \left(\sum_x P(x|y) f \right)^2 \ . \tag{40}$$

Then

$$\begin{aligned} Var_{\underline{x},\underline{y}}(f) &= \sum_{x,y} P(x,y) f^2 - \left(\sum_{x,y} P(x,y) f \right)^2 && (41) \\ &= \left\{ \begin{array}{l} \sum_{x,y} P(x,y) f^2 - A \\ + \left(A - \left(\sum_{x,y} P(x,y) f \right)^2 \right) \end{array} \right. && (42) \end{aligned}$$

$$E_{\underline{y}}[Var_{\underline{x}|\underline{y}}(f)] = \sum_y P(y)\left(\sum_x P(x|y)f^2 - \left(\sum_x P(x|y)f\right)^2\right) \tag{43}$$

$$= \sum_{x,y} P(x,y)f^2 - A \tag{44}$$

$$Var_{\underline{y}}(E_{\underline{x}|\underline{y}}[f]) = \sum_y P(y)\left(\sum_x P(x|y)f\right)^2 - \left(\sum_y P(y)\sum_x P(x|y)f\right)^2 \tag{45}$$

$$= A - \left(\sum_{x,y} P(x,y)f\right)^2 \tag{46}$$

QED

0.18 Normal Distribution

For $x, \mu, \sigma \in \mathbb{R}$, $\sigma > 0$, we define the Normal Distribution (see Fig.4) by

$$\mathcal{N}(x;\mu,\sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} . \tag{47}$$

For a **standard deviation** σ, the **precision** τ is defined as $\tau = \frac{1}{\sigma^2}$.

Claim 2 *If*

$$\underline{x}_1 \sim \mathcal{N}(\mu_1, \sigma_1^2) \tag{48}$$

and

$$\underline{x}_2 \sim \mathcal{N}(\mu_2, \sigma_2^2) \tag{49}$$

then

$$\underline{x} = \underline{x}_1 + \underline{x}_2 \sim \mathcal{N}(\mu_1 + \mu_2, \sigma_1^2 + \sigma_2^2) . \tag{50}$$

proof:

$$P(\underline{x} = x) = \mathcal{N}(!x)\int_{-\infty}^{+\infty} dx_2\ P(\underline{x}_1 + \underline{x}_2 = x|\underline{x}_2 = x_2)P(x_2) \tag{51}$$

$$= \mathcal{N}(!x)\int_{-\infty}^{+\infty} dx_2\ \mathcal{N}(x - x_2;\mu_1,\sigma_1^2)\mathcal{N}(x_2;\mu_2,\sigma_2^2) \tag{52}$$

$$= \mathcal{N}(x;\mu_1 + \mu_2;\sigma_1^2 + \sigma_2^2) \tag{53}$$

QED

Figure 4: Normal Distribution $\mathcal{N}(x; \mu, \sigma^2)$.

0.19 Uniform Distribution

For $a < b$, $x \in [a, b]$

$$\mathcal{U}(x; a, b) = \frac{1}{b - a} \tag{54}$$

0.20 Softmax function (aka Boltzmann Distribution)

The Softmax function is defined by

$$P(x_i|x.) = \frac{e^{x_i}}{\sum_i e^{x_i}} = \text{softmax}(x.)_i \tag{55}$$

The Boltzmann distribution is defined as

$$P(\underline{E}_a = E_a) = \frac{\exp(-\frac{E_a}{kT})}{\sum_a \exp(-\frac{E_a}{kT})} = P(\frac{-E_a}{kT}|E.) \tag{56}$$

for a system with energies E_a and temperature T, where k is Boltzmann's constant.

The function softmax() is called softmax because if we approximate the exponentials, both in the numerator and denominator of Eq.(55), by the largest one of them or zero, we get

$$\text{softmax}(x.)_i \approx \mathbb{1}(i = \operatorname*{argmax}_k x_k) \ . \tag{57}$$

Thus, softmax($x.$) returns a continuous function that approximates a one-hot vector that is 1 at the ith component, where $i = \underset{k}{\text{argmax}}(x_k)$, and zero at the other components.

Note that

$$\frac{\partial \ln P(x_i|x)}{\partial x_a} = \frac{\partial}{\partial x_a} \ln \left[\frac{e^{x_i}}{\sum_i e^{x_i}} \right] = \delta(a,i) - P(x_a|x) \tag{58}$$

For 2 variables x_0, x_1,

$$\begin{aligned} P(x_0|x.) &= \frac{e^{x_0}}{e^{x_0} + e^{x_1}} & (59) \\ &= \text{smoid}(x_0 - x_1) \ , & (60) \end{aligned}$$

$$P(x_1|x.) = \text{smoid}(x_1 - x_0) \ . \tag{61}$$

0.21 Sigmoid and log-odds functions

The **sigmoid (aka exp-it, logistic) function** smoid:$\mathbb{R} \to [0,1]$ is defined by

$$\text{smoid}(x) = \frac{1}{1 + e^{-x}} \tag{62}$$

smoid() is monotonically increasing with smoid$(-\infty) = 0$, smoid$(0) = 1/2$ and smoid$(+\infty) = 1$.

$$\begin{aligned} \text{smoid}(x) + \text{smoid}(-x) &= \frac{1}{1 + e^{-x}} + \frac{1}{1 + e^{x}} & (63) \\ &= \frac{2 + e^{x} + e^{-x}}{2 + e^{x} + e^{-x}} & (64) \\ &= 1 & (65) \end{aligned}$$

The **log-odds (aka log-it) function** lodds:$[0,1] \to \mathbb{R}$ is defined by

$$\text{lodds}(p) = \ln \frac{p}{1-p} \tag{66}$$

lodds() is the inverse of smoid() and vice-versa. For $p \in [0,1]$ and $x \in \mathbb{R}$,

$$\text{lodds}[\text{smoid}(x)] = x \tag{67}$$

$$\text{smoid}[\text{lodds}(p)] = p \tag{68}$$

Claim 3

$$\text{smoid}'(x) = \text{smoid}(x)[1 - \text{smoid}(x)] \tag{69}$$

$$\text{smoid}''(x) = \text{smoid}'(x)[1 - 2\text{smoid}(x)] \tag{70}$$

proof:

In this proof, we will abbreviate smoid(x) by $s(x)$.

$$1 - s(x) = 1 - \frac{1}{1 + e^{-x}} = \frac{e^{-x}}{1 + e^{-x}} \tag{71}$$

$$s'(x) = \frac{e^{-x}}{(1 + e^{-x})^2} = s(x)[1 - s(x)] \tag{72}$$

$$\begin{aligned} s''(x) &= s'(x)[1 - s(x)] + s(x)(-1)s'(x) && (73) \\ &= s'(x)[1 - 2s(x)] && (74) \\ &= s(x)[1 - s(x)][1 - 2s(x)] && (75) \end{aligned}$$

QED

0.22 Estimand, Estimator, Bias

We say $\underline{\hat{\theta}}$ is an **estimator (aka estimate)** of a quantity θ if $E[\underline{\hat{\theta}}] = \theta + b$ for some **bias** b. We say this estimator is an **unbiased estimator** if $b = 0$. θ is sometimes called **the estimand** of $\underline{\hat{\theta}}$.

0.23 Maximum Likelihood Estimate, Likelihood Ratio Test

Given a bnet, let $P(x|\theta)$ be its full joint probability distribution, where x denotes the joint state of all the nodes and θ denotes all the parameters. $P(x|\theta)$ is often called the **likelihood function of** θ and is denoted by

$$L(\theta) = P(x|\theta) \tag{76}$$

It's called a likehood of θ because, even though it's a probability, it isn't the probability of θ, but rather of x.

The value of θ that we obtain by maximizing $L(\theta)$ over θ is called the **maximum likelihood estimate (MLE) of** θ. Let us denote it by $\hat{\theta}$. Note that[2]

$$\sup_{\theta \in S} L(\theta) = L(\hat{\theta}) \tag{77}$$

Let S_0, S_1 be disjoint sets such that $S = S_0 \cup S_1$. We'll say the **null hypothesis** H_0 holds if $\theta \in S_0$, and the **alternative hypothesis** H_1 holds if $\theta \in S_1$. The **likelihood ratio (LR) test statistic** is defined by

$$R = -2 \ln \left(\frac{\sup_{\theta \in S_0} L(\theta)}{\sup_{\theta \in S} L(\theta)} \right) \tag{78}$$

$R \geq 0$ and $R = 0$ if $S_0 = S$. For some small $c > 0$, if $R < c$, then we reject the alternative hypothesis, and if $R > c$, we accept it.

If $S_0 = \{\theta_0\}$, then

$$R = -2[\ln L(\theta_0) - \ln L(\hat{\theta})] \tag{79}$$

0.24 Mean Square Error (MSE)

Suppose we are given $nsam$ samples $y^\sigma \in \mathbb{R}$ labeled by an index σ, and an **estimator** function $\hat{y}^\sigma(a) \in \mathbb{R}$ that depends on a parameter $a \in \mathbb{R}$. Define the **Mean Square Error (MSE)** by

$$MSE(a) = \frac{1}{nsam} \sum_\sigma (y^\sigma - \hat{y}^\sigma(a))^2 \ . \tag{80}$$

For example, in Linear Regression (LR), we have $\hat{y}^\sigma = a_0 + a_1 x^\sigma$ where $a = (a_0, a_1)$ is a deterministic parameter. If the samples y^σ are i.i.d, then we can also write

$$MSE(a) = E_{|a}[(\underline{y} - \underline{\hat{y}}(a))^2] \ . \tag{81}$$

and for LR, $\underline{\hat{y}}(a) = a_0 + a_1 \underline{x}$.

Define the **residual** $\Delta \underline{y}$ by:

$$\Delta \underline{y}(a) = \underline{y} - \underline{\hat{y}}(a) \quad \text{(Hence } \underline{y} = \underline{\hat{y}} + \Delta \underline{y}) \tag{82}$$

In the rest of this section, we will discuss the case that $\hat{y}^\sigma(a)$ is independent of x^σ. I call this the **deterministic MSE (D-MSE)** model. Note that this is different from the LR case where $\hat{y}^\sigma(a)$ does depend on x^σ. In LR, we are trying to fit a line to a cigar-shaped 2-D scatter plot. Here, we are just trying to estimate the mean value (center of mass) of a scatter plot.

[2] "sup" stands for supremum. It's a generalization of the function max() to arbitrary sets that might not be discrete or finite. If S is a finite set, then $\sup_{\theta \in S} f(\theta) = \max_{\theta \in S} f(\theta)$ for any function $f : S \to \mathbb{R}$. Likewise, "inf" stands for infimum, and it generalizes the min() function.

Claim 4 *MSE is minimized over all functions $\hat{y}$ if*

$$\hat{y} = E_{|a}[\underline{y}] \tag{83}$$

proof:

$$MSE = E_{|a}[\underline{y}^2] - 2\hat{y}E_{|a}[\underline{y}] + \hat{y}^2 \tag{84}$$

$$0 = \frac{d}{d\hat{y}}MSE = 2(-E_{|a}[\underline{y}] + \hat{y}) \tag{85}$$

Hence,

$$\hat{y} = E_{|a}[\underline{y}] \tag{86}$$

QED

Sometimes, we will use the notation

$$\hat{y}_{MSE} = E_{|a=a_{MSE}}[\underline{y}] \ . \tag{87}$$

Claim 5 *Suppose $f(a)$ is a function of a. If $\hat{y} = E_{|a}[\underline{y}]$, then*

$$E_{|a}[\Delta\underline{y}] = E[\Delta\underline{y}] = 0 \tag{88}$$

$$E_{|a}\left[\Delta\underline{y}f(\underline{a})\right] = E\left[\Delta\underline{y}f(\underline{a})\right] = 0 \tag{89}$$

proof:

$$E_{|a}[\Delta\underline{y}] = E_{|a}\left[\underline{y} - E_{|a}[\underline{y}]\right] = E_{|a}[\underline{y}] - E_{|a}[\underline{y}] = 0 \tag{90}$$

$$E[\Delta\underline{y}] = E_{\underline{a}}[E_{|\underline{a}}[\Delta\underline{y}]] = 0 \tag{91}$$

$$E_{|\underline{a}}\left[\Delta\underline{y}f(\underline{a})\right] \quad = \quad f(\underline{a})\underbrace{E_{|\underline{a}}[\Delta\underline{y}]}_{=0} \tag{92}$$

$$E[\Delta\underline{y}f(\underline{a})] = E_{\underline{a}}[E_{|\underline{a}}[\Delta\underline{y}f(\underline{a})]] = 0 \tag{93}$$

QED

Claim 6 *If $\hat{y} = E_{|a}[\underline{y}]$, then*

$$\left\langle \Delta\underline{y}, \hat{y} \right\rangle_{|a} = 0 \tag{94}$$

$$Var_{|a}[\underline{y}] = Var_{|a}[\hat{y}] + Var_{|a}[\Delta\underline{y}] \tag{95}$$

The same results hold without the conditioning on a.

proof:

$$\left\langle \Delta\underline{y}, \underline{\hat{y}} \right\rangle_{|a} = \underbrace{E_{|a}[\Delta\underline{y} \underbrace{\underline{\hat{y}}}_{f(\underline{a})}]}_{=0} - \underbrace{E_{|a}[\Delta\underline{y}]}_{=0} E_{|a}[\underline{\hat{y}}] \tag{96}$$

$$\begin{aligned} Var_{|a}[\underline{y}] &= \left\langle \hat{y} + \Delta\underline{y}, \hat{y} + \Delta\underline{y} \right\rangle_{|a} && (97) \\ &= \langle \hat{y}, \hat{y} \rangle_{|a} + \left\langle \Delta\underline{y}, \Delta\underline{y} \right\rangle_{|a} \text{ (by Eq.(94))} && (98) \\ &= Var_{|a}[\hat{y}] + Var_{|a}[\Delta\underline{y}] && (99) \end{aligned}$$

The same proof holds if we remove all the $|a$ subscripts.
QED

Fig.5 illustrates how $\underline{y} = \underline{\hat{y}} + \Delta\underline{y}$ and the variances of these quantities add.

Figure 5: $\underline{y} = \hat{y} + \Delta y$ and the variances (not standard deviations) of these quantities add.

0.25 Bayes Rule, Bayesian Updating And Conjugate Priors

Suppose

$$P(x,\theta) = P(x|\theta)P(\theta) \ . \tag{100}$$

Bayes Rule says

$$\underbrace{P(\theta|x)}_{\text{posterior}} = \mathcal{N}(!\theta)\ \underbrace{P(x|\theta)}_{\text{likelihood}}\ \underbrace{P(\theta)}_{\text{prior}} \ . \tag{101}$$

If, for a given likelihood, the prior and posterior distributions belong to the same family (for instance, they are both Beta distributions), then we say that the prior is the **conjugate prior** of that likelihood.

For example, Beta $\sim$ Bernoulli*Beta. Hence, the Beta distribution[3] is the conjugate prior of the Bernoulli distribution[4]. More explicitly, if

$$p_1 \sim \text{Beta}(p_1;\alpha,\beta) \tag{102}$$

and

$$x|p_1 \sim \text{Bernoulli}(x;p_1) \ , \tag{103}$$

where $p_1 = P(x=1)$, then

$$p_1|x \sim \text{Beta}(p_1;\alpha',\beta') \tag{104}$$

where

$$\alpha' = \alpha + x \tag{105}$$

$$\beta' = \beta + (1-x) \tag{106}$$

Ref.[76] has a table of conjugate priors.

Conjugate priors facilitate Bayesian updating of the prior to posterior in a feedback loop(see Fig.6).

0.26 Linear regression, Ordinary Least Squares (OLS)

Wikipedia articles

1. Linear Regression (LR)

[3]See Ref.[67] for a discussion of the Beta distribution.
[4]See Ref.[66] for a discussion of the Bernoulli distribution

Figure 6: Bayesian updating facilitated by conjugate prior. In this figure, $x_{\leq t} = (x_0, x_1, \ldots, x_{t-1}, x_t)$.

- linear regression, Ref.[101]
- simple linear regression, Ref.[119]
- errors in variable, Ref.[80]

2. Least squares (LS)

- least squares, Ref.[99]
- ordinary least squares (OLS), Ref.[113]

Some nomenclature: In LR, the data consists of **independent x-variables** $x_1^\sigma, x_2^\sigma, \ldots x_n^\sigma$ and a **dependent y-variable** y^σ. We find a linear fit $\hat{y}^\sigma = \beta_0 + \sum_{i=1}^n \beta_i x_i^\sigma$ to the data. $\hat{y}^\sigma$ is called the **estimate** of y^σ. The coefficients β_0, β_i are called **regression coefficients**. $y^\sigma - \hat{y}^\sigma = \epsilon^\sigma$ are called the **residuals**. $\mathcal{E} = \sum_\sigma (\epsilon^\sigma)^2$ is called the **error or cost**. We choose the regression coefficients so as to minimize the error.

Below, we consider two types of LR:

1. LR in which the independent x-variables are non-random.

2. LR in which the independent x-variables are random and i.i.d.

The term OLS is often used to refer to LR of type 1.

For LR of type 2, there is randomness in y coming from the randomness in x and in the residuals. For LR of type 1, there is randomness in y too, but it comes from the residuals only.

Once one assumes that certain variables are random, a "model" (i.e., a bnet, with probabilities expressed as TPMs) must be specified.

0.26.1 LR, assuming x_σ are non-random

Let

$\sigma \in \{0, 1, 2, \ldots, nsam - 1\}$: sample index
$i_0 \in \{0, 1, 2, \ldots, n\}$: index that can assume values 0 to n
$i \in \{1, 2, \ldots, n\}$: index that can assume values 1 to n. i is never equal to 0.
$y_\sigma \in \mathbb{R}$: dependent y-variables
$x_{\sigma i} \in \mathbb{R}$: independent x-variables
$\epsilon_\sigma \in \mathbb{R}$: residuals
$\beta_0, \beta_i \in \mathbb{R}$: regression coefficients

$$y_\sigma = \beta_0 + \sum_{i=1}^{n} x_{\sigma i}\beta_i + \epsilon_\sigma \tag{107}$$

If we define

$$x_{\sigma 0} = 1 \tag{108}$$

for all σ, then

$$y_\sigma = \sum_{i_0=0}^{n} x_{\sigma i_0}\beta_{i_0} + \epsilon_\sigma \ . \tag{109}$$

If y and ϵ are $nsam$ dimensional column vectors and β is an $n+1$ dimensional column vector, and X is an $nsam \times (n+1)$ matrix, then we can write the previous equation in matrix form as:

$$y = X\beta + \epsilon \ . \tag{110}$$

Derivation of LR From Minimization of Error

Let $W = [W_{\sigma,\sigma'}]$ be a symmetric matrix with non-negative diagonal elements $W_{\sigma,\sigma} \geq 0$ for all σ. W is called the **weight matrix**. The following claim describes the method of **Weighted LR** when $W \neq 1$ and of simple LR when $W = 1$.

Claim 7 *Assume the Einstein summation convention; i.e., implicit sum over repeated indices. The error function $\mathcal{E}$ given by*

$$\mathcal{E} = \underbrace{(y_\sigma - X_{\sigma,j_0}\beta_{j_0})}_{residual\ \epsilon_\sigma} W_{\sigma,\sigma'} \underbrace{(y_{\sigma'} - X_{\sigma',k_0}\beta_{k_0})}_{\epsilon_{\sigma'}} \ , \tag{111}$$

is minimized over β_{k_0} for all $k_0 \in \{0, 1, \ldots, n\}$, if β_{k_0} is given by:

$$\hat{\beta} = (X^T W X)^{-1} X^T W y \ . \tag{112}$$

When $W = 1$,

$$\hat{\beta} = (X^T X)^{-1} X^T y \ . \tag{113}$$

proof:

At the minimum of $\mathcal{E}$, the variation $\delta\mathcal{E}$ must vanish:

$$0 = \delta\mathcal{E} = -2X_{\sigma j_0}(\delta\beta_{j_0})W_{\sigma,\sigma'}(y_{\sigma'} - X_{\sigma' k_0}\beta_{k_0}) \ . \tag{114}$$

Thus,

$$X^T W y - X^T W X \beta = 0 \tag{115}$$

which implies Eq.(112).
QED

Geometry of LR with non-random x_σ.

Recall that

$$y = X\beta + \epsilon \ . \tag{116}$$

Define the **projection matrices**

$$\wedge = X(X^T X)^{-1} X^T \ , \ \ \vee = 1 - \wedge \tag{117}$$

A square matrix M is symmetric if $M^T = M$ and is idempotent if $M^2 = M$. $\wedge$ is symmetric and idempotent and so is $\vee$. Note that $\wedge$ and $\vee$ also satisfy:

$$\vee\wedge = \wedge\vee = 0 \tag{118}$$

and

$$\wedge X = X \ , \ \ \vee X = 0 \ . \tag{119}$$

One has

$$\beta = (X^T X)^{-1} X^T (y - \epsilon) \ . \tag{120}$$

Define

$$\boxed{\hat{\beta} = \underbrace{(X^T X)^{-1} X^T}_{B} \ y \ ,} \tag{121a}$$

$$\hat{y} = X\hat{\beta} = \wedge y \ , \tag{121b}$$

and

$$\hat{\epsilon} = y - X\hat{\beta} = y - \hat{y} = (1 - \wedge)y = \vee y \ . \tag{121c}$$

$\wedge$ is sometimes called the **hat matrix**, because it gives y a hat.

Given any function $f = f(y, X, \epsilon)$ and a scalar factor $\xi \in \mathbb{R}$, suppose $f(\xi y, \xi X, \xi\epsilon) = \xi^{\mathcal{O}} f(y, X, \epsilon)$. Then we will say that $f(\cdot)$ is of **order $\mathcal{O}$ under scaling**. Note that $\{\hat{y}, \hat{\epsilon}\}$ are all of order 1 under scaling, $\{\beta, \hat{\beta}, \wedge, \vee\}$ are all of order 0 under scaling, and B is of order -1 under scaling. Thus, each estimator (i.e., symbol with a hat) scales the same way as its estimand (i.e., same symbol without a hat). Furthermore, β, its estimator $\hat{\beta}$, and the projection matrices $\wedge, \vee$ are invariant ($\mathcal{O} = 0$) under scaling.

Note that y can be expressed as a sum of 2 orthogonal estimators:

$$y = \underbrace{\hat{y}}_{\wedge y} + \underbrace{\hat{\epsilon}}_{\vee y} . \tag{122}$$

Fig.7 shows triangles representing $y = X\beta + \epsilon$ and $y = \hat{y} + \hat{\epsilon}$.

Figure 7: Triangles representing $y = X\beta + \epsilon$ and $y = \hat{y} + \hat{\epsilon}$.

LR Goodness of Fit, R^2

Assume the components of ϵ are random with zero mean:

$$E[\underline{\epsilon}] = \langle \underline{\epsilon} \rangle = 0 \tag{123}$$

Assume X and β are not random. This makes $\underline{y} = X\beta + \underline{\epsilon}$ and $\underline{\hat{\beta}} = (X^T X)^{-1} X^T \underline{y}$ random. One finds that

$$\langle \underline{y} \rangle = X\beta \tag{124a}$$

$$\langle \underline{\hat{y}} \rangle = \wedge \underbrace{\langle \underline{y} \rangle}_{X\beta} = \langle \underline{y} \rangle \tag{124b}$$

$$\langle \underline{\hat{\epsilon}} \rangle = \vee \underbrace{\langle \underline{y} \rangle}_{X\beta} = 0 \tag{124c}$$

$$\left\langle \underline{\hat{\beta}} \right\rangle = \beta \tag{124d}$$

So far, we have assumed a zero mean value for ϵ. Next, assume **"homoscedasticity" (homo-spread)**[5], which means that

$$\left\langle \underline{\epsilon}, \underline{\epsilon}^T \right\rangle = \xi^2 I_{nsam} \tag{124e}$$

where $\xi \geq 0$, and I_{nsam} is the $nsam \times nsam$ identity matrix. It follows that

$$\left\langle \underline{y}, \underline{y}^T \right\rangle = \left\langle \underline{\epsilon}, \underline{\epsilon}^T \right\rangle = \xi^2 I_{nsam} \ , \tag{125a}$$

$$\left\langle \underline{\hat{\epsilon}}, \underline{\hat{\epsilon}}^T \right\rangle = \vee \left\langle \underline{y}, \underline{y}^T \right\rangle \vee^T = \xi^2 \vee \ , \tag{125b}$$

$$\left\langle \underline{\hat{y}}, \underline{\hat{y}}^T \right\rangle = \wedge \left\langle \underline{y}, \underline{y}^T \right\rangle \wedge^T = \xi^2 \wedge \tag{125c}$$

and

$$\left\langle \underline{\hat{\beta}}, \underline{\hat{\beta}}^T \right\rangle = B \left\langle \underline{y}, \underline{y}^T \right\rangle B^T = \xi^2 (X^T X)^{-1} \ . \tag{125d}$$

For any random column vector $\underline{a}$, let

$$\| \ \underline{a} \ \|^2 = \underline{a}^T \underline{a} = \mathrm{tr}(\underline{a}\underline{a}^T) \tag{126}$$

and

$$\left\langle \| \ \underline{a} - \langle \underline{a} \rangle \ \|^2 \right\rangle = \left\langle \underline{a}^T \underline{a} \right\rangle - \left\langle \underline{a}^T \right\rangle \langle \underline{a} \rangle = \mathrm{tr} \left\langle \underline{a}, \underline{a}^T \right\rangle \ . \tag{127}$$

Define the following sums of squares (SS):

$$SS_{\underline{y}} = \left\langle \| \ \underline{y} - \left\langle \underline{y} \right\rangle \ \|^2 \right\rangle = \left\langle \underline{y}^T \underline{y} \right\rangle - \left\langle \underline{y}^T \right\rangle \left\langle \underline{y} \right\rangle = \mathrm{tr} \left\langle \underline{y}, \underline{y}^T \right\rangle \tag{128a}$$

$$SS_{\underline{\hat{y}}} = \left\langle \| \ \underline{\hat{y}} - \left\langle \underline{\hat{y}} \right\rangle \ \|^2 \right\rangle = \left\langle \underline{\hat{y}}^T \underline{\hat{y}} \right\rangle - \left\langle \underline{\hat{y}}^T \right\rangle \left\langle \underline{\hat{y}} \right\rangle = \mathrm{tr} \left\langle \underline{\hat{y}}, \underline{\hat{y}}^T \right\rangle \tag{128b}$$

$$SS_{res} = \left\langle \| \ \underline{y} - \underline{\hat{y}} \ \|^2 \right\rangle = \left\langle \| \ \underline{\hat{\epsilon}} \ \|^2 \right\rangle = \mathrm{tr} \left\langle \underline{\hat{\epsilon}}, \underline{\hat{\epsilon}}^T \right\rangle \tag{128c}$$

Claim 8 *The following is true without homo-spread:*

$$\underbrace{\mathrm{tr} \left\langle \underline{y}, \underline{y}^T \right\rangle}_{SS_{\underline{y}}} = \underbrace{\mathrm{tr} \left\langle \underline{\hat{y}}, \underline{\hat{y}}^T \right\rangle}_{SS_{\underline{\hat{y}}}} + \underbrace{\mathrm{tr} \left\langle \underline{\hat{\epsilon}}, \underline{\hat{\epsilon}}^T \right\rangle}_{SS_{res}} \tag{129}$$

This is like the Pythagorean Theorem for the lilac right triangle in Fig.7.

[5] I find the word "homoscedasticity" unnecessarily long, cryptic and easy to misspell so I like to replace it by "homo-spread". The opposite of "homoscedasticity" is "heteroscedasticity", which I like to replace with "hetero-spread".

proof:

From Eqs.125 and 128, we see that

$$SS_{\underline{y}} = \mathrm{tr}\left\langle \underline{y}, \underline{y}^T \right\rangle \tag{130}$$

$$SS_{\underline{\hat{y}}} = \mathrm{tr}\left\langle \underline{\hat{y}}, \underline{\hat{y}}^T \right\rangle = \mathrm{tr}\left\langle \wedge \underline{y}, \underline{y}^T \right\rangle \tag{131}$$

$$SS_{res} = \mathrm{tr}\left\langle \underline{\hat{\epsilon}}, \underline{\hat{\epsilon}}^T \right\rangle = \mathrm{tr}\left\langle \vee \underline{y}, \underline{y}^T \right\rangle \tag{132}$$

Now use $\wedge + \vee = 1$.

QED

The goodness of fit for this model is often measured using the **coefficient of determination** R^2. R^2 is defined by

$$R^2 = 1 - \frac{SS_{res}}{SS_{\underline{y}}} = \frac{SS_{\underline{\hat{y}}}}{SS_{\underline{y}}} = \frac{\mathrm{tr}\left\langle \underline{\hat{y}}, \underline{\hat{y}}^T \right\rangle}{\mathrm{tr}\left\langle \underline{y}, \underline{y}^T \right\rangle} \tag{133}$$

If homo-spread holds, then R^2 reduces to

$$R^2 = \frac{\mathrm{tr}\ \wedge}{nsam}\ . \tag{134}$$

0.26.2 LR, assuming x_σ are random

Let

$i_0 \in \{0, 1, 2, \ldots, n\}$: index that can assume values 0 to n
$i \in \{1, 2, \ldots, n\}$: index that can assume values 1 to n. i is never equal to 0.
$\underline{y} \in \mathbb{R}$: true value of dependent y-variable
$\underline{\hat{y}} \in \mathbb{R}$: estimator of dependent y-variable
$\underline{\epsilon} \in \mathbb{R}$: residual
$\underline{x}_i \in \mathbb{R}$: independent x-variables for $i \in \{1, \ldots, n\}$
$\beta_0, \beta_i \in \mathbb{R}$: regression coefficients

$$\begin{aligned} \underline{\hat{y}} &= \beta_0 + \sum_{j=1}^{n} \beta_j \underline{x}_j && (135) \\ &= \sum_{j_0=0}^{n} \beta_{j_0} \underline{x}_{j_0} \quad (\text{Assume } \underline{x}_0 = 1.) && (136) \end{aligned}$$

$$\underline{y} = \underline{\hat{y}} + \underline{\epsilon} \tag{137}$$

Transforming expressions from non-random to random x_σ

Define the following population averages:

$$E_\sigma[x^\sigma] = \frac{1}{nsam}\sum_\sigma x^\sigma \ , \tag{138}$$

$$E_\sigma[x^\sigma y^\sigma] = \frac{1}{nsam}\sum_\sigma x^\sigma y^\sigma \ , \tag{139}$$

$$\langle x^\sigma, y^\sigma \rangle_\sigma = E_\sigma[x^\sigma y^\sigma] - E_\sigma[x^\sigma]E_\sigma[y^\sigma] \ . \tag{140}$$

Claim 9 *If the x_σ are i.i.d. random variables,*

$$E_\sigma[x^\sigma] = \langle \underline{x} \rangle \tag{141}$$

$$E_\sigma[x^\sigma y^\sigma] = \langle \underline{x}\underline{y} \rangle \tag{142}$$

$$\langle x^\sigma, y^\sigma \rangle_\sigma = \langle \underline{x}, \underline{y} \rangle \tag{143}$$

proof:

$$\begin{aligned} \frac{1}{nsam}\sum_\sigma x^\sigma &= \frac{1}{nsam}\sum_{x\in S_{\underline{x}}} x \underbrace{\sum_\sigma \mathbb{1}(x^\sigma = x)}_{N(x^\sigma = x)} && (144) \\ &= \sum_x x\ P(x) && (145) \\ &= \langle \underline{x} \rangle && (146) \end{aligned}$$

$$\begin{aligned} \frac{1}{nsam}\sum_\sigma x^\sigma y^\sigma &= \frac{1}{nsam}\sum_{x\in S_{\underline{x}}}\sum_{y\in S_{\underline{y}}} xy \underbrace{\sum_\sigma \mathbb{1}(x^\sigma = x, y^\sigma = y)}_{N(x^\sigma = x, y^\sigma = y)} && (147) \\ &= \sum_{x,y} xy\ P(x,y) && (148) \\ &= \langle \underline{x}\underline{y} \rangle && (149) \end{aligned}$$

Eq.(143) follows from Eq.(141) and Eq.(142).
QED

Recall that

$$Y_\sigma = \beta_0 + \sum_{j=1}^{n} X_{\sigma,j}\beta_j + \epsilon_\sigma \ . \tag{150}$$

Assume

$$E_\sigma[X_{\sigma,k}\epsilon_\sigma] = E_\sigma[X_{\sigma,k}]\underbrace{E_\sigma[\epsilon_\sigma]}_{=0} = 0 \ . \tag{151}$$

Then we have

$$E_\sigma[X_{\sigma,k}Y_\sigma] = E_\sigma[X_{\sigma,k}]\beta_0 + \sum_{j=1}^{n} E_\sigma[X_{\sigma,k}X_{\sigma,j}]\beta_j + \underbrace{E_\sigma[X_{\sigma,k}\epsilon_\sigma]}_{=0} \tag{152}$$

and

$$E_{\sigma'}[X_{\sigma',k}]E_\sigma[Y_\sigma] = E_{\sigma'}[X_{\sigma',k}]\beta_0 + \sum_{j=1}^{n} E_{\sigma'}[X_{\sigma',k}]E_\sigma[X_{\sigma,j}]\beta_j + \underbrace{E_{\sigma'}[X_{\sigma',k}]E_\sigma[\epsilon_\sigma]}_{=0} \ . \tag{153}$$

Subtracting Eq.(153) from Eq.(152), we get

$$\langle X_{\sigma,k}, Y_\sigma\rangle_\sigma = \sum_{j=1}^{n} \langle X_{\sigma,k}, X_{\sigma,j}\rangle_\sigma \beta_j \ . \tag{154}$$

Define the n dimensional covariance matrix C by

$$C_{k,j} = \langle X_{\sigma,k}, X_{\sigma,j}\rangle_\sigma \ . \tag{155}$$

Then Eq.(154) implies

$$\beta_j = \sum_{k=1}^{n} C^{-1}_{j,k} \langle X_{\sigma,k}, Y_\sigma\rangle_\sigma \tag{156}$$

for all $j = 1, 2, \ldots, n$.

If we assume that the x_σ are i.i.d., then, by virtue of Claim 9, the matrix C tends to

$$C_{k,j} \to \langle \underline{x}_k, \underline{x}_j\rangle \tag{157}$$

and Eq.(156) implies

$$\beta_j = \sum_{k=1}^{n} C^{-1}_{j,k} \langle \underline{x}_k, \underline{y}\rangle \ . \tag{158}$$

Geometry of LR with random x_σ

Recall that

$$\underline{y} = \underbrace{\beta_0 + \sum_{j=1}^{n} \beta_j \underline{x}_j}_{\hat{\underline{y}}} + \underline{\epsilon} \ . \tag{159}$$

Assume

$$\langle \underline{\epsilon} \rangle = 0 \tag{160}$$

and

$$\left\langle \underline{x}_j, \underline{\epsilon} \right\rangle = 0 \tag{161}$$

for all j.

For $k = 1, \ldots, n$,

$$\left\langle \underline{x}_k, \underline{y} \right\rangle = \sum_{j=1}^{n} \beta_j \left\langle \underline{x}_k, \underline{x}_j \right\rangle \ . \tag{162}$$

Let $\underline{x}^n$ and β^n be n-dimensional column vectors. Then Eq.(162) can be represented in matrix notation by

$$\left\langle \underline{x}^n, \underline{y} \right\rangle = \left\langle \underline{x}^n, (\underline{x}^n)^T \right\rangle \beta^n \ . \tag{163}$$

Hence,

$$\boxed{\beta^n = \left\langle \underline{x}^n, (\underline{x}^n)^T \right\rangle^{-1} \left\langle \underline{x}^n, \underline{y} \right\rangle \ .} \tag{164}$$

For β_0, use

$$\beta_0 = \left\langle \underline{y} \right\rangle - \left\langle \underline{x}^n \right\rangle^T \beta^n \tag{165}$$

Notice that Eq.(164) for the regression coefficients is the same as Eq.(158). So we have rederived the same formula via a different method.

Next, we will write Eq.(164) for the special cases $n = 1$ and $n = 2$, where n is the number of independent x-variables $\underline{x}_j$

1. $n = 1$ ($\underline{y}$ fitted by a line)

$$\underline{y} = \beta_0 + \beta_1 \underline{x} + \underline{\epsilon} \tag{166}$$

Eq.(164) becomes

$$\beta_1 = \frac{\left\langle \underline{y}, \underline{x} \right\rangle}{\left\langle \underline{x}, \underline{x} \right\rangle} \tag{167}$$

2. $n = 2$ ($\underline{y}$ fitted by a plane)

$$\underline{y} = \beta_0 + \beta_1 \underline{x}_1 + \beta_2 \underline{x}_2 + \underline{\epsilon} \tag{168}$$

Define

$$C_{i,j} = \left\langle \underline{x}_i, \underline{x}_j \right\rangle \tag{169}$$

for all i, j. Then Eq.(164) becomes[6]

[6] Recall that if $M = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ then $M^{-1} = \frac{1}{\det M} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$

$$\begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} = C^{-1} \begin{bmatrix} \langle \underline{y}, \underline{x}_1 \rangle \\ \langle \underline{y}, \underline{x}_2 \rangle \end{bmatrix} \tag{170}$$

$$= \frac{1}{\det C} \begin{bmatrix} C_{22} & -C_{12} \\ -C_{21} & C_{11} \end{bmatrix} \begin{bmatrix} \langle \underline{y}, \underline{x}_1 \rangle \\ \langle \underline{y}, \underline{x}_2 \rangle \end{bmatrix} \tag{171}$$

Hence,

$$\beta_1 = \frac{\partial \underline{y}}{\partial \underline{x}_1} = \frac{C_{22} \langle \underline{y}, \underline{x}_1 \rangle - C_{12} \langle \underline{y}, \underline{x}_2 \rangle}{C_{11} C_{22} - C_{12}^2} \tag{172}$$

Eq.(172) agrees with the value of $\beta_{YX,Z}$ in Ref.[31] by Pearl, if we replace in Pearl's formulae $X \to \underline{x}_1$, $Y \to \underline{y}$, $Z \to \underline{x}_2$.

Regression interpreted as differentiation of y

Finding the derivative of y with respect to (wrt) X (aka "**Regressing y on X**") is finding $\hat{\beta} = \frac{d}{dX} \underbrace{y}_{X\beta}$.

Recall that

$$\underline{y} = \beta_0 + \sum_{k=1}^{n} \underline{x}_k \beta_k + \underline{\epsilon} \ . \tag{173}$$

Therefore,

$$\langle \underline{x}_i, \underline{y} \rangle = \sum_{k=1}^{n} \langle \underline{x}_i, \underline{x}_k \rangle \beta_k \tag{174}$$

$$= \langle \underline{x}_i, \underline{x}_i \rangle \beta_i + \sum_{k=1}^{n} \mathbb{1}(k \neq i) \langle \underline{x}_i, \underline{x}_k \rangle \beta_k \ . \tag{175}$$

Hence,

$$\beta_i = \frac{\langle \underline{x}_i, \underline{y} \rangle}{\langle \underline{x}_i, \underline{x}_i \rangle} - \sum_{k=1}^{n} \mathbb{1}(k \neq i) \frac{\langle \underline{x}_i, \underline{x}_k \rangle}{\langle \underline{x}_i, \underline{x}_i \rangle} \beta_k \ . \tag{176}$$

Let's represent the linear operator $\langle \underline{x}_i, \underline{x}_i \rangle^{-1} \langle \underline{x}_i, \cdot \rangle$ as a derivative:

$$\frac{d \cdot}{d \underline{x}_i} = \frac{\langle \underline{x}_i, \cdot \rangle}{\langle \underline{x}_i, \underline{x}_i \rangle} \ . \tag{177}$$

Eq.(176) can be expressed in derivative notation as:

$$\beta_i = \frac{d \underline{y}}{d \underline{x}_i} - \sum_{k=1}^{n} \mathbb{1}(k \neq i) \frac{d \underline{x}_k}{d \underline{x}_i} \beta_k \tag{178}$$

Note that Eq.(178) evokes the formula for differentials:

$$d\underline{y} = \sum_{k=1}^{n} \beta_k d\underline{x}_k \ . \tag{179}$$

Note also that, because of the linearity of the derivative operator, Eq.(178) implies:

$$\beta_i = \frac{d}{d\underline{x}_i} \left(\underbrace{\underline{y} - \sum_{k=1}^{n} \mathbb{1}(k \neq i) \underline{x}_k \beta_k}_{\underline{y} - \underline{x}_i \beta_i} \right) \ . \tag{180}$$

Eq.(180) can be used to find $\hat{\beta}_i$ in two steps:
STEP 1: Regress $\underline{y} - \underline{x}_i \beta_i$ on $(\underline{x}_k)_{k \in \{1,2,\ldots,n\} - \{i\}}$. Get estimates $(\hat{\beta}_k)_{k \in \{1,2,\ldots,n\} - \{i\}}$.
STEP 2: Regress $\underline{y} - \sum_{k \neq i} \underline{x}_k \hat{\beta}_k$ on $\underline{x}_i$. Get estimate $\hat{\beta}_i$.
Of course, one can also find $\hat{\beta}_i$ by regressing $\underline{y}$ on $(\underline{x}_k)_{k \in \{1,2,\ldots,n\}}$, to get estimates $(\hat{\beta}_k)_{k \in \{1,2,\ldots,n\}}$.

0.27 Entropy, Kullback-Liebler divergence, Cross-Entropy

For probability distributions $p(x), q(x)$ of $x \in S_{\underline{x}}$

- Entropy:

$$H(p) = -\sum_x p(x) \ln p(x) \geq 0 \tag{181}$$

- Kullback-Liebler divergence:

$$D_{KL}(p \parallel q) = \sum_x p(x) \ln \frac{p(x)}{q(x)} \geq 0 \tag{182}$$

- Cross entropy:

$$\begin{aligned} CE(p \parallel q) &= -\sum_x p(x) \ln q(x) & (183) \\ &= H(p) + D_{KL}(p \parallel q) & (184) \end{aligned}$$

0.28 Definition of various entropies used in Shannon Information Theory

- **(plain) Entropy of $\underline{x}$**

$$H(\underline{x}) = -\sum_x P(x) \ln P(x) \tag{185}$$

This quantity measures the spread of $P_{\underline{x}}$. $H(\underline{x}) \geq 0$ and it vanishes iff $P(x) = \delta(x, x_0)$ (deterministic case)

- **Conditional Entropy of $\underline{y}$ given $\underline{x}$**

$$\begin{aligned} H(\underline{y}|\underline{x}) &= -\sum_{x,y} P(x,y) \ln P(y|x) & (186) \\ &= H(\underline{y}, \underline{x}) - H(\underline{x}) & (187) \end{aligned}$$

This quantity measures the conditional spread of $\underline{y}$ given $\underline{x}$. $H(\underline{y}|\underline{x}) \geq 0$.

- **Mutual Information (MI) of $\underline{x}$ and $\underline{y}$.**

$$\begin{aligned} H(\underline{y} : \underline{x}) &= \sum_{x,y} P(x,y) \ln \frac{P(x,y)}{P(x)P(y)} & (188) \\ &= H(\underline{x}) + H(\underline{y}) - H(\underline{y}, \underline{x}) & (189) \end{aligned}$$

This quantity measures the correlation between $\underline{x}$ and $\underline{y}$. $H(\underline{y} : \underline{x}) \geq 0$ and it vanishes iff $P(x,y) = P(x)P(y)$.

- **Conditional Mutual Information (CMI)[7] of $\underline{x}$ and $\underline{y}$ given $\underline{\lambda}$**

$$\begin{aligned} H(\underline{y} : \underline{x}|\underline{\lambda}) &= \sum_{x,y,\lambda} P(x,y,\lambda) \ln \frac{P(x,y|\lambda)}{P(x|\lambda)P(y|\lambda)} & (190) \\ &= H(\underline{x}|\underline{\lambda}) + H(\underline{y}|\underline{\lambda}) - H(\underline{y}, \underline{x}|\underline{\lambda}) & (191) \end{aligned}$$

This quantity measures the conditional correlation of $\underline{x}$ and $\underline{y}$ given $\underline{\lambda}$. $H(\underline{y} : \underline{x}|\underline{\lambda}) \geq 0$ and it vanishes iff $P(x,y|\lambda) = P(x|\lambda)P(y|\lambda)$.

[7] CMI can be read as "see me".

An interesting special case occurs when $P(\lambda) = \delta(\lambda, \lambda_0)$ (the frequentist case of no λ prior.) In that case CMI reduces to

$$H(\underline{y} : \underline{x}|\lambda_0) = \sum_{x,y} P(x, y|\lambda_0) \ln \frac{P(x, y|\lambda_0)}{P(x|\lambda_0)P(y|\lambda_0)} \geq 0 \tag{192}$$

- **Kullback-Liebler Divergence from $P_{\underline{x}}$ to $P_{\underline{y}}$.**

  Assume random variables $\underline{x}$ and $\underline{y}$ have the same set of states $S_{\underline{x}} = S_{\underline{y}}$. Then

$$D_{KL}(P_{\underline{x}} \parallel P_{\underline{y}}) = \sum_x P_{\underline{x}}(x) \ln \frac{P_{\underline{x}}(x)}{P_{\underline{y}}(x)} \tag{193}$$

  This quantity measures a non-symmetric distance between the probability distributions $P_{\underline{x}}$ and $P_{\underline{y}}$. $D_{KL}(P_{\underline{x}} \parallel P_{\underline{y}}) \geq 0$ and it equals zero iff $P_{\underline{x}} = P_{\underline{y}}$.

0.29 Pearson Chi-Squared Test

The **Pearson divergence** (aka **Pearson Chi-squared test statistic**) for two probability distributions $PO(x)$ and $PE(x)$, where $x \in S_{\underline{x}}$, is defined as follows:

$$D_{\chi^2} = \sum_x \frac{[PO(x) - PE(x)]^2}{PE(x)} = \sum_x \frac{PO^2(x)}{PE(x)} - 1 \ . \tag{194}$$

Usually PO is the observed probability distribution and PE is the expected, theoretical one.

As the following claim shows, the Pearson divergence is closely related to the Kullback-Liebler divergence.

Claim 10 *If* $\left|\frac{PO(x)}{PE(x)} - 1\right| << 1$ *for all* $x \in S_{\underline{x}}$, *then*

$$D_{KL}(PO \parallel PE) \approx D_{\chi^2} \ . \tag{195}$$

proof:

$$\begin{aligned}
D_{KL}(PO \parallel PE) &= \sum_x PO(x) \ln \frac{PO(x)}{PE(x)} && (196)\\
&= \sum_x PO(x) \ln \left(1 + \frac{PO(x)}{PE(x)} - 1\right) && (197)\\
&\approx \sum_x PO(x) \left(\frac{PO(x)}{PE(x)} - 1\right) && (198)\\
&= \sum_x \frac{PO^2(x)}{PE(x)} - 1 && (199)\\
&= D_{\chi^2} && (200)
\end{aligned}$$

QED

Let $nx = |S_{\underline{x}}|$. Let $P_{\chi^2}(y)$ be the χ^2 (with $nx - 1$ degrees of freedom) probability distribution, and let $F_{\chi^2}(\alpha)$ be its cumulative distribution. Find α such that

$$95\% = \int_0^\alpha dy\ P_{\chi^2}(y) = F_{\chi^2}(\alpha) \tag{201}$$

If $D_{\chi^2} < \alpha$, then we say that $PO = PE$ to 95% significance level (SL), whereas if $D_{\chi^2} > \alpha$, we say that $PO \neq PE$ to 95% SL (i.e., SL= 95%). The higher SL becomes, the higher α becomes, and the bigger the divergence D_{χ^2} has to be, before we are willing to declare that $PO \neq PE$.

0.30 Demystifying Population and Sample Variances

Let $x[\sigma] = x^\sigma$. Given i.i.d.real variables $(x^\sigma)_{\sigma=0,1,\ldots,n-1}$, let[8]

$$\hat{\mu} = \overline{x} = \frac{1}{n}\sum_\sigma x^\sigma \tag{202}$$

$$(\widehat{\sigma^2})_\infty = \frac{1}{n}\sum_\sigma (x^\sigma - \mu)^2 \tag{203}$$

$$\widehat{\sigma^2} = \frac{1}{n-1}\sum_\sigma (x^\sigma - \hat{\mu})^2 \tag{204}$$

Statisticians[9] call $(\widehat{\sigma^2})_\infty$ the "population variance". I will call it the **population variance for fixed** μ. Note that it depends on the fixed parameter μ. Statisticians call $\widehat{\sigma^2}$ the "sample variance". Instead, I will call $\widehat{\sigma^2}$ the **population variance for random** μ.

If one treats x^σ as a random variable, then one must treat $\hat{\mu}$ as a random variable too. Let

$$E[\underline{x}^\sigma] = \mu \tag{205}$$

and

$$\left\langle \underline{x}^\sigma, \underline{x}^{\sigma'} \right\rangle = \delta(\sigma, \sigma')\sigma^2\ . \tag{206}$$

[8] Do not confuse the sample index σ and the standard deviation σ.

[9] In the language of Statisticians, a "population" is supposed to be so large that its μ does not fluctuate, and a "sample" is supposed to be a small subset of that population for which the μ is assumed to fluctuate. In this book, I use the word "population" to mean a set of any size containing individuals, I use the word "sub-population" to refer to a subset of the population, and I use the word "sample" (aka individual, observation, unit, record) to mean a single individual of the population.

Then one can show that

$$E[\underline{(\widehat{\sigma^2})_\infty}] = \frac{1}{n} E\left[\sum_\sigma (\underline{x}^\sigma - \mu)^2\right] \tag{207}$$

$$= \sigma^2 \tag{208}$$

and

$$E[\underline{\widehat{\sigma^2}}] = \frac{1}{n-1} E\left[\sum_\sigma (\underline{x}^\sigma - \underline{\hat{\mu}})^2\right] \tag{209}$$

$$= \sigma^2 \tag{210}$$

This is the reason why we use an $n-1$ instead of an n in $\widehat{\sigma^2}$. Because it makes $E[\widehat{\sigma^2}] = \sigma^2$ so $\widehat{\sigma^2}$ is an unbiased estimator of the single individual variance σ^2.

The intuitive reason for why $\widehat{\sigma^2}$ is divided by $n-1$ instead of n is that whereas μ in $(\widehat{\sigma^2})_\infty$ is kept fixed and is "quiet", the $\underline{\hat{\mu}}$ in $\widehat{\sigma^2}$ is a random variable, noisy instead of quiet. The fluctuations in $\underline{\hat{\mu}}$ are strongly correlated with the fluctuations of the $\underline{x}^\sigma$, so they decrease the fluctuations in $\widehat{\sigma^2}$ compared to those in $(\widehat{\sigma^2})_\infty$. By dividing by $n-1$ instead of n, we compensate for this decrease in fluctuations so that the ratio of the numerator and denominator of $\widehat{\sigma^2}$ equals σ^2, instead of something *smaller* than σ^2, as would happen if were to divide by n instead of $n-1$. In terms of "degrees of freedom"(DOFs), $(\widehat{\sigma^2})_\infty$ has n DOFs (namely one for each $\underline{x}^\sigma$), whereas $\widehat{\sigma^2}$ has $n-1$ DOFs. (the presence of $\underline{\hat{\mu}}$ subtracts one DOF). In both $(\widehat{\sigma^2})_\infty$ and $\widehat{\sigma^2}$, one divides by the number of DOFs.

0.31 Independence of $\hat{\mu}$ and $\hat{\sigma^2}$

Let $x[\sigma] = x^\sigma$. Consider i.i.d.real variables $(x^\sigma)_{\sigma=0,1,\ldots,n-1}$ such that[10]

$$E[\underline{x}^\sigma] = \mu \tag{211}$$

$$\left\langle \underline{x}^\sigma, \underline{x}^{\sigma'} \right\rangle = \delta(\sigma, \sigma')\sigma^2 \ . \tag{212}$$

$$\hat{\mu} = \overline{x} = \frac{1}{n} \sum_\sigma x^\sigma \tag{213}$$

$$(\widehat{\sigma^2})_\infty = \frac{1}{n} \sum_\sigma (x^\sigma - \mu)^2 \tag{214}$$

[10]Do not confuse the sample index σ and the standard deviation σ.

$$\widehat{\sigma^2} = \frac{1}{n-1} \sum_{\sigma} (x^\sigma - \hat{\mu})^2 \tag{215}$$

Claim 11 *Let*

$$\underline{\Delta}^\sigma = \underline{x}^\sigma - \mu \ . \tag{216}$$

For any $\sigma_1, \sigma_2, \sigma_3$,

$$\langle \underline{\Delta}^{\sigma_1} \underline{\Delta}^{\sigma_2}, \underline{\Delta}^{\sigma_3} \rangle = 0 \ . \tag{217}$$

proof:

Suppose $\sigma_2 \neq \sigma_3$. Then

$$\langle \underline{\Delta}^{\sigma_1} \underline{\Delta}^{\sigma_2}, \underline{\Delta}^{\sigma_3} \rangle = \underbrace{\langle \underline{\Delta}^{\sigma_2} \rangle}_{0} \langle \underline{\Delta}^{\sigma_1}, \underline{\Delta}^{\sigma_3} \rangle = 0 \ . \tag{218}$$

So assume $\sigma_2 = \sigma_3 = \sigma$ and evaluate $\langle \underline{\Delta}^{\sigma_1} \underline{\Delta}^{\sigma}, \underline{\Delta}^{\sigma} \rangle$.

Suppose $\sigma_1 \neq \sigma$. Then

$$\langle \underline{\Delta}^{\sigma_1} \underline{\Delta}^{\sigma}, \underline{\Delta}^{\sigma} \rangle = \underbrace{\langle \underline{\Delta}^{\sigma_1} \rangle}_{0} \langle \underline{\Delta}^{\sigma}, \underline{\Delta}^{\sigma} \rangle = 0 \ . \tag{219}$$

So suppose $\sigma_1 = \sigma$ and evaluate $\langle (\underline{\Delta}^\sigma)^2, \underline{\Delta}^\sigma \rangle$.

$$\left\langle (\underline{\Delta}^\sigma)^2, \underline{\Delta}^\sigma \right\rangle = \underbrace{\left\langle (\underline{\Delta}^\sigma)^3 \right\rangle}_{0} - \left\langle (\underline{\Delta}^\sigma)^2 \right\rangle \underbrace{\langle \underline{\Delta}^\sigma \rangle}_{0} = 0 \ . \tag{220}$$

QED

Claim 12

$$\left\langle \underline{\widehat{\sigma^2}}, \underline{\hat{\mu}} \right\rangle = 0 \ . \tag{221}$$

proof:

$$\begin{aligned}
\left\langle \underline{\widehat{\sigma^2}}, \underline{\hat{\mu}} \right\rangle &= \frac{1}{n(n-1)} \sum_{\sigma,\sigma'} \left\langle \left(\underline{x}^\sigma - \frac{1}{n} \sum_{\sigma''} \underline{x}^{\sigma''} \right)^2, \underline{x}^{\sigma'} \right\rangle && (222) \\
&= \frac{1}{n(n-1)} \sum_{\sigma,\sigma'} \left\langle \left(\underline{x}^\sigma - \frac{1}{n} \sum_{\sigma''} \underline{x}^{\sigma''} \right)^2, \underline{\Delta}^{\sigma'} \right\rangle && (223) \\
&= \frac{1}{n(n-1)} \sum_{\sigma,\sigma'} \left\langle \left(\underline{\Delta}^\sigma - \frac{1}{n} \sum_{\sigma''} \underline{\Delta}^{\sigma''} \right)^2, \underline{\Delta}^{\sigma'} \right\rangle && (224) \\
&= 0 \quad \text{by Claim 11} \ . && (225)
\end{aligned}$$

QED

0.32 Chi-square distribution

This section is based on Ref.[73].

$$\underline{q} \longleftarrow \underline{z}.$$

Figure 8: Bnet used to define the Chi-square distribution.

Let $q \in \mathbb{R}$ and $z. = \{z_i\}_{i=0,1,\ldots,\nu-1}$ where $z_i \in \mathbb{R}$. Consider the bnet of Fig.8. The TPMs, printed in blue, for that bnet, are as follows:[11]

$$P(z_i) = \mathcal{N}(z_i; \mu = 0, \sigma^2 = 1) \ . \tag{226}$$

We want

$$\underline{q} = \sum_{i=0}^{\nu-1} (\underline{z}_i)^2 \tag{227}$$

so $P(q|z.)$ is a Dirac delta function:

$$P(q|z.) = \delta(q - \sum_{i=0}^{\nu-1} (z_i)^2) \ . \tag{228}$$

Therefore

$$\begin{aligned} P(q) &= \prod_{i=0}^{\nu-1} \left\{ \int dz_i \ P(z_i) \right\} P(q|z.) && (229) \\ &= \mathcal{N}(!q) q^{\frac{\nu}{2}-1} e^{-q/2} = \chi^2(q;\nu) \ , && (230) \end{aligned}$$

where $\mathcal{N}(!q)$ is a constant that does not depend on q and is adjusted so that $\int_0^\infty dq \ P(q) = 1$.

0.33 Student's t-distribution

This section is based on Ref.[122].

Let $x[\sigma] = x^\sigma$. Consider i.i.d.real variables $(x^\sigma)_{\sigma=0,1,\ldots,n-1}$ such that[12]

$$E[\underline{x}^\sigma] = \mu \tag{231}$$

[11] Don't confuse the q independent constant $\mathcal{N}(!q)$ with the normal probability distribution $\mathcal{N}(x;\mu,\sigma^2)$.

[12] Do not confuse the sample index σ and the standard deviation σ.

$$\left\langle \underline{x}^{\sigma}, \underline{x}^{\sigma'} \right\rangle = \delta(\sigma, \sigma')\sigma^2 \ . \tag{232}$$

$$\hat{\mu} = \overline{x} = \frac{1}{n}\sum_{\sigma} x^{\sigma} \tag{233}$$

$$(\widehat{\sigma^2})_{\infty} = \frac{1}{n}\sum_{\sigma}(x^{\sigma} - \mu)^2 \tag{234}$$

$$\widehat{\sigma^2} = \frac{1}{n-1}\sum_{\sigma}(x^{\sigma} - \hat{\mu})^2 \ . \tag{235}$$

If we define

$$z = \frac{\hat{\mu} - \mu}{\frac{\sigma}{\sqrt{n}}} \ , \tag{236}$$

then $\underline{z}$ has a Standard Normal Distribution (SND):

$$P(z) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} = \mathcal{N}(z; \mu = 0, \sigma^2 = 1) \tag{237}$$

But what if we allow the standard deviation σ to fluctuate in the expression Eq.(236) for z? Define

$$t = \frac{\hat{\mu} - \mu}{\sqrt{\frac{\widehat{\sigma^2}}{n}}} \ . \tag{238}$$

Then one can show that $\underline{t}$ has the **Student's t-distribution** $\mathrm{Stud}(t; \nu = n-1)$ given by:

$$P(t) = \mathcal{N}(!t)(1 + \frac{t^2}{\nu})^{-\frac{\nu+1}{2}} = \mathrm{Stud}(t; \nu = n-1) \tag{239}$$

Note that if we use the approximation $e^x \approx 1 + x + \mathcal{O}(x^2)$, we can show that $\mathrm{Stud}(t)$ tends to the SND when $n >> 1$:

$$\begin{aligned} P(t) &= \mathcal{N}(!t)(1 + \frac{t^2}{\nu})^{-\frac{\nu+1}{2}} && (240) \\ &\approx \mathcal{N}(!t) e^{-\frac{t^2}{2}\frac{\nu+1}{\nu}} && (241) \\ &\approx \mathcal{N}(t; \mu = 0, \sigma^2 = 1) \ . && (242) \end{aligned}$$

Partial derivation of the explicit form of $\mathrm{Stud}(t)$**.**

Note that the z definition Eq.(236) and the t definition Eq.(238), imply that

$$t = z \underbrace{\sqrt{\frac{\sigma^2}{\widehat{\sigma^2}}}}_{\varrho} , \tag{243}$$

In the expression $\underline{t} = \underline{z}\underline{\varrho}$, the random variables $\underline{z}$ and $\underline{\varrho}$ are independent because, as shown in Section 0.31, $\hat{\mu}$ and $\hat{\sigma^2}$ are independent. Therefore, the random variable $\underline{t}$ can be defined using the bnet of Fig.9.

Figure 9: Bnet used to define the Student's t-distribution.

The TPMs, printed in blue, for the bnet of Fig.9, are as follows:

$$P(t|z, \varrho) = \delta(t - z\varrho) \quad \text{(Dirac delta function)} \tag{244}$$

$$P(z) = \mathcal{N}(z; \mu = 0, \sigma^2 = 1) \tag{245}$$

$$P(\varrho) = \text{given by Eq.(258) below.} \tag{246}$$

Note that

$$\begin{aligned} P(\underline{t} = t) &= P(\underline{z}\underline{\varrho} = t) && (247)\\ &= \int d\varrho \; P(\underline{z} = \frac{t}{\varrho}|\varrho) P(\varrho) && (248)\\ &= \int d\varrho \; \mathcal{N}(\frac{t}{\varrho}; 0, 1) P(\varrho) && (249)\\ &= \int d\varrho \; \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{t}{\varrho})^2} P(\varrho) \; . && (250) \end{aligned}$$

If we define q by

$$q = \frac{n-1}{\varrho^2} \; , \tag{251}$$

then

$$q = \frac{(n-1)\widehat{\sigma^2}}{\sigma^2} = \frac{1}{\sigma^2}\sum_{\sigma=0}^{n-1}(x^\sigma - \hat{\mu})^2 \ . \tag{252}$$

As a consequence of "Cochran's Theorem" (see Ref.[75]), q given by Eq.(252) must have a Chi-square probability distribution with $\nu = n-1$ degrees of freedom:[13]

$$P(q) = \chi^2(q; \nu = n-1) \tag{253}$$

Henceforth, let $\nu = n-1$. From the definition Eq.(251) of q, we get

$$dq = \frac{-2\nu}{\varrho^3} d\varrho \ . \tag{254}$$

Therefore,

$$\begin{aligned} P(\varrho)d\varrho &= P(q)dq & (255)\\ &= \chi^2(\frac{\nu}{\varrho^2};\nu)\frac{(-2\nu)}{\varrho^3}d\varrho & (256)\\ &= \mathcal{N}(!\varrho)\left(\frac{\nu}{\varrho^2}\right)^{\frac{\nu}{2}-1} e^{-\frac{\nu}{2\varrho^2}}\frac{d\varrho}{\varrho^3} & (257)\\ &= \mathcal{N}(!\varrho)\frac{d\varrho}{\varrho^{\nu+1}}e^{-\frac{\nu}{2\varrho^2}} \ . & (258) \end{aligned}$$

Hence,

$$\begin{aligned} P(t) &= \mathcal{N}(!t)\int_0^\infty \frac{d\varrho}{\varrho^{\nu+1}}e^{-\frac{\nu}{2\varrho^2}}e^{-\frac{1}{2}(\frac{t}{\varrho})^2} & (259)\\ &= \mathcal{N}(!t)\int_0^\infty \frac{d\varrho}{\varrho^{\nu+1}}e^{-\frac{1}{2}\frac{t^2+\nu}{\varrho^2}} \ . & (260) \end{aligned}$$

0.34 Hypothesis testing and 3 classic test statistics (Likelihood, Score, Wald)

Suppose we have data $\vec{x} = [x^\sigma]_{\sigma=0,1,\ldots,nsam-1}$ which is distributed according to a probability distribution $P(\vec{x};\theta)$ which depends on some parameters $\theta = (\theta_i)_{i=0,1,\ldots,n-1} \in \mathbb{R}^n$. We define the

[13]Note that this q is a quadratic form $q = \vec{x}^T M \vec{x}$, where $\vec{x}$ is an n dimensional column vector with components x^σ, and M is an $n \times n$ matrix. Cochran's Theorem diagonalizes M and replaces the vectors $\vec{x}$ by equivalent ones in a new basis. Then the number of DOFs (degrees of freedom) of the chi-square distribution is the number of non-zero diagonal elements in the diagonalized M (this number is called the rank of M). In the particular case of Eq.(252), $DOF = n-1$.

- **Likelihood of** θ by

$$L(\theta) = P(\vec{x}|\theta) \ , \tag{261}$$

- the **Maximum Likelihood estimate of** θ by

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\, L(\theta) \ , \tag{262}$$

- the **Log-likelihood of** θ by

$$LL(\theta) = \ln L(\theta) \ , \tag{263}$$

- the **Score** or **Lagrange Multiplier vector**[14] of θ by

$$sc(\theta) = [sc_i(\theta)] \in \mathbb{R}^n \ , \tag{264}$$

  where

$$sc_i(\theta) = \partial_{\theta_i} LL(\theta) \ , \tag{265}$$

- and the **Fisher Information Matrix of** θ by

$$FI(\theta) = [FI_{i,j}(\theta)] \in \mathbb{R}^{n\times n} \tag{266}$$

  where

$$FI_{i,j}(\theta) = -E_{\underline{\vec{x}}|\theta}[\partial_{\theta_i}\partial_{\theta_j} \overbrace{\ln P(\underline{\vec{x}}|\theta)}^{LL(\theta)}] \ . \tag{267}$$

  Note that if

$$LL(\theta) = \ln P(x|\theta) = \frac{-(\theta-\hat{\theta})^2}{2\sigma^2} + \mathcal{N}(!\theta) \ , \tag{268}$$

then

$$sc(\theta) = \frac{-(\theta-\hat{\theta})}{\sigma^2} \tag{269}$$

[14] $sc_i(\theta) = \partial_{\theta_i} LL(\theta)$ is called a Lagrange multiplier because to maximize $LL(\theta)$ over θ subject to the constraint $\theta = \theta^*$, we maximize the Lagrangian $\mathcal{L} = LL(\theta) - \sum_i \alpha_i[\theta_i - \theta_i^*]$ over (θ_i, α_i) for all i. The latter gives $\alpha_i = \partial_{\theta_i} LL(\theta^*)$ and $\theta = \theta^*$. For a general constrained optimization problem, $\mathcal{L} = LL(\theta) - \sum_i \alpha_i c_i(\theta)$ for some constraint functions $c_i(\theta)$. Hence, $\alpha_i = \partial_{\theta_i} LL/\partial_{\theta_i} c_i$ and $c_i(\theta) = 0$ for all i. So, in general, a Lagrange multiplier is a ratio of two partial derivatives.

and

$$FI(\theta) = \frac{1}{\sigma^2} \tag{270}$$

In hypothesis testing, one has a **null hypothesis** H_0: $\theta = \theta^*$, and an **alternative hypothesis** H_1: $\theta \neq \theta^*$. We use a **test statistic** $\lambda(\theta^*, \hat{\theta}) \geq 0$ which measures a kind of distance or separation between the estimate $\hat{\theta}$ and the value θ^* for the null Hypothesis H_0. For some **confidence level** $C > 0$, if $\lambda(\theta^*, \hat{\theta}) > C$, H_0 is **rejected**, whereas if $\lambda(\theta^*, \hat{\theta}) \leq C$, H_0 is **accepted**.

$\alpha = 1 - C$ is called the **significance level**. Usually $\alpha << 1$ represents the area under both tails of a Normal distribution, and $C \approx 1$ represents all the area except the tails of a Normal distribution.

Figure 10: For $n = 1$ and $\theta \in \mathbb{R}$, this figure shows the geometrical significance of certain quantities that characterize the 3 classic test statistics (Likelihood, Score, Wald) for hypothesis testing.

Henceforth in this section, we will occasionally use the Einstein summation convention; i.e., implicit sum over repeated indices.

Three classic test statistics are (See Fig.10):

1. **Likelihood Ratio test statistic** (Ref.[100].)

$$\lambda_{Li} = 2 \ln \left[\frac{L(\hat{\theta})}{L(\theta^*)} \right] = 2[LL(\hat{\theta}) - LL(\theta^*)] \tag{271}$$

2. **Score (a.k.a. Lagrange multiplier) test statistic** (Ref.[118].)

$$\lambda_{Sc} = \partial_{\theta_i} LL(\theta^*) \left[FI(\theta^*)^{-1} \right]_{i,j} \partial_{\theta_j} LL(\theta^*) \tag{272}$$

$$= \frac{[\partial_\theta LL(\theta^*)]^2}{FI(\theta^*)} \quad \text{if } n = 1 \tag{273}$$

Doesn't depend on $\hat{\theta}$.

3. **Wald test statistic** (Ref.[131].)

$$\lambda_{Wa} = (\hat{\theta} - \theta^*)_i \left[\left\langle \underline{\hat{\theta}}, \underline{\hat{\theta}}^T \right\rangle^{-1} \right]_{i,j} (\hat{\theta} - \theta^*)_j \tag{274}$$

$$= \frac{(\theta^* - \hat{\theta})^2}{\left\langle \hat{\theta}, \hat{\theta} \right\rangle} \quad \text{if } n = 1 \tag{275}$$

More generally, one can replace $\theta^* \to R\theta^*$ and $\hat{\theta} \to R\hat{\theta}$ in Eq.(274), where θ^* and $\hat{\theta}$ are n dimensional column vectors, and $R \in \mathbb{R}^{\nu \times n}$. The null and alternative hypotheses become: $H_0 : R\theta = R\theta^*$ and $H_1 : R\theta \neq R\theta^*$. Note that ν is the number of constraints imposed by the null hypothesis. R is called a reparametrization of θ. The Wald test is not reparametrization invariant (i.e., R invariant), but the Likelihood Ratio test is.

Note that if $LL(\theta)$ is given by Eq.(268), then $\left\langle \underline{\hat{\theta}}, \underline{\hat{\theta}} \right\rangle = \sigma^2 = \frac{1}{FI(\theta)}$. Hence,

$$\lambda_{Li} = \lambda_{Sc} = \lambda_{Wa} = \frac{(\hat{\theta} - \theta^*)^2}{\sigma^2} \tag{276}$$

Many other commonly used test statistics (or their squares) are special cases of one of the 3 classic test statistics. For example, the z-statistic used with normal distributions, the t-statistic used with the Student t-distribution, the F-statistic used in linear regression, the chi-squared statistic used to do Pearson's chi-squared test.

Asymptotic Behavior

If the data $\vec{x}$ is i.i.d.,

$$P(\vec{x}|\theta) = \prod_{\sigma=0}^{nsam-1} P(x^\sigma|\theta) \tag{277}$$

Hence, as $nsam \to \infty$,

$$LL(\theta) = \ln P(\vec{x}|\theta) \tag{278}$$

$$= \sum_\sigma \ln P(x^\sigma|\theta) \tag{279}$$

$$\to nsam \sum_x P(x|\theta) \ln P(x|\theta) \tag{280}$$

$$= -nsam\ H(\underline{x}|\theta) \tag{281}$$

Thus, *maximizing* the log likehood $LL(\theta)$ and *minimizing* the entropy $H(\underline{x}|\theta)$ give the same estimate $\hat{\theta}$.

When the data is i.i.d. and $nsam \to \infty$, it is also possible to prove that the 3 test statistics defined above all tend to the same probability distribution, namely $\mathcal{X}^2(\theta^*;\nu)$, the chi-square distribution with ν degrees of freedom, where $\theta \in \mathbb{R}^n$, $R \in \mathbb{R}^{\nu\times n}$, and $\nu = n$ if $R = 1$.

0.35 Error Bars

Never report measurements without error bars!! Quick reminder of error bars:

Normal distribution with mean μ and standard deviation σ:

$$\mathcal{N}(x;\mu,\sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \ . \tag{282}$$

Standard Normal Distribution (SND):

$$P(z) = \mathcal{N}(z;0,1) \tag{283}$$

Cumulative distribution for $P(z)$:

$$\Phi(z) = \int_{-\infty}^{z} dz' \ P(z') \ . \tag{284}$$

Figure 11: Interpretation of confidence level C and p-value as areas under curve of the Standard Normal Distribution (SND).

Confidence Level C and corresponding z^* **value** (see Fig.11):

$$C = \int_{-z^*}^{z^*} dz \ P(z) = \Phi(z^*) - \Phi(-z^*) = 2\left(\Phi(z^*) - \frac{1}{2}\right) \tag{285}$$

Equivalent definition:

$$C = P\left(\underbrace{\frac{|\underline{x}-\mu|}{\frac{\sigma}{\sqrt{n}}}}_{|z|} < z^*\right) \tag{286}$$

For $C = 95\%$, $z^* = 1.960 \approx 2$. For $C = 99\%$, $z^* = 2.576$.

Area of each tail in Fig.11 is usually called α, and the area of both tails is called the **p-value**:

$$C + \underbrace{2\alpha}_{p-value} = 1 \ . \tag{287}$$

Estimators[15] of mean μ and standard deviation σ from measurements x^σ of a sub-population Σ_1 of size $n = |\Sigma_1|$:

$$\hat{\mu} = \overline{x} = \frac{1}{n} \sum_{\sigma \in \Sigma_1} x^\sigma \tag{288}$$

$$\hat{\sigma}^2 = \frac{1}{n-1} \sum_{\sigma \in \Sigma_1} (x^\sigma - \overline{x})^2 \tag{289}$$

We get from Eq.(286), the **Error bars (aka confidence interval)** and **Error** E **(aka margin of error)**:

$$\text{estimate of } x \text{ with error bars} = \overline{x} \pm \underbrace{z^* \frac{\hat{\sigma}}{\sqrt{n}}}_{E} \tag{290}$$

$$n = \left(\frac{z^* \hat{\sigma}}{E} \right)^2 \tag{291}$$

So far, we have assumed that the sub-population (aka sample population) is normally distributed. This might be false for several reasons. Some red flags: (1) n is too small (according to a rule of thumb derived from Central Limit Theorem, n should be larger than 30 to insure a Normal Distribution). (2) Sub-population not truly random (i.i.d.) because was taken without replacement. In many cases, especially when $n < 30$, the Student's t-distribution models the sub-population statistics much better than the Normal distribution.

The Student's t-distribution $\text{Stud}(t; \nu = n-1)$, depends on a parameter ν called the number of degrees of freedom. In the case being considered here, ν equals the sub-population size n minus one. When fitting the data with Stud(), variable t replaces variable z, and $\text{Stud}(t; \nu = n-1)$ replaces the Standard Normal distribution (SND) $\mathcal{N}(z; \mu = 0, \sigma = 1)$. Stud() is symmetric about the origin like SND, but its tails are fatter. When fitting the data with Stud(), the z^* value is replaced by a t^* value. Eq.(285) is replaced by

$$C = \int_{-t^*}^{t^*} dt \ \text{Stud}(t) = \Phi_S(t^*) - \Phi_S(-t^*) = 2 \left(\Phi_S(t^*) - \frac{1}{2} \right) \ , \tag{292}$$

[15] Don't confuse the sample index σ with the standard deviation σ.

where $\Phi_S()$ is the cumulative distribution for Stud(). Also, Eq.(290) is replaced by

$$\text{estimate of } x \text{ with error bars} = \overline{x} \pm \underbrace{t^* \frac{\hat{\sigma}}{\sqrt{n}}}_{E} \ . \tag{293}$$

Tables of $t^*(C, \nu = n-1)$ are available. Note that t^* depends on both C and ν, whereas $z^*(C)$ depends only on C.

0.36 Short Summary of Boolean Algebra

See Ref.[70] for more info about this topic.

Suppose $x, y, z \in \{0, 1\}$. Define

$$x \text{ or } y = x \vee y = x + y - xy \ , \tag{294}$$

$$x \text{ and } y = x \wedge y = xy \ , \tag{295}$$

and

$$\text{not } x = \overline{x} = 1 - x \ , \tag{296}$$

where we are using normal addition and multiplication on the right hand sides.[16]

Actually, since $x \wedge y = xy$, we can omit writing the symbol $\wedge$. The symbol $\wedge$ is useful to exhibit the symmetry of the identities, and to remark about the analogous identities for sets, where $\wedge$ becomes intersection $\cap$ and $\vee$ becomes union $\cup$. However, for practical calculations, $\wedge$ is an unnecessary nuisance.

Since $x \in \{0, 1\}$,

$$P(\overline{x}) = 1 - P(x) \ . \tag{297}$$

Clearly, from analyzing the simple event space $(x, y) \in \{0, 1\}^2$,

$$P(x \vee y) = P(x) + P(y) - P(x \wedge y) \ . \tag{298}$$

[16] Note the difference between $\vee$ and modulus 2 addition $\oplus$. For $\oplus$ (aka XOR): $x \oplus y = x + y - 2xy$.

Associativity	$x \vee (y \vee z) = (x \vee y) \vee z$ $x \wedge (y \wedge z) = (x \wedge y) \wedge z$
Commutativity	$x \vee y = y \vee x$ $x \wedge y = y \wedge x$
Distributivity	$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$ $x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$
Identity	$x \vee 0 = x$ $x \wedge 1 = x$
Annihilator	$x \wedge 0 = 0$ $x \vee 1 = 1$
Idempotence	$x \vee x = x$ $x \wedge x = x$
Absorption	$x \wedge (x \vee y) = x$ $x \vee (x \wedge y) = x$
Complementation	$x \wedge \overline{x} = 0$ $x \vee \overline{x} = 1$
Double negation	$\overline{(\overline{x})} = x$
De Morgan Laws	$\overline{x} \wedge \overline{y} = \overline{(x \vee y)}$ $\overline{x} \vee \overline{y} = \overline{(x \wedge y)}$

Table 1: Boolean Algebra Identities

Definition of a Bayesian Network

A **directed graph** $G = (V, E)$ consists of two sets, V and E. V contains the **vertices (nodes)** and E contains the **edges (arrows)**. An arrow $\underline{a} \to \underline{b}$ is an ordered pair $(\underline{a}, \underline{b})$ where $\underline{a}, \underline{b} \in V$.

The **parents** of a node $\underline{x}$ are those nodes $\underline{a}$ such that there are arrows $\underline{a} \to \underline{x}$. The **children** of a node $\underline{x}$ are those nodes $\underline{b}$ such that there are arrows $\underline{x} \to \underline{b}$. A **root node** is a node with no parents. A **leaf node** is a node with no children. The **neighbors** of a node $\underline{x}$ is the set of parents and children of $\underline{x}$.

A **path** is a set of nodes that are connected by arrows, so that all nodes have 1 or 2 neighbors, but only two nodes (**open path**) or zero nodes (**closed path**) have only one neighbor. A **directed path** is a path in which all the arrows point in the same direction. A **loop** is a closed path;i.e., a path in which all nodes have exactly 2 neighbors. A **cycle** is a directed loop. A **Directed Acyclic Graph (DAG)** is a directed graph that has no cycles.

A **fully connected directed graph** is a directed graph in which every node has all other nodes as neighbors. Figs.12 and 13 show 2 different ways of drawing the same directed graph, a fully connected graph with 4 nodes. Note that a convenient way to label the nodes of a fully connected directed graph with N nodes is to point arrows from $\underline{x}_k$ to $\underline{x}_j$ where $j = 0, 1, 2, \ldots, N-1$ and $k = j-1, j-2, \ldots, 0$.

$$\begin{tikzcd}
\underline{x}_3 & \underline{x}_2 \arrow[l] & \underline{x}_1 \arrow[l] \arrow[ll, bend right] & \underline{x}_0 \arrow[l] \arrow[lll, bend right] \arrow[lll, bend right=60]
\end{tikzcd}$$

Figure 12: Fully connected directed graph with 4 nodes, drawn as a line.

$$\begin{tikzcd}
\underline{x}_0 \arrow[r] \arrow[d] \arrow[dr] & \underline{x}_1 \arrow[d] \arrow[dl] \\
\underline{x}_2 \arrow[r] & \underline{x}_3
\end{tikzcd}$$

Figure 13: Fully connected directed graph with 4 nodes, drawn as a square.

A **connected graph** is a graph for which there is no way of separating the nodes into two sets so that there is no arrow from one set to the other. A **tree** is a

directed graph in which all nodes have a single parent except for a single node called the "root" node which has no parents. A **polytree** is a DAG with no loops.

A **Bayesian network (bnet)** consists of a DAG and a **Transition Probability Matrix (TPM)** associated with each node of the graph. A TPM is often called a **Conditional Probability Table (CPT)**. The **structure** of a bnet is its DAG alone, sans the TPMs. The **skeleton** of a bnet is the undirected graph beneath the bnet's DAG.

In this book, random variables are indicated by underlined letters and their values by non-underlined letters. Each node of a bnet is labelled by a random variable. Thus, $\underline{x} = x$ means that node $\underline{x}$ is in state x.

Some sets of nodes associated with each node $\underline{a}$ of a bnet

- $ch(\underline{a})$ = children of $\underline{a}$.
- $pa(\underline{a})$ = parents of $\underline{a}$.
- $nb(\underline{a}) = pa(\underline{a}) \cup ch(\underline{a})$ = neighbors of $\underline{a}$.
- $de(\underline{a}) = \cup_{n=1}^{\infty} ch^n(\underline{a}) = ch(\underline{a}) \cup ch \circ ch(\underline{a}) \cup \ldots$, descendants of $\underline{a}$.
- $an(\underline{a}) = \cup_{n=1}^{\infty} pa^n(\underline{a}) = pa(\underline{a}) \cup pa \circ pa(\underline{a}) \cup \ldots$, ancestors of $\underline{a}$.

In this book, we will use $\underline{a}.$ to indicate a **multi-node (node set, node array)** $\underline{a}. = (\underline{a}_j)_{j=0,1,\ldots,na-1}$. We will often treat multinodes as if they were sets, and combine them with the usual set operators. For instance, for two multinodes $\underline{a}.$ and $\underline{b}.$, we define $\underline{a}. \cup \underline{b}.$, $\underline{a}. \cap \underline{b}.$, $\underline{a}. - \underline{b}.$ and $\underline{a}. \subset \underline{b}.$ in the obvious way. We will indicate a singleton set (single node multi-node) $\underline{a}. = \{\underline{a}\}$ simply by $\underline{a}. = \underline{a}$. For instance, $\underline{a}. - \underline{b} = \underline{a}. - \{\underline{b}\}$.

The TPM of a node $\underline{x}$ of a bnet is a matrix of probabilities $P(\underline{x} = x | pa(\underline{x}) = a.)$.

A bnet with nodes $\underline{x}.$ represents a probability distribution

$$P(x.) = \prod_j P(\underline{x}_j = x_j | (\underline{x}_k = x_k)_{k:\underline{x}_k \in pa(\underline{x}_j)}) \ . \tag{299}$$

Note that for a fully connected bnet with N nodes, Eq.(299) becomes

$$P(x.) = \prod_{j=0}^{N-1} P(x_j | (x_k)_{k=j-1,j-2,\ldots,0}) \ . \tag{300}$$

For example, if $N = 4$, Eq.(300) becomes

$$P(x_0, x_1, x_2, x_3) = P(x_3 | x_2, x_1, x_0) P(x_2 | x_1, x_0) P(x_1 | x_0) P(x_0) \ . \tag{301}$$

We see that Eq.(300) is just the chain rule for conditional probabilities.

In this book, we often use the following conventions for bnet instantiations:

Random variables are underlined and their values are not. For example, $\underline{a} = a$ means the random variable $\underline{a}$ takes the value a. Diagrams with nodes that are

underlined represent Bayesian Networks (bnets) and the same diagram with the letters not underlined represents a specific **instantiation** of that bnet. For example $\underline{a} \to \underline{b}$ represents the bnet with conditional probability distribution $P(b|a)$, whereas $a \to b$ represents $P(b|a)$ itself.

If $\underline{a}$ is a root node, then $\sum a$ signifies a weighted sum $\sum_a P(a)$. For example, $\sum a \to b = \sum_a P(a)P(b|a)$. If $\underline{a}$ is not a root node as in $x \to \sum a \to y = \sum_a P(y|a)P(a|x)$, then $\sum a$ signifies a simple unweighted sum $\sum_a$.

In this book, Unobserved nodes are indicated by enclosing them in a dashed circle. For example, $\overset{(}{\underset{)}{u}}$.

Given an arbitrarily large dataset of samples for the random variables $(\underline{x}_i)_{i=0,1,\ldots,N-1}$, there may be several bnets (differing in the direction of some arrows) that fit the data well. However, according to Pearl's causality theory, only one of these bnets is used by Nature. I like to refer to that single one as the **causally correct (CC) Bayesian network**.[17] In this book, whenever we speak of causal issues, we will assume, often without mentioning it, that the CC bnet is being used.[18]

[17] The uniqueness of a CC bnet can be taken to be an implicit axiom of causality theory. Alternatively, instead of assuming uniqueness, one can assume that we are using a DAG which is, according to some measure of goodness of causal fit, the best one among the DAGs of a given set of DAGs. (see Chapter 26).

[18] We won't use the term "causal bnet" in this book. Pearl defines a causal bnet to be a CC bnet that is also a "SCM" (i.e., a bnet whose internal nodes are deterministic and external ones are probabilistic.)

Bayesian Networks and the Passage of Time

I wrote an article about this in 2021 for my blog "Quantum Bayesian Networks". See Ref.[53]. For more info on 2 topics mentioned in the blog post, see Chapter 27 on Granger Causality and Chapter 26 on Goodness of Causal Fit.

Chapter 1

AdaBoost

This chapter is based on Ref.[60].

Adaptive Boosting (AdaBoost) is a method of constructing a strong classifier function as a linear combination of an ensemble of weak classifier functions.

Below, we will abbreviate "ensemble classifiers" by "e-classifiers" and "weak classifiers' by "w-classifiers".

Chapter 12 defines decision trees (dtrees) and explains how to construct them. A **tree stump** is a dtree with only one parent and 2 children nodes. Usually the w-classifier functions for AdaBoost come from tree stumps (because tree stumps are w-classifiers and simple to compute), but the core AdaBoost algorithm is oblivious to where the w-classifier functions came from.

Boosting (see this chapter on AdaBoost and Chapter 76 on XGBoost) and bagging (see Chapter 54 on Random Forest) are two methods of building a classifier function from an ensemble of classifier functions. These two methods are most commonly applied to dtrees: Boosting for an ensemble of small dtrees, and Bagging for a random forest (which is an ensemble of dtrees that are usually much more complicated than small dtrees).

1.1 AdaBoost for general ensemble of w-classifiers

In this section we discuss the core AdaBoost algorithm, valid for a generic ensemble of w-classifiers.

Let $L = [0, 1, 2, \ldots, nsam-1]$ be a list of individuals (samples) in a population. In this chapter, we will use the notation $A^\sigma = A[\sigma]$ and $\vec{A} = [A^\sigma : \sigma \in L]$ for a list (vector, 1-D array) indexed by L. We will refer to $DS = (\vec{x}, \vec{y})$ where $x^\sigma \in S_{\underline{x}}$, $y^\sigma \in S_{\underline{y}}$, as a dataset. Let $T = \{0, 1, \ldots, nt-1\}$. Let $x^\sigma = (x_0^\sigma, x_1^\sigma, \ldots, x_{nt-1}^\sigma) \in S_{\underline{x}_0} \times S_{\underline{x}_1} \times \ldots \times S_{\underline{x}_{nt-1}} = S_{\underline{x}}$.

AdaBoost assumes that the classifier class set $S_{\underline{y}}$ and all the feature sets $S_{\underline{x}_t}$ are binary: $S_{\underline{y}} = S_{\underline{x}_t} = \{-1, 1\}$ for all $t \in T$.

Suppose that we are given an ensemble of nt **w-classifiers** $Y_t : S_{\underline{x}} \to \{-1, 1\}$, where $t \in T$. Suppose we want to find **intermediate e-classifiers** $\mathcal{Y}_t : S_{\underline{x}} \to \mathbb{R}$ given

by

$$\mathcal{Y}_t(x^\sigma) = \sum_{t'=0}^{t} \alpha_{t'} Y_{t'}(x^\sigma) \in \mathbb{R} \tag{1.1}$$

for $t \in T$ and a **final e-classifier** given by

$$\mathcal{Y}_{fin}(x^\sigma) = \text{sign}(\mathcal{Y}_{nt-1}(x^\sigma)) \in \{-1, 1\} \ . \tag{1.2}$$

The AdaBoost algorithm yields a set of coefficients α_t for which the final e-classifier is much stronger (i.e., less error prone) than any of the w-classifiers of the ensemble.

Note that

$$\mathcal{Y}_t(x^\sigma) = \mathcal{Y}_{t-1}(x^\sigma) + \alpha_t Y_t(x^\sigma) \tag{1.3}$$

for $t \in T$ if we define $\mathcal{Y}_{-1} = 0$. Hence

$$\underbrace{e^{-y^\sigma \mathcal{Y}_t(x^\sigma)}}_{Z_t w^\sigma_{t+1}} = \underbrace{e^{-y^\sigma \mathcal{Y}_{t-1}(x^\sigma)}}_{Z_{t-1} w^\sigma_t} e^{-\alpha_t y^\sigma Y_t(x^\sigma)} \tag{1.4}$$

where the **weights** w^σ_t and and the **partition function** Z_t are defined by

$$w^\sigma_{t+1} = \begin{cases} 1/nsam & \text{for } t = -1 \\ \frac{\exp(-y^\sigma \mathcal{Y}_t(x^\sigma))}{Z_t} & \text{for } t \geq 0 \end{cases} \tag{1.5}$$

and

$$Z_t = \sum_\sigma e^{-y^\sigma \mathcal{Y}_t(x^\sigma)} \ . \tag{1.6}$$

Note that the probability distribution $P(\sigma|t+1) = w^\sigma_{t+1}$ of weights at time $t+1$ emphasizes (i.e., gives higher probability to) the errors (i.e., occurrences of $y^\sigma \mathcal{Y}_t(x^\sigma) = -1$ for some population individual σ) of the previous (i.e., at time t) intermediate e-classifier $\mathcal{Y}_t$. In other words, every new intermediate e-classifier $\mathcal{Y}_{t+1}$ concentrates on those individuals σ on which the previous e-classifier $\mathcal{Y}_t$ performed poorly.

Note also that the partition function Z_t is a good measure of the **classification error** (i.e., occurrences of $y^\sigma \mathcal{Y}_t(x^\sigma) = -1$) of $\mathcal{Y}_t$. We will therefore use Z_t for that purpose, as an error measure.

For $t > 1$, we have

$$\begin{aligned} Z_t &= \sum_\sigma e^{-y^\sigma \mathcal{Y}_t(x^\sigma)} && (1.7) \\ &= \sum_\sigma \underbrace{e^{-y^\sigma \mathcal{Y}_{t-1}(x^\sigma)}}_{Z_{t-1} w^\sigma_t} e^{-\alpha_t y^\sigma Y_t(x^\sigma)} && (1.8) \\ &= Z_{t-1} E_\sigma[e^{-\alpha_t y^\sigma Y_t(x^\sigma)}] && (1.9) \end{aligned}$$

Define the **success rate** by

$$S_t = \sum_\sigma w_t^\sigma \mathbb{1}(y^\sigma Y_t(x^\sigma) = 1) \tag{1.10}$$

$$= E_\sigma[\mathbb{1}(\underbrace{y^\sigma Y_t(x^\sigma) = 1}_{\text{iff } y^\sigma = Y_t(x^\sigma)})] \tag{1.11}$$

and the **failure rate** by

$$F_t = \sum_\sigma w_t^\sigma \mathbb{1}(y^\sigma Y_t(x^\sigma) = -1) \tag{1.12}$$

$$= E_\sigma[\mathbb{1}(\underbrace{y^\sigma Y_t(x^\sigma) = -1}_{\text{iff } y^\sigma \neq Y_t(x^\sigma)})] \ . \tag{1.13}$$

Note that

$$S_t + F_t = 1 \ , \tag{1.14}$$

and

$$Z_t = Z_{t-1}(e^{-\alpha_t} S_t + e^{+\alpha_t} F_t) \ . \tag{1.15}$$

Figure 1.1: Plot of function $\alpha_t = \frac{1}{2} \ln \frac{1-F_t}{F_t}$.

We can find the α_t values that minimize the classification error Z_t, and then evaluate Z_t at those optimum α_t values:

$$\frac{dZ_t}{d\alpha_t} = Z_{t-1}(-e^{-\alpha_t} S_t + e^{\alpha_t} F_t) = 0 \tag{1.16}$$

$$e^{2\alpha_t} = \frac{S_t}{F_t} \tag{1.17}$$

$$\alpha_t = \frac{1}{2}\ln\frac{S_t}{F_t} = \frac{1}{2}\ln\frac{1-F_t}{F_t} \tag{1.18}$$

$$\frac{Z_t}{Z_{t-1}} = 2\sqrt{S_t F_t} = 2\sqrt{(1-F_t)F_t} \leq 1 \tag{1.19}$$

$f(x) = 2\sqrt{(1-x)x}$ for $x \in [0,1]$ is dome shaped and its maximum is 1, which is achieved iff $x = \frac{1}{2}$

Figure 1.2: Bnet for AdaBoost with 5 w-classifiers, $nt = 5$. All nodes labelled $(\vec{x}, \vec{y})$ are the same node.

The AdaBoost algo described in this chapter is summarized by the bnet of Fig.1.2. The TPMs, printed in blue, for the nodes of that bnet, are as follows:

$$P(w_0^\sigma) = \frac{1}{nsam} \tag{1.20}$$

for all σ.

$$P(\vec{w}_{t+1}|\vec{w}_t, \alpha_t, Y_t, \vec{x}, \vec{y}) = \prod_\sigma \mathbb{1}\left(\quad w_{t+1}^\sigma = \frac{w_t^\sigma e^{-\alpha_t y^\sigma Y_t(x^\sigma)}}{\sum_\sigma numerator} \quad \right) \tag{1.21}$$

for $t \geq 0$.

$$P(\alpha_t|\vec{w}_t, \vec{x}, \vec{y}) = \mathbb{1}\left(\quad \alpha_t = \frac{1}{2}\ln\frac{1-F_t}{F_t} \text{ where } F_t = F_t(\vec{w}_t, \vec{x}, \vec{y}) \quad \right) \tag{1.22}$$

for $t \in T$.

$$P(\mathcal{Y}_t|\mathcal{Y}_{t-1}, \alpha_t, Y_t) = \mathbb{1}\left(\quad \mathcal{Y}_t = \mathcal{Y}_{t-1} + \alpha_t Y_t \quad \right) \tag{1.23}$$

for $t \in T$, where $\mathcal{Y}_{-1} = 0$.

1.2 AdaBoost for ensemble of tree stumps

Keep in mind that AdaBoost assumes $S_{\underline{y}} = S_{\underline{x}_t} = \{-1, 1\}$ for all $t \in T = \{0, 1, \ldots, nt-1\}$. In order to implement AdaBoost, we need to specify nt w-classifiers $Y_t : \{-1, 1\}^{nt} \to \{-1, 1\}$ for $t \in T$. One can either specify the nt w-classifiers a priori or build them on-the-fly.

- **w-classifiers specified a priori**

  Define a classifier for each feature x_t where $t \in T$ by:

  $$Y_t(x^\sigma) = x_t^\sigma \in \{-1, 1\} \tag{1.24}$$

  Hence, for this classifier, $y^\sigma Y_t(x^\sigma) = y^\sigma x_t^\sigma$.

- **w-classifiers built on-the-fly**

  Recall dataset $(\vec{x}, \vec{y}) = [(x^\sigma, y^\sigma) : \sigma \in L]$ is indexed by the list $L = [0, 1, \ldots, nsam-1]$. If L_j is a list (possibly with duplicate items) such that $set(L_j) \subset set(L)$, then define $DS_j = (\vec{x}, \vec{y})_{L_j} = ((x^\sigma)_{\sigma \in L_j}, (y^\sigma)_{\sigma \in L_j})$. We will refer to DS_j as the **restriction of $(\vec{x}, \vec{y})$ to L_j.**

  The idea behind on-the-fly w-classifiers is to choose $Y_t(x^\sigma) = x_t^\sigma$, where x_t is the feature with the lowest Gini in the current dataset $(\vec{x}, \vec{y})_{L_t}$. To build $(\vec{x}, \vec{y})_{L_t}$, we select at random from $L = [0, 1, \ldots, nsam - 1]$, a list L_t of the same length as L, using the probability distribution $\vec{w}_{t-1}$. By choosing L_t with probabilities $\vec{w}_{t-1}$, we emphasize individuals σ that are failing. Then we define $(\vec{x}, \vec{y})_{L_t}$ as the restriction of $(\vec{x}, \vec{y})$ to L_t.

  Perhaps a causal diagram will make all these new steps clearer to the reader. The bnet Fig.1.3 is a modification of the bnet Fig.1.2 to include these new steps. The TPMs, printed in blue, for new or changed nodes, are as follows:

  $$P(Y_t|(\vec{x}, \vec{y})_{L_t}) = \mathbb{1}(\quad Y_t(x^\sigma) = x_t^\sigma \text{ where } x_t \text{ is feature in } (\vec{x}, \vec{y})_{L_t} \text{ the with lowest Gini.} \quad) \tag{1.25}$$

  $$P(L_{t+1}^\sigma = \sigma'|\vec{w}_t) = w_t^{\sigma'} \tag{1.26}$$

  $$P((\vec{x}, \vec{y})_{L_t}|L_t, (\vec{x}, \vec{y})) = \mathbb{1}(\quad (\vec{x}, \vec{y})_{L_t} = \text{ restriction of } (\vec{x}, \vec{y}) \text{ to } L_t \quad) \tag{1.27}$$

Figure 1.3: Modification of the bnet of Fig.1.2 to include on-the-fly generation of the w-classifiers.

Chapter 2

ARACNE structure learning

This chapter is based on Ref.[23].

The ARACNE algo is an algo for learning the structure of a bnet from data. The algo considers data samples for n random variables $(x_i)_{i=0,1,\ldots,n-1}$, and estimates the mutual information $MI_{i,j} = H(\underline{x}_i : \underline{x}_j)$ between every pair of nodes. The set UG is initialized to contain all the edges of a fully connected undirected graph. Next the algo removes from UG every edge with $MI_{i,j} < \epsilon$ for some threshold $0 < \epsilon << 1$. Then the algo examines every triplet of edges in UG, and marks for removal the edge of the triplet with the smallest MI. Finally, the algo removes from UG all edges marked for removal. Each triplet is analyzed irrespective of whether its edges have been marked for removal when considering a prior triplet. Thus the network constructed by the algorithm is independent of the order in which the triplets are examined. Some of the unoriented edges in UG can be given an orientation using the same techniques used to orient edges in constraint based structure learning (see Chapter 66).

Ref.[23] incorrectly claims that removing the smaller of 3 MI's is "an application" of the Data Processing Inequality (DPI) of Shannon Information Theory. See Chapter 39 for more info about DPI. Note that DPI is only valid for a Markov chain, and not all triplets of random variables are in a Markov chain. Removing the smaller of 3 numbers does not require DPI.

Fig.2.1 gives an example of the application of the ARACNE algo.

See Chapter 9 on Chow-Liu trees (CLT). A CLT is just a maximum spanning tree where the weights are mutual informations $MI_{i,j}$ estimated from data.

Sometimes, the outcome of the ARACNE algo is a CLT. For example, Fig.2.1 (a) was considered in Chapter 9 on CLTs, and the CLT algo also gave Fig.2.1 (c) as the final structure.

According to Ref.[23], the ARACNE algo sometimes yields a polytree (i.e., a connected graph with no loops). It may even yield a structure with loops. Hence, it does not always yield a CLT.

By breaking all cliques (i.e., fully connected subgraphs) with 3 edges and 3 nodes, ARACNE breaks all cliques with 3 or more nodes. However, cliques are not uncommon in Nature, especially 3 node cliques. Cliques become less likely in Nature

Figure 2.1: An example where the ARACNE algo gives a Chow-Liu tree. (a) Fully connected undirected graph with weights $MI_{i,j}$ along the edges. (b) All 4 possible triplets of edges with nonzero weights. Edges marked for removal have their weights printed in red.(c) Final structure.

the bigger the number of nodes they have *after* 3. Therefore, a nice generalization of ARACNE would be to list all 4 node cliques, and break each of them by eliminating their edge with the smallest MI. This will have the effect of breaking all cliques with 4 or more nodes but keeping 3 node cliques. One could also break some, not all, of the 3 node cliques, by consecutively removing the clique-breaking-edge with the smallest MI of all edges of all 3 node cliques. Let β stand for banned clique number of nodes. Then the current ARACNE has $\beta = 3$. We are suggesting that a β of 4 might be more likely to occur in Nature.

Chapter 3

Backdoor Adjustment Formula

The backdoor (BD) adjustment formula is proven in Chapter 15 from the rules of Do Calculus. The goal of this chapter is to give examples of the use of that theorem. We will restate the theorem in this chapter, sans proof. There is no need to understand the theorem's proof in order to use it. However, you will need to skim Chapter 15 in order to familiarize yourself with the notation used to state the theorem. This chapter also assumes that you are comfortable with the rules for checking for d-separation. Those rules are covered in Chapter 17.

Suppose that we have access to data that allows us to estimate a probability distribution $P(x., y., z.)$. Hence, the variables $\underline{x}., \underline{y}., \underline{z}.$ are ALL observed (i.e, not hidden). Then we say that the backdoor $\underline{z}.$ satisfies the **backdoor adjustment criterion** relative to $(\underline{x}., \underline{y}.)$ if

1. All backdoor paths from $\underline{x}.$ to $\underline{y}.$ are blocked by conditioning on $\underline{z}.$.
2. $\underline{z}. \cap de(\underline{x}.) = \emptyset$.

Claim 13 Backdoor Adjustment Formula

If $\underline{z}.$ satisfies the backdoor criterion relative to $(\underline{x}., \underline{y}.)$, then

$$P(y.|\mathcal{D}\underline{x}. = x.) = \sum_{z.} P(y.|x., z.)P(z.) \tag{3.1}$$

$$= \quad \sum z. \tag{3.2}$$

$$\begin{tikzcd} & \\ x. \arrow[r] & y. \end{tikzcd}$$

where $\sum z.$ means node $\underline{z}.$ is summed over.

proof: See Chapter 15.
QED

3.1 Examples

1.

$$\begin{tikzcd} & \underline{z} \arrow[dr] & \\ \underline{x} \arrow[ur] \arrow[rr] & & \underline{y} \end{tikzcd} \tag{3.3}$$

BD criterion satisfied if $\underline{x}. = \underline{x}, \underline{y}. = \underline{y}, \underline{z}. = \emptyset$. No adjustment necessary.

$$P(y|\mathcal{D}\underline{x} = x) = P(y|x) \tag{3.4}$$

2.

$$\begin{tikzcd} & \underline{z} \arrow[dl] \arrow[dr] & \\ \underline{x} \arrow[rr] & & \underline{y} \end{tikzcd} \tag{3.5}$$

BD criterion satisfied if $\underline{x}. = \underline{x}, \underline{y}. = \underline{y}, \underline{z}. = \underline{z}$.

Note that here the backdoor formula adjusts the parents of $\underline{x}$..

3.

$$\begin{tikzcd} & \underline{z} \arrow[dl] \arrow[dr] & \\ \underline{x} \arrow[r] & \underline{m} \arrow[r] & \underline{y} \end{tikzcd} \tag{3.6}$$

BD criterion satisfied if $\underline{x}. = \underline{x}, \underline{y}. = \underline{y}, \underline{z}. = \underline{z}$.

4.

$$\begin{tikzcd} & \underline{z} \arrow[dl] \arrow[dr] & \\ \underline{x} \arrow[r] & \underline{m} \arrow[r] & \underline{y} \end{tikzcd} \tag{3.7}$$

BD criterion is impossible to satisfy if $\underline{x}. = \underline{x}, \underline{y}. = \underline{y}$. However, the frontdoor criterion can be satisfied. See Chapter 23.

5.

$$\begin{tikzcd} \underline{w} \arrow[r] \arrow[d] & \underline{z} \arrow[d] \\ \underline{x} \arrow[r] & \underline{y} \end{tikzcd} \tag{3.8}$$

BD criterion satisfied if $\underline{x}. = \underline{x}, \underline{y}. = \underline{y}, \underline{z}. = \underline{z}$. We are able to block the backdoor path by conditioning on $\underline{z}$.

6.

$$\begin{tikzcd}
\underline{e} \arrow[r] \arrow[d] & \underline{z} \arrow[dl] \arrow[dr] & \underline{a} \arrow[l] \arrow[d] \\
\underline{x} \arrow[rr] & & \underline{y}
\end{tikzcd} \tag{3.9}$$

Conditioning on $\underline{z}$ blocks backdoor path $\underline{x}-\underline{z}-\underline{y}$, but opens path $\underline{x}-\underline{e}-\underline{z}-\underline{a}-\underline{y}$ because $\underline{z}$ is a collider for that path. That path is blocked if we also condition on $\underline{a}$, which is possible because $\underline{a}$ is observed. In conclusion, the BD criterion is satisfied if $\underline{x}. = \underline{x}$, $\underline{y}. = \underline{y}$ and $\underline{z}. = (\underline{z}, \underline{a})$.

Conditioning on the parents of $\underline{x}.$ is often enough to block all backdoor paths. However, sometimes some of the parents are unobserved and one must condition on other nodes that are not parents of $\underline{x}.$ in order to satisfy the BD criterion.

7.

$$\begin{tikzcd}
\underline{z} \arrow[d] & & \underline{t} \arrow[ll] \arrow[d] \\
\underline{w} & \underline{x} \arrow[l] \arrow[r] & \underline{y}
\end{tikzcd} \tag{3.10}$$

No need to control anything because only possible backdoor path is blocked by not conditioning on collider $\underline{w}$. Hence,

$$P(y|\mathcal{D}\underline{x} = x) = P(y|x) \ . \tag{3.11}$$

However, if for some reason we want to control $\underline{t}$, we can do so. We can't control $\underline{w}$ though, because $\underline{w} \in de(\underline{x})$. Thus, the BD criterion is satisfied if $\underline{x}. = \underline{x}$, $\underline{y}. = \underline{y}$ and $\underline{z}. = \underline{t}$. Therefore,

$$P(y|\mathcal{D}\underline{x} = x) = \sum_t P(y|x,t)P(t) \ . \tag{3.12}$$

---

8. Discuss what to do if several sets $\underline{z}.$ satisfy the BD criterion.

   - Can evaluate $P(y.|\mathcal{D}\underline{x}. = x.)$ multiple ways and compare the results. This is a test that the causal bnet is correct.
   - It might be easier or less expensive to get data for some $\underline{z}.$ more than for others.

---

9. (Taken from online course notes Ref.[15])

Consider the bnet

$$\begin{tikzcd}
\underline{x}_2 \arrow[r] \arrow[d] & \underline{x}_3 & \underline{x}_4 \arrow[l] \arrow[d] & \\
\underline{x}_1 \arrow[r] \arrow[d] & \underline{x}_6 \arrow[r] \arrow[d] & \underline{x}_5 \arrow[d] & \underline{x}_7 \arrow[l] \\
\underline{x}_8 & \underline{x}_9 & \underline{x}_{10} &
\end{tikzcd} \tag{3.13}$$

If $\underline{x}. = \underline{x}_1$ and $\underline{y}. = \underline{x}_5$, find all possible adjustment multinodes $\underline{z}.$ that satisfy the BD criterion. Ans:

- $\emptyset$
- $\underline{x}_2$
- $\underline{x}_4$
- $\underline{x}_2, \underline{x}_4$
- $\underline{x}_2, \underline{x}_3$
- $\underline{x}_3, \underline{x}_4$
- $\underline{x}_2, \underline{x}_3, \underline{x}_4$

Add $\underline{x}_7$ to each of the previous 7 possible $\underline{z}.$. This gives a total of 14 possible adjustment multinodes $\underline{z}.$.

Chapter 4

Back Propagation (Automatic Differentiation)

4.1 General Theory

4.1.1 Jacobians

Suppose $f : \mathbb{R}^{nx} \to \mathbb{R}^{nf}$ and

$$y = f(x) \ . \tag{4.1}$$

Then the Jacobian $\frac{\partial y}{\partial x}$ is defined as the matrix with entries[1]

$$\left[\frac{\partial y}{\partial x}\right]_{i,j} = \frac{\partial y_i}{\partial x_j} \ . \tag{4.2}$$

Jacobian of function composition. Suppose $f : \mathbb{R}^{nx} \to \mathbb{R}^{nf}$, $g : \mathbb{R}^{nf} \to \mathbb{R}^{ng}$. If

$$y = g \circ f(x) \ , \tag{4.3}$$

then

$$\frac{\partial y}{\partial x} = \frac{\partial g}{\partial f}\frac{\partial f}{\partial x} \ . \tag{4.4}$$

Right hand side of last equation is a product of two matrices so order of matrices is important.

(a) Composition

(b) Forward-p

(c) Back-p

Figure 4.1: bnets for function composition, forward propagation and back propagation for $nf = 5$ nodes.

4.1.2 Bnets for function composition, forward propagation and back propagation

Let

$$y = f^4 \circ f^3 \circ f^2 \circ f^1(x) \ . \tag{4.5}$$

This function composition chain can be represented by the bnet Fig.4.1(a) with TPMs

$$P(f^\mu | f^{\mu-1}) = \mathbb{1}(f^\mu = f^\mu(f^{\mu-1})) \tag{4.6}$$

for $\mu = 1, 2, 3, 4$.

[1] Mnemonic for remembering order of indices: i in numerator/j in denominator becomes index i/j of Jacobian matrix.

Note that

$$\begin{aligned}
\frac{\partial y}{\partial x} &= \frac{\partial y}{\partial f^3}\frac{\partial f^3}{\partial f^2}\left[\frac{\partial f^2}{\partial f^1}\frac{\partial f^1}{\partial x}\right] && (4.7)\\
&= \frac{\partial y}{\partial f^3}\left[\frac{\partial f^3}{\partial f^2}\frac{\partial f^2}{\partial x}\right] && (4.8)\\
&= \left[\frac{\partial y}{\partial f^3}\frac{\partial f^3}{\partial x}\right] && (4.9)\\
&= \frac{\partial y}{\partial x}\,. && (4.10)
\end{aligned}$$

This forward propagation can be represented by the bnet Fig.4.1(b) with node TPMs

$$P(\frac{\partial f^{\mu+1}}{\partial x} \mid \frac{\partial f^{\mu}}{\partial x}) = \mathbb{1}(\frac{\partial f^{\mu+1}}{\partial x} = \frac{\partial f^{\mu+1}}{\partial f^{\mu}}\frac{\partial f^{\mu}}{\partial x}) \tag{4.11}$$

for $\mu = 1, 2, 3$.

Note that

$$\begin{aligned}
\frac{\partial y}{\partial x} &= \left[\frac{\partial y}{\partial f^3}\frac{\partial f^3}{\partial f^2}\right]\frac{\partial f^2}{\partial f^1}\frac{\partial f^1}{\partial x} && (4.12)\\
&= \left[\frac{\partial y}{\partial f^2}\frac{\partial f^2}{\partial f^1}\right]\frac{\partial f^1}{\partial x} && (4.13)\\
&= \left[\frac{\partial y}{\partial f^1}\frac{\partial f^1}{\partial x}\right] && (4.14)\\
&= \frac{\partial y}{\partial x}\,. && (4.15)
\end{aligned}$$

This back propagation can be represented by the bnet Fig.4.1(c) with node TPMs

$$P(\frac{\partial y}{\partial f^{\mu}} \mid \frac{\partial y}{\partial f^{\mu+1}}) = \mathbb{1}(\frac{\partial y}{\partial f^{\mu}} = \frac{\partial y}{\partial f^{\mu+1}}\frac{\partial f^{\mu+1}}{\partial f^{\mu}}) \tag{4.16}$$

for $\mu = 2, 1, 0$.

$\frac{\partial f^{\mu+1}}{\partial f^{\mu}}$ is a Jacobian matrix so the order of multiplication matters. In forward prop, it pre-multiplies, and in back prop it post-multiplies.

4.2 Application to Neural Networks

4.2.1 Absorbing b_i^{λ} into $w_{i|j}$.

Below are, printed in blue, the TPMs for the nodes of a NN bnet, as given in Chapter 48.

Figure 4.2: Nodes $\underline{h}_0^0, \underline{h}_0^1, \underline{Y}_0$ are all set to 1. They allow us to absorb b_i^λ into the first column of $w_{i|j}^\lambda$.

For all hidden layers $\lambda = 0, 1, \ldots, \Lambda - 2$,

$$P(h_i^\lambda \mid h_\cdot^{\lambda-1}) = \delta\left(h_i^\lambda, \mathcal{A}_i^\lambda(\sum_j w_{i|j}^\lambda h_j^{\lambda-1} + b_i^\lambda)\right) \tag{4.17}$$

for $i = 0, 1, \ldots, nh(\lambda) - 1$. For the output visible layer $\lambda = \Lambda - 1$:

$$P(Y_i \mid h_\cdot^{\Lambda-2}) = \delta\left(Y_i, \mathcal{A}_i^{\Lambda-1}(\sum_j w_{i|j}^{\Lambda-1} h_j^{\Lambda-2} + b_i^{\Lambda-1})\right) \tag{4.18}$$

for $i = 0, 1, \ldots, ny - 1$.

For each λ, replace the matrix $w_{\cdot|\cdot}^\lambda$ by the augmented matrix $[b^\lambda., w_{\cdot|\cdot}^\lambda]$ so that the new $w_{\cdot|\cdot}^\lambda$ satisfies

$$w_{i|0}^\lambda = b_i^\lambda \tag{4.19}$$

Let the nodes $\underline{h}_0^\lambda$ for all λ and $\underline{Y}_0$ be root nodes (so no arrows pointing into them). For each λ, draw arrows from $\underline{h}_0^\lambda$ to all other nodes in that same layer. Draw arrows from $\underline{Y}_0$ to all other nodes in that same layer.

After performing the above steps, the TPMs, printed in blue, for the nodes of the NN bnet are as follows:

For all hidden layers $\lambda = 0, 1, \ldots, \Lambda - 2$,

$$P(h_0^\lambda) = \delta(h_0^\lambda, 1)\ , \tag{4.20}$$

and

$$P(h_i^\lambda \mid h_\cdot^{\lambda-1}, h_0^\lambda = 1) = \delta\left(h_i^\lambda, \mathcal{A}_i^\lambda(\sum_j w_{i|j}^\lambda h_j^{\lambda-1})\right) \tag{4.21}$$

for $i = 1, \ldots, nh(\lambda) - 1$. For the output visible layer $\lambda = \Lambda - 1$:

$$P(Y_0) = \delta(Y_0, 1)\ , \tag{4.22}$$

and

$$P(Y_i \mid h_{\cdot}^{\Lambda-2}, Y_0 = 1) = \delta\left(Y_i, \mathcal{A}_i^{\Lambda-1}(\sum_j w_{i|j}^{\Lambda-1} h_j^{\Lambda-2})\right) \tag{4.23}$$

for $i = 1, 2, \ldots, ny - 1$.

4.2.2 Bnets for function composition, forward propagation and back propagation for NN

$$\underline{\mathcal{A}^3} \longleftarrow \underline{\mathcal{B}^3} \longleftarrow \underline{\mathcal{A}^2} \longleftarrow \underline{\mathcal{B}^2} \longleftarrow \underline{\mathcal{A}^1} \longleftarrow \underline{\mathcal{B}^1} \longleftarrow \underline{\mathcal{A}^0} \longleftarrow \underline{\mathcal{B}^0} \longleftarrow \underline{x}$$

(a)

$$\underline{\frac{\partial \mathcal{A}^3}{\partial x}} \longleftarrow \underline{\frac{\partial \mathcal{B}^3}{\partial x}} \longleftarrow \underline{\frac{\partial \mathcal{A}^2}{\partial x}} \longleftarrow \underline{\frac{\partial \mathcal{B}^2}{\partial x}} \longleftarrow \underline{\frac{\partial \mathcal{A}^1}{\partial x}} \longleftarrow \underline{\frac{\partial \mathcal{B}^1}{\partial x}} \longleftarrow \underline{\frac{\partial \mathcal{A}^0}{\partial x}} \longleftarrow \underline{\frac{\partial \mathcal{B}^0}{\partial x}} \longleftarrow \underline{1}$$

(b)

$$\underline{1} \longrightarrow \underline{\frac{\partial Y}{\partial \mathcal{B}^3}} \longrightarrow \underline{\frac{\partial Y}{\partial \mathcal{A}^2}} \longrightarrow \underline{\frac{\partial Y}{\partial \mathcal{B}^2}} \longrightarrow \underline{\frac{\partial Y}{\partial \mathcal{A}^1}} \longrightarrow \underline{\frac{\partial Y}{\partial \mathcal{B}^1}} \longrightarrow \underline{\frac{\partial Y}{\partial \mathcal{A}^0}} \longrightarrow \underline{\frac{\partial Y}{\partial \mathcal{B}^0}} \longrightarrow \underline{\frac{\partial Y}{\partial x}}$$

(c)

Figure 4.3: bnets for (a) function composition, (b) forward propagation and (c) back propagation for a neural net with 4 layers (3 hidden and output visible).

.

From here on, we will rename y above by $Y = \hat{y}$ and consider samples $y[i]$ for $i = 0, 1, \ldots, nsam - 1$. The Error (aka loss or cost function) is

$$\mathcal{E} = \frac{1}{nsam} \sum_{\sigma=0}^{nsam-1} \sum_{i=0}^{ny-1} |Y_i - y_i[\sigma]|^2 \tag{4.24}$$

To perform simple gradient descent, one uses:

$$(w_{i|j}^{\lambda})' = w_{i|j}^{\lambda} - \eta \frac{\partial \mathcal{E}}{\partial w_{i|j}^{\lambda}} \ . \tag{4.25}$$

One has

$$\frac{\partial \mathcal{E}}{\partial w^{\lambda}_{i|j}} = \frac{1}{nsam} \sum_{\sigma=0}^{nsam-1} \sum_{i=0}^{ny-1} 2(Y_i - y_i[\sigma]) \frac{\partial Y}{\partial w^{\lambda}_{i|j}} \ . \tag{4.26}$$

Define $\mathcal{B}^{\lambda}_i$ thus

$$\mathcal{B}^{\lambda}_i(h^{\lambda-1}) = \sum_j w^{\lambda}_{i|j} h^{\lambda-1}_j \ . \tag{4.27}$$

Then

$$\frac{\partial Y}{\partial w^{\lambda}_{i|j}} = \frac{\partial Y}{\partial \mathcal{B}^{\lambda}_i} \frac{\partial \mathcal{B}^{\lambda}_i}{\partial w^{\lambda}_{i|j}} \tag{4.28}$$

$$= \frac{\partial Y}{\partial \mathcal{B}^{\lambda}_i} h^{\lambda-1}_j \tag{4.29}$$

$$\frac{\partial \mathcal{E}}{\partial w^{\lambda}_{i|j}} = \frac{\partial \mathcal{E}}{\partial \mathcal{B}^{\lambda}_j} \frac{\partial \mathcal{B}^{\lambda}_j}{\partial w^{\lambda}_{i|j}} \tag{4.30}$$

$$= \frac{\partial \mathcal{E}}{\partial \mathcal{B}^{\lambda}_j} h^{\lambda-1}_j \ . \tag{4.31}$$

This suggest that we can calculate the derivatives of the error $\mathcal{E}$ with respect to the weights $w^{\lambda}_{i|j}$ in two stages, using an intermediate quantity δ^{λ}_j:

$$\begin{cases} \delta^{\lambda}_j = \frac{\partial \mathcal{E}}{\partial \mathcal{B}^{\lambda}_j} \\ \frac{\partial \mathcal{E}}{\partial w^{\lambda}_{i|j}} = \delta^{\lambda}_j h^{\lambda-1}_j \end{cases} \tag{4.32}$$

To apply what we learned in the earlier General Theory section of this chapter, consider a NN with 4 layers (3 hidden, and the output visible one). Define the functions f_i as follows:

$$f^0_i = x_i \tag{4.33}$$

$$\text{Layer 0:} \quad f^1_i = \mathcal{B}^0_i(x_i), \quad f^2_i = \mathcal{A}^0_i(\mathcal{B}^0_i) \tag{4.34}$$

$$\text{Layer 1:} \quad f^3_i = \mathcal{B}^1_i(\mathcal{A}^0_i), \quad f^4_i = \mathcal{A}^1_i(\mathcal{B}^1_i) \tag{4.35}$$

$$\text{Layer 2:} \quad f^5_i = \mathcal{B}^2_i(\mathcal{A}^1_i), \quad f^6_i = \mathcal{A}^2_i(\mathcal{B}^2_i) \tag{4.36}$$

$$\text{Layer 3:} \quad f^7_i = \mathcal{B}^3_i(\mathcal{A}^2_i), \quad f^8_i = \mathcal{A}^3_i(\mathcal{B}^3_i) \tag{4.37}$$

See Fig.4.3. The TPMs, printed in blue, for the nodes of the bnet (c) for back propagation, are:

$$P(\frac{\partial Y}{\partial \mathcal{B}^\lambda} \mid \frac{\partial Y}{\partial \mathcal{B}^{\lambda+1}}) = \mathbb{1}(\frac{\partial Y}{\partial \mathcal{B}^\lambda} = \frac{\partial Y}{\partial \mathcal{B}^{\lambda+1}} \frac{\partial \mathcal{B}^{\lambda+1}}{\partial \mathcal{A}^\lambda} \frac{\partial \mathcal{A}^\lambda}{\partial \mathcal{B}^\lambda}) \ . \tag{4.38}$$

One has

$$\frac{\partial \mathcal{A}_i^\lambda}{\partial \mathcal{B}_j^\lambda} = D\mathcal{A}_i{}^\lambda(\mathcal{B}_i^\lambda)\delta(i,j) \tag{4.39}$$

where $D\mathcal{A}_i^\lambda(z)$ is the derivative of $\mathcal{A}_i^\lambda(z)$.

From Eq.(4.27)

$$\mathcal{B}_i^{\lambda+1}(\mathcal{A}^\lambda) = \sum_j w_{i|j}^{\lambda+1} \mathcal{A}_j^\lambda \tag{4.40}$$

so

$$\frac{\partial \mathcal{B}_i^{\lambda+1}}{\partial \mathcal{A}_j^\lambda} = w_{i|j}^{\lambda+1} \ . \tag{4.41}$$

Therefore, Eq.(4.38) implies

$$P(\frac{\partial Y}{\partial \mathcal{B}_j^\lambda} \mid \frac{\partial Y}{\partial \mathcal{B}_j^{\lambda+1}}) = \mathbb{1}(\frac{\partial Y}{\partial \mathcal{B}_j^\lambda} = \sum_i \frac{\partial Y}{\partial \mathcal{B}_i^{\lambda+1}} D\mathcal{A}_j^\lambda(\mathcal{B}_j^\lambda) w_{i|j}^{\lambda+1}) \ , \tag{4.42}$$

$$P(\frac{\partial \mathcal{E}}{\partial \mathcal{B}_j^\lambda} \mid \frac{\partial \mathcal{E}}{\partial \mathcal{B}_j^{\lambda+1}}) = \mathbb{1}(\frac{\partial \mathcal{E}}{\partial \mathcal{B}_j^\lambda} = \sum_i \frac{\partial \mathcal{E}}{\partial \mathcal{B}_i^{\lambda+1}} D\mathcal{A}_j^\lambda(\mathcal{B}_j^\lambda) w_{i|j}^{\lambda+1}) \ , \tag{4.43}$$

$$\boxed{P(\delta_j^\lambda \mid \delta_j^{\lambda+1}) = \mathbb{1}(\delta_j^\lambda = \sum_i \delta_i^{\lambda+1} D\mathcal{A}_j^\lambda(\mathcal{B}_j^\lambda)) w_{i|j}^{\lambda+1})} \ . \tag{4.44}$$

First delta of iteration, belonging to output layer $\lambda = \Lambda - 1$:

$$\begin{aligned}
\delta_j^{\Lambda-1} &= \frac{\partial \mathcal{E}}{\partial \mathcal{B}_j^{\Lambda-1}} && (4.45)\\
&= \frac{1}{nsam} \sum_{\sigma=0}^{nsam-1} \sum_{i=0}^{ny-1} 2(Y_i - y_i[\sigma]) D\mathcal{A}_i^{\Lambda-1}(\mathcal{B}_i^{\Lambda-1})\delta(i,j) && (4.46)\\
&= \frac{1}{nsam} \sum_{\sigma=0}^{nsam-1} 2(Y_j - y_j[\sigma]) D\mathcal{A}_j^{\Lambda-1}(\mathcal{B}_j^{\Lambda-1}) && (4.47)
\end{aligned}$$

Cute expression for derivative of sigmoid function:

$$D\text{smoid}(x) = \text{smoid}(x)(1 - \text{smoid}(x)) \tag{4.48}$$

4.3 General bnets instead of Markov chains induced by layered structure of NNs

$$P(\delta_{\underline{x}} \mid (\delta_{\underline{a}})_{\underline{a} \in ch(\underline{x})}) = \mathbb{1}(\delta_{\underline{x}} = \sum_{\underline{a} \in ch(\underline{x})} \delta_{\underline{a}} D\mathcal{A}_{\underline{x}}(\mathcal{B}_{\underline{x}})) w_{\underline{a}|\underline{x}}) \tag{4.49}$$

Reverse arrows of original bnet and define the TPM of nodes of "time reversed" bnet by

$$\boxed{P(\delta_{\underline{x}} \mid (\delta_{\underline{a}})_{\underline{a} \in pa(\underline{x})}) = \mathbb{1}(\delta_{\underline{x}} = \sum_{\underline{a} \in pa(\underline{x})} \delta_{\underline{a}} D\mathcal{A}_{\underline{x}}(\mathcal{B}_{\underline{x}})) w^T_{\underline{x}|\underline{a}})} \tag{4.50}$$

Chapter 5

Basic Curve Fitting Using Gradient Descent

$$
\begin{tikzcd}
 & \underline{\vec{x}} \arrow[r] \arrow[d] & \underline{\vec{y}} \arrow[d] & \\
\phi \arrow[r] \arrow[rrr, bend right] & \underline{\hat{\vec{y}}} \arrow[r] & \mathcal{E} \arrow[r] & \phi'
\end{tikzcd}
$$

Figure 5.1: Basic curve fitting bnet.

Samples $(x[\sigma], y[\sigma]) \in S_{\underline{x}} \times S_{\underline{y}}$ are given. $nsam(\vec{x}) = nsam(\vec{y})$.
Estimator function $\hat{y}(x;\phi)$ for $x \in S_{\underline{x}}$ and $\phi \in \mathbb{R}$ is given.
Let

$$P_{\underline{x},\underline{y}}(x,y) = \frac{1}{nsam(\vec{x})} \sum_{\sigma} \mathbb{1}(x = x[\sigma], y = y[\sigma]) \ . \tag{5.1}$$

Let

$$\mathcal{E}(\vec{x}, \vec{y}, \phi) = \frac{1}{nsam(\vec{y})} \sum_{\sigma} |y[\sigma] - \hat{y}(x[\sigma];\phi)|^2 \tag{5.2}$$

$\mathcal{E}$ is called the mean square error.

Best fit is parameters ϕ^* such that

$$\phi^* = \operatorname*{argmin}_{\phi} \mathcal{E}(\vec{x}, \vec{y}, \phi) \ . \tag{5.3}$$

The node TPMs for the basic curve fitting bnet Fig.5.1 are printed below in blue.

$$P(\phi) = \text{given} \ . \tag{5.4}$$

The first time it is used, ϕ is arbitrary. After the first time, it is determined by previous stage.

$$P(\vec{x}) = \prod_{\sigma} P_{\underline{x}}(x[\sigma]) \tag{5.5}$$

$$P(\vec{y}|\vec{x}) = \prod_{\sigma} P_{\underline{y}|\underline{x}}(y[\sigma] \mid x[\sigma]) \tag{5.6}$$

$$P(\hat{y}[\sigma]|\phi, \vec{x}) = \delta(\hat{y}[\sigma], \hat{y}(x[\sigma]; \phi)) \tag{5.7}$$

$$P(\mathcal{E}|\vec{\hat{y}}, \vec{y}) = \delta(\mathcal{E}, \frac{1}{nsam(\vec{x})} \sum_{\sigma} |y[\sigma] - \hat{y}[\sigma]|^2) \ . \tag{5.8}$$

$$P(\phi'|\phi, \mathcal{E}) = \delta(\phi', \phi - \eta \partial_{\phi} \mathcal{E}) \tag{5.9}$$

$\eta > 0$ is the descent rate. If $\Delta\phi = \phi' - \phi = -\eta \frac{\partial \mathcal{E}}{\partial \phi}$, then $\Delta\mathcal{E} = \frac{-1}{\eta}(\Delta\phi)^2 < 0$ so this will minimize the error $\mathcal{E}$. This is called "gradient descent".

Chapter 6

Bell and Clauser-Horne Inequalities in Quantum Mechanics

Figure 6.1: bnet used to discuss Bell and Clauser-Horne inequalities in Quantum Mechanics.

I wrote an article about this in 2008 for my blog "Quantum Bayesian Networks". See Ref.[54].

Chapter 7

Berkson's Paradox

For more information about Berkson's Paradox (BP), see Ref.[65]

Figure 7.1: Bnet used to discuss Berkson's Paradox (BP). $\underline{a}$ and $\underline{b}$ are common causes of collider $\underline{x}$.

Consider the bnet of Fig.7.1. For that bnet, we have

$$P(a, b, x) = P(a)P(b)P(x|a, b) \ . \tag{7.1}$$

Summing Eq.(7.1) over x, we get

$$P(a, b) = P(a)P(b) \tag{7.2}$$

so $\underline{a}$ and $\underline{b}$ are independent. It follows that a can be ignored in calculating the probability of b; i.e.,

$$\boxed{P(b|a) = P(b)} \ . \tag{7.3}$$

However, a cannot be ignored in calculating the probability of b, if x is being held fixed; i.e.,

$$\boxed{P(b|a, x) \neq P(b|x)} \ . \tag{7.4}$$

Indeed,

$$P(b|a, x) = \frac{P(b)P(x|a, b)}{\sum_b P(b)P(x|a, b)} \tag{7.5}$$

whereas

$$P(b|x) = \frac{\sum_a P(a)P(b)P(x|a,b)}{\sum_{a,b} P(a)P(b)P(x|a,b)} \ . \tag{7.6}$$

The two boxed equations are what is referred to as BP.

BP is also called **collider bias** because $\underline{x}$ is a collider.

BP is also called **explaining away** in the special case that $\underline{a}, \underline{b}, \underline{x} \in \{true, false\} = \{0, 1\}$. In that case, if $\underline{x}$ is fixed to true, and the cause $\underline{a}$ is known to be true, then the cause $\underline{b}$ is less likely to be true. For example, suppose a car engine fails ($\underline{x} = 1$) and the two most likely causes of the failure are alternator ($\underline{a}$) and battery ($\underline{b}$). Once we know that the alternator has failed ($\underline{a} = 1$), it is less likely that the battery is failing ($\underline{b} = 1$) than when the status of $\underline{a}$ was not known; i.e., $P(b = 1|x = 1, a = 1) < P(b = 1|x = 1)$.

Figure 7.2: Example of Berkson's paradox (BP).

Fig.7.2 presents an example of BP. The figure consists of a scatter plot with axes x=intelligence, y=attractiveness, for a population of possible dates for a single person. For the full population,

$$(a, b) \sim P(a, b) = P(a)P(b) \tag{7.7}$$

whereas for the population in the white swath,

$$(a, b) \sim P(a, b|x) = P(b|a, x)P(a|x) \neq P(b|x)P(a|x) \ . \tag{7.8}$$

As shown by Fig.7.2, BP is an example of **selection bias**. Selection bias happens when a non-representative subset of the total population is considered (i.e., selected).

Chapter 8

Binary Decision Diagrams

x1	x2	x3	f
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Figure 8.1: Binary decision tree and truth table for the function $f(x_1, x_2, x_3) = \bar{x}_1(x_2 + \bar{x}_3) + x_1 x_2$

Figure 8.2: BDD for the function f of Fig.8.1.

This chapter is based on Wikipedia article Ref.[69].

Binary Decision Diagrams (BDDs) can be understood as a special case of Decision Trees (dtrees). We will assume that the reader has read Chapter 12 on dtrees before reading this chapter.

Both Figs.8.1 and 8.2 were taken from the aforementioned Wikipedia article. They give a simple example of a function $f : \{0,1\}^3 \to \{0,1\}$ represented in Fig.8.1 as a **binary decision tree** and in Fig.8.2 as a **binary decision diagram (BDD)**. It is possible to find, for each of those two figures, a bnet with the same graph structure. We show how to do this next.

We begin by noting that the function $f : \{0,1\}^3 \to \{0,1\}$ is a special case of a probability distribution $P : \{0,1\}^3 \to [0,1]$. In fact, if we restrict P to be deterministic, then $P_{det} : \{0,1\}^3 \to \{0,1\}$ has the same domain and range as f. Henceforth, we will refer to $f(x_1, x_2, x_3)$ as $P(x_1, x_2, x_3)$, keeping in mind that we are restricting our attention to deterministic probability distributions.

If we apply the chain rule for conditional probabilities to $P(x_1, x_2, x_3)$, we get

$$P(x_1, x_2, x_3) = P(x_3|x_1, x_2)P(x_2|x_1)P(x_1) \ , \tag{8.1}$$

which can be represented by the bnet:

Figure 8.3: Most general 3 node bnet.

But in Chapter 12, we learned how to represent the dtree of Fig.8.1 as the image bnet Fig.8.4. The node TPMs, printed in blue, for the image bnet Fig.8.4 are as follows. Note that the TPMs for Fig.8.4 can be constructed from the TPMs for the bnet Fig.8.3. If $x_1, x_2, x_3, x_4 \in \{0, 1, null\}$ and $a, b, c \in \{0, 1\}$, then

$$P(\underline{x_1} = x_1) = \begin{cases} P_{\underline{x}_1}(x_1) & \text{if } x_1 \in \{0,1\} \\ 0 & \text{if } x_1 = null \end{cases} \tag{8.2}$$

$$P(\underline{x_2|a} = x_2 \mid \underline{x_1} = x_1) = \begin{cases} P_{\underline{x}_2|\underline{x}_1}(x_2|a) & \text{if } x_1 = a \\ \mathbb{1}(x_2 = null) & \text{otherwise} \end{cases} \tag{8.3}$$

$$P(\underline{x_3|a, b} = x_3 \mid \underline{x_2|a} = x_2) = \begin{cases} P_{\underline{x}_3|\underline{x}_1,\underline{x}_2}(x_3|a, b) & \text{if } x_2 = b \\ \mathbb{1}(x_3 = null) & \text{otherwise} \end{cases} \tag{8.4}$$

Figure 8.4: Image bnet for binary dtree of Fig.8.1.

$$P(\underline{x_4|a,b,c} = x_4 \mid \underline{x_3|a,b} = x_3) = \begin{cases} \delta(x_4, c) & \text{if } x_3 = c \\ \mathbb{1}(x_4 = null) & \text{otherwise} \end{cases} \tag{8.5}$$

Note that if $P_{\underline{x}_3|\underline{x}_1,\underline{x}_2} = P_{\underline{x}_3|\underline{x}_2}$ in Eq.(8.4), then the bnet Fig.8.3 reduces to a Markov chain $\underline{x}_1 \to \underline{x}_2 \to \underline{x}_3$.

The BDD shown in Fig.8.2 emphasizes the fact that

$$P(x_1, x_2, x_3|x_1 = 1) = P(x_2|x_1 = 1) = x_2 \ . \tag{8.6}$$

The BDD of Fig.8.2 has as image bnet Fig.8.5. Define

$$pa(\underline{0}) = pa(\underline{1}) = (x_2|1, x_3|00, x_3|01) \ . \tag{8.7}$$

Let $pa(\underline{0}) = abc$ mean the same as $pa(\underline{0}) = (a, b, c)$. The TPMs of the image bnet Fig.8.5 are the same as those for the image bnet Fig.8.4 except for the TPMs of the nodes $\underline{0}$ and $\underline{1}$. For those two nodes, the TPMs, printed in blue, are as follows.

$$P(\underline{0} = x|pa(\underline{0})) = \begin{cases} \delta(x, 0) & \text{if } pa(\underline{0}) = 011 \\ \delta(x, null) & \text{otherwise} \end{cases} \tag{8.8}$$

$$P(\underline{1} = x|pa(\underline{1})) = \begin{cases} \delta(x, 1) & \text{if } pa(\underline{1}) = 101 \\ \delta(x, null) & \text{otherwise} \end{cases} \tag{8.9}$$

Figure 8.5: Image bnet for BDD of Fig.8.2.

Chapter 9

Chow-Liu Trees and Tree Augmented Naive Bayes (TAN)

This chapter is mostly based on chapter 8 of Pearl's 1988 book Ref.[35]. See also Ref.[74] and references therein.

This chapter uses various Shannon Information Theory entropies. Our notation for these entropies is described in Chapter Notational Conventions and Preliminaries.

9.1 Chow-Liu Trees

Chow-Liu trees refers to an algorithm for finding a bnet tree that fits an a priori given probability distribution as closely as possible.

Consider a bnet with n nodes $\underline{x}^n = (\underline{x}_0, \underline{x}_1, \ldots, \underline{x}_{n-1})$ such that $\underline{x}_i \in S_{\underline{x}_i}$ for all i. Let its total probability distribution be $P_{\underline{x}^n}$. For simplicity, we will abbreviate $P_{\underline{x}^n}$ by P. Hence

$$P(x^n) = P_{\underline{x}^n}(x^n) \ . \tag{9.1}$$

Suppose we want to fit $P_{\underline{x}^n}$ by a tree bnet with nodes $\underline{t}^n = (\underline{t}_0, \underline{t}_1, \ldots, \underline{t}_{n-1})$ such that $\underline{t}_i \in S_{\underline{t}_i} = S_{\underline{x}_i}$ for all i. For simplicity, we will abbreviate $P_{\underline{t}^n}$ by P_T. Hence

$$P_T(x^n) = P_{\underline{t}^n}(x^n) \ . \tag{9.2}$$

Throughout this chapter, let $V = \{0, 1, \ldots, n-1\}$, the set of vertices. Suppose μ is a function $\mu : V \to V$ such that $\mu(i) < i$. Let $T_\mu = \{\underline{t}_{\mu(i)} \to \underline{t}_i : i \in V - \{0\}\}$. Then T_μ is a tree that spans (i.e., it includes all nodes) $\underline{t}^n$. Its root node is $\underline{t}_0$, because $\underline{t}_0$ has no parents. All other nodes $\underline{t}_i$ have exactly one parent, namely $\underline{t}_{\mu(i)}$. Let P_T, the total probability distribution for the tree, be parameterized by the function μ as

follows:

$$P_T(x^n) = \prod_{i=0}^{n-1} P_T(x_i|x_{\mu(i)}) \ , \tag{9.3}$$

where, for the root node 0, $P_T(x_0|x_{\mu(0)}) = P_T(x_0)$.

Claim 14 *$D_{KL}(P \parallel P_T)$ is minimized over all probability distributions P_T that are expressible as Eq.(9.3) iff*

$$P_T(x_i|x_{\mu(i)}) = P(x_i|x_{\mu(i)}) \tag{9.4}$$

for all i, and

$$\sum_i H(\underline{x}_i : \underline{x}_{\mu(i)}) \tag{9.5}$$

is maximized over all μ.

proof:

$$\begin{aligned}
D_{KL}(P \parallel P_T) &= \sum_{x^n} P(x^n) \ln \frac{P(x^n)}{P_T(x^n)} && (9.6)\\
&= -\sum_{x^n} \sum_i P(x^n) \ln P_T(x_i|x_{\mu(i)}) + \sum_i P(x^n) \ln P(x^n) && (9.7)\\
&= -\sum_i \sum_{x_i, x_{\mu(i)}} P(x_i, x_{\mu(i)}) \ln P_T(x_i|x_{\mu(i)}) - H(\underline{x}^n) && (9.8)\\
&= -\sum_i \sum_{x_{\mu(i)}} P(x_{\mu(i)}) \left[\sum_{x_i} P(x_i|x_{\mu(i)}) \ln P_T(x_i|x_{\mu(i)}) \right] - H(\underline{x}^n) \ . && (9.9)
\end{aligned}$$

Now note that

$$\sum_{x_i} P(x_i|x_{\mu(i)}) \ln \frac{P(x_i|x_{\mu(i)})}{P_T(x_i|x_{\mu(i)})} \geq 0 \tag{9.10}$$

and this inequality becomes an equality iff

$$P(x_i|x_{\mu(i)}) = P_T(x_i|x_{\mu(i)}) \ . \tag{9.11}$$

Therefore

$$D_{KL}(P \parallel P_T) \geq -\sum_i \underbrace{\sum_{x_{\mu(i)}} P(x_{\mu(i)}) \left[\sum_{x_i} P(x_i|x_{\mu(i)}) \ln P(x_i|x_{\mu(i)}) \right]}_{=H(\underline{x}_i|\underline{x}_{\mu(i)})=H(\underline{x}_i:\underline{x}_{\mu(i)})-H(\underline{x}_i)} - H(\underline{x}^n) \ , \tag{9.12}$$

and this inequality becomes an equality iff Eq.(9.11) is satisfied.

Note from the last equation that

$$\operatorname*{argmin}_{\mu} D_{KL}(P \parallel P_T) = \operatorname*{argmax}_{\mu} \sum_i H(\underline{x}_i : \underline{x}_{\mu(i)}) \ . \tag{9.13}$$

QED

Claim 15

$$\operatorname*{argmin}_{\mu} H(\underline{x}^n) = \operatorname*{argmax}_{\mu} \sum_i H(\underline{x}_i : \underline{x}_{\mu(i)}) \tag{9.14}$$

proof:

$$\begin{aligned}
H(\underline{x}^n) &= -\sum_{x^n} P(x^n) \sum_i \ln P(x_i|x_{\mu(i)}) && (9.15)\\
&= -\sum_i \sum_{x_i, x_{\mu(i)}} P(x_i, x_{\mu(i)}) \ln P(x_i|x_{\mu(i)}) && (9.16)\\
&= -\sum_i \sum_{x_i, x_{\mu(i)}} P(x_i, x_{\mu(i)}) \left[\ln \frac{P(x_i|x_{\mu(i)})}{P(x_i)} + \ln P(x_i) \right] && (9.17)\\
&= -\sum_i \left[H(\underline{x}_i : \underline{x}_{\mu(i)}) - H(\underline{x}_i) \right] && (9.18)\\
&= \sum_i H(\underline{x}_i) - \sum_i H(\underline{x}_i : \underline{x}_{\mu(i)}) && (9.19)
\end{aligned}$$

QED

The meaning of Claims 14 and 15 is as follows. If $D_{KL}(P \parallel P_T)$ is minimized over all P_T, then

1. P_T inherits its TPMs from P, and
2. P_T gets its structure, which is being parameterized by the function μ, by maximizing the score given by

$$\text{score} = \sum_i H(\underline{x}_i : \underline{x}_{\mu(i)}) \ . \tag{9.20}$$

(mutual information $H(\underline{a} : \underline{b})$ measures correlation between $\underline{a}$ and $\underline{b}$). Maximizing the score is the same as minimizing the entropy $H(\underline{x}^n)$ over all the structures μ. (i.e., finding least complex structure).

So far, we have studied the properties of those probability distributions P_T for a tree bnet that best approximates an a priori given probability distribution P, but we haven't yet described how to build a Chow-Liu tree based on empirical data. Next we give Chow-Liu's algorithm for doing so.

1. **Find MST using Kruskal's algorithm[1]. (see Fig.9.1)**
   Calculate weights $w_{i,j} = H(\underline{x}_i : \underline{x}_j)$ for all $i, j \in V$ and store them in a dictionary D that maps edges to weights.
   Order D by weight size.
   Let T be a list of the edges in the tree. Initialize T to empty.
   Repeat this until T has $n-1$ elements:
   Remove largest weight w from D and corresponding edge e.
   Add e to T if $\{e\} \cup T$ has no loops. Otherwise discard e and w.

2. **Give directions to edges in T. (see Fig.9.2)**
   Let DT be a list of directed edges. Initialize DT to empty.
   Choose any node as root node.
   Point arrows along edges in T, away from root node.
   Add new arrows to DT.
   Repeat this until DT has $n-1$ elements:
   Point arrows along edges in T, away from leaf nodes of current DT.
   Add new arrows to DT.

Figure 9.1: Example of finding MST (maximum spanning tree)

Figure 9.2: Example of giving directions to edges of spanning tree.

Nodes in a Chow-Liu tree can be rated in terms of their relative importance. Here are 2 possible metrics for measuring the importance of a node $\underline{a}$:

$$N_{nb}(\underline{a}) = \text{ number of neighbors of } \underline{a} \tag{9.21}$$

[1]Kruskal's algorithm is one several famous algorithms (Prim's algo is another one) for finding an MST (maximum or minimum spanning tree). An MST algorithm takes an undirected graph with weights along its edges as input. It then finds a tree subgraph (i.e., subset of the edges of the graph with no loops) that (1) spans the graph (i.e., includes every vertex of the graph) and (2) maximizes (or minimizes) the sum of weights among all possible tree subgraphs. For more information, see Ref[105] and references therein, or any other of numerous explanations of MST in the Internet.

$$\text{traffic}(\underline{a}) = \sum_{\underline{n} \in nb(\underline{a})} H(\underline{a} : \underline{n}) \tag{9.22}$$

For example, to get a tree with low depth, one can choose as the root node the node which has largest N_{nb}, and if there are several with the same largest N_{nb}, choose out of those the one with the largest traffic.

9.2 Tree Augmented Naive Bayes (TAN)

Recall from Chapter 47 that a Naive Bayes bnet consists of a class node $\underline{c}$ with n children nodes $\underline{x}^n$, called the feature nodes. A Tree Augmented Naive Bayes (TAN) bnet is a Naive Bayes bnet with a tree grafted onto it like a chimera. More precisely, one starts with a Naive Bayes bnet and adds arrows between the feature nodes. The arrows are added in such a way that the TAN bnet sans node $\underline{c}$ constitutes a tree. It's not the most well motivated bnet in human history, but at least it adds a bit of correlation between the feature nodes of the Naive Bayes bnet. Those nodes are independent at fixed $\underline{c}$ in the Naive Bayes bnet, but are no longer so in the TAN bnet. See Figs.9.3 and 9.4 for an example of a TAN bnet.

Figure 9.3: bnet for Naive Bayes with 4 feature nodes and another bnet for a tree made of the same feature nodes.

Figure 9.4: TAN bnet constructed by merging Naive Bayes bnet and tree bnet of Fig.9.3.

The total probability distribution P_{TAN} for a TAN bnet can be parameterized as follows.

$$P_{TAN}(x^n, c) = P_{TAN}(c) \prod_{i=0}^{n-1} P_{TAN}(x_i | x_{\mu(i)}, c) \ . \tag{9.23}$$

As with Chow Liu trees, we can attempt to find a TAN bnet whose total probability $P_{TAN} = P_{\underline{t}^n,\underline{c}}$ best approximates an a priori given probability distribution $P = P_{\underline{x}^n,\underline{c}}$.

Note that

Claim 16

$$\underset{\mu}{\operatorname{argmin}}\, H(\underline{x}^n, \underline{c}) = \underset{\mu}{\operatorname{argmax}} \sum_i H(\underline{x}_i : \underline{x}_{\mu(i)} | \underline{c}) \tag{9.24}$$

proof:

$$H(\underline{x}^n, \underline{c}) = -\sum_{x^n,c} P(x^n, c) \left[\ln P(c) + \sum_i \ln P(x_i | x_{\mu(i)}, c) \right] \tag{9.25}$$

$$= -\sum_{x^n,c} P(x^n, c) \left[\ln P(c) + \sum_i \ln \left(\frac{P(x_i, x_{\mu(i)}|c)}{P(x_i|c)P(x_{\mu(x_i)}|c)} P(x_i|c) \right) \right] \tag{9.26}$$

$$= \sum_i H(\underline{x}_i, \underline{c}) - \sum_i H(\underline{x}_i : \underline{x}_{\mu(i)} | \underline{c}) \tag{9.27}$$

QED

Following the same line of reasoning that we followed for Chow-Liu trees, we conclude that:

If $D_{KL}(P \parallel P_{TAN})$ is minimized over all P_{TAN}, then

1. P_{TAN} inherits its TPMs from P, and
2. P_{TAN} gets its structure, which is being parameterized by the function μ, by maximizing the score defined by

$$\text{score} = \sum_i H(\underline{x}_i : \underline{x}_{\mu(i)} | \underline{c}) \tag{9.28}$$

One can build a TAN bnet from empirical data as follows:

Calculate a Chow-Liu Tree for each $c \in S_{\underline{c}}$. For each of those trees, create a TAN bnet, and calculate its score given by Eq.(9.28). Keep the TAN bnet with the largest score.

Chapter 10

Counterfactual Reasoning

10.1 The 3 Rungs of Causal AI

According to Judea Pearl, there are 3 rungs in the ladder of causal AI. These are (as I see them):

1. **Observing Passively:** Collecting data and fitting curves to it, without any plan designed to investigate Nature's causal connections.

2. **Doing causal experiments:** Doing experiments consciously designed to elucidate Nature's causal connections. Even cats do this!, but current AI doesn't.

3. **Imagining counterfactual situations, Analogizing:** Imagining gedanken experiments to further understand Nature's causal connections, and to decide what future courses of action are more likely to succeed, even if those courses of action are unprecedented, and have never been taken before. Making predictions about events that have never happened ("counterfactuals") is a very Bayesian concern, well out of the purview of frequentists. Nevertheless, humans do such "analogizing" all the time to great advantage. It becomes possible if there is some indirect but similar data that can be transported (transplanted, applied) to the situation of interest.

Chapter 41 on message passing is about rung 1. Chapter 15 on Do Calculus is about rung 2. This chapter is dedicated to rung 3.

Judea Pearl is fond of discussing rung 3 solely in terms of SCM.[1] In this chapter, we define rung 3 without using SCM, using solely bnets. This gives a more general version of rung 3, because SCM are a subset of bnets.

We will use the term **intervention operator (or simply "intervention")** to refer to an operator that maps a bnet to another bnet. In Chapter 15, we introduced an intervention operator called the **do operator** $\mathcal{D}_{\underline{x}=x}$ (this is our notation for

[1] SCM are what we call DEN. DEN (deterministic systems with external noise) are discussed in Chapter 36.

what Pearl symbolizes by $do(\underline{x}) = x$). The study of counterfactuals requires that we introduce a new kind of intervention operator that we will call an **imagine operator**, and denote by $\mathcal{I}_{\underline{x}\to\underline{y}}(\tilde{x})$. These 2 types of interventions will be defined next.

10.2 Do operator

Figure 10.1: Action of "do" operator $\mathcal{D}_{\underline{x}=5}$ on node $\underline{x}$.

The do operator $\mathcal{D}_{\underline{x}=5}$ is defined graphically in Fig.10.1. The TPM, printed in blue, for node $\tilde{\underline{x}}$ of Fig.10.1, is given by

$$P(\tilde{x}) = \delta(5, \tilde{x}) \tag{10.1}$$

The do operator $\mathcal{D}_{\underline{x}=5}$ amputates the incoming arrows of node $\underline{x}$ and sets the TPM of the new root node $\tilde{\underline{x}}$ to a delta function $\delta(\tilde{x}, 5)$ (or some state of $\underline{x}$ other than 5). Sometimes we call the new node $\mathcal{D}\underline{x}$ instead of $\tilde{\underline{x}}$.

The uses of the do operator are discussed in detail in Chapter 15.

10.3 Imagine operator

The imagine operator $\mathcal{I}_{\underline{x}\to\underline{y}}(5)$ is defined graphically in Fig.10.2. Note that Fig.10.2 actually defines two types of imagine operators, the general one (with an argument) $\mathcal{I}_{\underline{x}\to\underline{y}}(5)$ and its special case (without an argument) $\mathcal{I}_{\underline{x}\to\underline{y}}$. The TPMs, printed in blue, for various nodes in Fig.10.2, are as follows.

- For $\mathcal{I}_{\underline{x}\to\underline{y}}(\tilde{x})G$ with node $\underline{y}^{nx}$

$$P(y^{nx}|a.) = \prod_{\tilde{x}} P(\underline{y}(\tilde{x}) = y(\tilde{x})|a.) \tag{10.2}$$

$$P(y|y^{nx}, \tilde{x}) = \delta(y, y(\tilde{x})) \tag{10.3}$$

Figure 10.2: Action of "imagine" operators $\mathcal{I}_{\underline{x}\to\underline{y}}(5)$ and $\mathcal{I}_{\underline{x}\to\underline{y}}$ on arrow $\underline{x} \to \underline{y}$. In this figure, $y^{nx} = [y(x)]_{\forall x \in S_{\underline{x}}}$, where $nx = |S_{\underline{x}}|$ and $S_{\underline{x}}$ is the set of states of node $\underline{x}$.

- For $\mathcal{I}_{\underline{x}\to\underline{y}}(\tilde{x})G$ without node $\underline{y}^{nx}$

$$P(y|\tilde{x}, a.) = P(\underline{y}(\tilde{x}) = y|a.) \tag{10.4}$$

- For $\mathcal{I}_{\underline{x}\to\underline{y}}G$

$$P(y^{nx}|a.) = \prod_{\tilde{x}} P(\underline{y}(\tilde{x}) = y(\tilde{x})|a.) \tag{10.5}$$

$$P(y|y^{nx}, x) = \delta(y, y(x)) \tag{10.6}$$

The imagine operator $\mathcal{I}_{\underline{x}\to\underline{y}}(5)$ operates on an arrow whereas the $\mathcal{D}$ operator operates on a node. $\mathcal{I}_{\underline{x}\to\underline{y}}(5)$ deletes arrow $\underline{x} \to \underline{y}$ and creates a new root node $\tilde{\underline{x}}$ and a new arrow $\tilde{\underline{x}} \to \underline{y}$. It also creates a new node $\underline{y}^{nx}$ and an arrow $\underline{y}^{nx} \to \underline{y}$. Sometimes we call the new node $\mathcal{I}_{\underline{y}}\underline{x}$ instead of $\tilde{\underline{x}}$.

Fig.10.3 shows how the imagine operator arises in Potential Outcomes (PO) theory. PO theory is discussed extensively in Chapter 52. As you can see, PO theory only uses a limited version of rung 3, because it uses only the *special case* imagine operator $\mathcal{I}_{\underline{d}\to\underline{y}}$.

Fig.10.4 gives a connection between do and imagine operators. We see from

Figure 10.3: How imagine operator arises in Potential Outcomes (PO) theory.

Figure 10.4: $\mathcal{D}_{\underline{x}=5}\mathcal{I}_{\underline{x}\to\underline{y}}G$ gives a connection between do and imagine operators.

that figure that for $\mathcal{D}_{\underline{x}=\tilde{x}}\mathcal{I}_{\underline{x}\to\underline{y}}G$, we have[2]

$$P(y|\mathcal{D}\underline{x} = \tilde{x}, a.) = P(\underline{y}(\tilde{x}) = y|a.) \tag{10.7}$$

One can define a **do-imagine-calculus** whose objective is to express probabilities such as $P(\underline{y}|\mathcal{D}\underline{r} = r, \mathcal{I}_{\underline{b}}\underline{s} = s, t)$ in terms of observable probabilities that do not contain any do or imagine operators in them. As with Do Calculus, this reduction is not always possible, and we say a probability is **$\mathcal{D}$-identifiable**, **$\mathcal{I}$-identifiable** or **$\mathcal{DI}$-identifiable** if it can be expressed without do, imagine or both operators.

[2]In the notation favored by Pearl, Eq.(10.7) would be

$$P(y|do(X) = \tilde{x}, a.) = P(Y_{\tilde{x}} = y|a.)$$

Chapter 11

Cross-Validation

This chapter is based on Ref.[77].

Cross-Validation (CV) is a method of calculating the **"out-of-training-set" (OOTS) error** for a classifier. What this means is that the classifier is trained on a training set, and its propensity to err is evaluated on a set different from the training set.

In k**-fold CV**, the most common CV method, and the only one we will discuss in this chapter, one partitions a dataset into k disjoint datasets of equal length. One uses $k-1$ of those sub-datasets to train a model, and saves the last sub-dataset to validate the model just trained. One actually rotates which of the k sub-datasets is used for validation purposes, and calculates k validation errors $\mathcal{E}_j$ for $j = 0, 1, \ldots, k-1$. Then one averages over the $\mathcal{E}_j$ to obtain a final OOTS error $\mathcal{E}$.

CV strongly resembles Jackknife Resampling (JR) (see Chapter 32), but in JR the validation sub-dataset is never used for anything, whereas in CV, it is used for validation purposes, to calculate an OOTS error.

Next, we will explain k-fold CV more explicitly, using equations and a bnet.

Let $L = [0, 1, 2, \ldots, nsam-1]$ be a list of individuals (samples) in a population. In this chapter, we will use the notation $A^\sigma = A[\sigma]$ and $\vec{A} = [A^\sigma : \sigma \in L]$ for a list (vector, 1-D array) indexed by L. We will refer to $DS = (\vec{x}, \vec{y})$ where $x^\sigma \in S_{\underline{x}}$, $y^\sigma \in S_{\underline{y}}$, as a dataset. If L_j is a list (possibly with duplicate items) such that $set(L_j) \subset set(L)$, then define $DS_j = (\vec{x}, \vec{y})_{L_j} = ((x^\sigma)_{\sigma \in L_j}, (y^\sigma)_{\sigma \in L_j})$. We will refer to DS_j as the **restriction of** $(\vec{x}, \vec{y})$ **to** L_j**.**

Let $J = \{0, 1, 2, \ldots, nj-1\}$.

Define a **training list(TL), validation list(VL) pair** (TL, VL) to be a pair of lists such that $set(TL)$ and $set(VL)$ are disjoint subsets of $set(L)$. Let (TL_j, VL_j) for $j \in J$ be nj such TL-VL pairs.

Fig.11.1 shows the TL-VL pairs that are used when doing k-fold CV. In that figure, $k = nj = 4$. As you can see, in k-fold CV, one chooses $nj = k$ list pairs (TL_j, VL_j) such that all individuals $\sigma \in L$ appear exactly once, in either TL_j or TV_j, but not in both.

We will refer to a function $Y : S_{\underline{x}} \to S_{\underline{c}}$ as a classifier. It maps a vector of

Figure 11.1: 4-fold CV with $|L| = 12$. For all j, $|VL_j| = 3$ and $|TL_j| = 9$. All individuals $\sigma \in L$ appear exactly once, in either TL_j or TV_j, but not in both.

features x to a class c. Let Y_j for $j \in J$ denote nj classifiers.

If $Y_j : S_{\underline{x}} \to S_{\underline{c}}$ and $S_{\underline{c}}$ is a discrete set ("categorical"), then define the **OOTS error for the jth classifier** as:

$$\mathcal{E}_j = \frac{1}{|VL_j|} \sum_{\sigma \in VL_j} \mathbb{1}(y^\sigma \neq Y_j(x^\sigma)) \ . \tag{11.1a}$$

On the other hand, if $S_{\underline{c}} = \mathbb{R}$, it makes more sense to define $\mathcal{E}_j$ as a mean square error:

$$\mathcal{E}_j = \frac{1}{|VL_j|} \sum_{\sigma \in VL_j} (y^\sigma - Y_j(x^\sigma))^2 \ . \tag{11.1b}$$

Finally, define the **final OOTS** error as

$$\mathcal{E} = \frac{1}{|J|} \sum_{j \in J} \mathcal{E}_j \ . \tag{11.2}$$

Fig.11.2 gives a bnet that represents the CV algorithm. The TPMs, printed in blue, for the bnet of Fig.11.2, are as follows:

$$P((\vec{x}, \vec{y})_{TL_j} | (\vec{x}, \vec{y})) = \mathbb{1}(\quad (\vec{x}, \vec{y})_{TL_j} = \text{restriction of } (\vec{x}, \vec{y}) \text{ to } TL_j. \quad) \tag{11.3}$$

$$P((\vec{x}, \vec{y})_{VL_j} | (\vec{x}, \vec{y})) = \mathbb{1}(\quad (\vec{x}, \vec{y})_{VL_j} = \text{restriction of } (\vec{x}, \vec{y}) \text{ to } VL_j. \quad) \tag{11.4}$$

$$P(Y_j | (\vec{x}, \vec{y})_{TL_j}) = \mathbb{1}(\quad Y_j = \text{classifier trained with } (\vec{x}, \vec{y})_{TL_j} \quad) \tag{11.5}$$

Figure 11.2: Bnet for 3-fold CV.

$$P(\mathcal{E}_j | Y_j, (\vec{x}, \vec{y})_{VL_j}) = \mathbb{1}(\quad \mathcal{E}_j = \text{ defined by Eqs.(11.1).} \quad) \tag{11.6}$$

$$P(\mathcal{E} | (\mathcal{E}_j)_{j \in J}) = \mathbb{1}(\quad \mathcal{E} = \text{ defined by Eq.(11.2).} \quad) \tag{11.7}$$

Chapter 12

Decision Trees

This chapter is based mainly on Ref.[50].

Figure 12.1: Example of dtree taken from Ref.[50]

Fig.12.1 shows a typical decision tree (dtree). This example was taken from Ref.[50], where it is analyzed in detail. As you can see, a dtree contains two main types of nodes: the non-leaf nodes, and the leaf nodes. The **non-leaf nodes** pose **questions**. In general, the **answers**[1] to those questions can be multiple choices with two or more choices. For each of those choices, a tree branch labeled by the choice comes down from the question node. The **leaf nodes** represent endpoints, goals, final conclusions, etc. Dtrees can be viewed as classifiers. They take in a large amount of information about a population and compress that information to just a few classes. If $S_{\underline{c}}$ is the set of distinct leaf node labels, then we call each $c \in S_{\underline{c}}$ a **class of the classifier**. In the case of Fig.12.1, $S_{\underline{c}} = \{False, True\}$.

[1]The **question-answer pairs** in dtrees are also called **attribute-value pairs**. Attributes are also called **features**.

Figure 12.2: Fig.12.1 with abridged labels.

Figure 12.3: Fig.12.2 converted to a bnet.

Dtrees can be used with probabilities attached to each node, or without probabilities (as a plain undirected graph(UG)). This is analogous to bnets, which can be used with probabilities attached to each node (as DAGs with TPMs specified for each node) or without probabilities (as plain DAGs). Dtrees differ from bnets in that their tree branches are labelled, whereas bnet arrows aren't labelled. Also, whereas the nodes of a bnet carry a matrix of probabilities (the TPM), the nodes of a dtree carry just a column vector of probabilities which represents a single probability distribution. Henceforth, we will refer to the column vector of probabilities carried by each node of a dtree as its **Transition Probability Vector (TPV)**. Without the TPVs, a dtree can be used as a deterministic classifier, to classify inputs. With the TPVs, it can be used as a probabilistic sampler (to generate random samples.)

$P(x\|a)$	$a = a_0$	$a \in S^-_{\underline{a}} - \{a_0\}$	*null*
0	p_0	0	0
1	p_1	0	0
$\vdots$	$\vdots$	0	0
$N^-_{\underline{x}} - 1$	$p_{N_{\underline{x}}-1}$	0	0
null	0	1	1

Table 12.1: TPM of a node of a dtree image bnet.

12.1 Transforming a dtree into a bnet

A trivial observation that is seldom made in the dtree pedagogical literature is that every dtree maps into a special bnet, let's call it its "image" bnet, in a very natural way. We use the dtree of Fig.12.1 as an example to show how to do this. As a first step, we go from Fig.12.1 to Fig.12.2 by replacing all the labels of the nodes and of the branches of the dtree by abridged symbols. Next, we go from Fig.12.2 to Fig.12.3, by replacing all tree branches by arrows pointing down, and by moving the tree branch labels down so that they become a suffix to the question that the tree branch leads to. At this point, we have created Fig.12.3, which constitutes the DAG of the image bnet. It remains for us to define a TPM for each node of this DAG.

Table 12.1 gives the TPM $P(x|a)$ for a node $\underline{x}$ with single parent $\underline{a}$ of a dtree image bnet. Say node $\underline{x}$ has a set $S^-_{\underline{x}}$ of possible tree branches coming out of it. Let $N^-_{\underline{x}} = |S^-_{\underline{x}}|$. Let $S_{\underline{x}} = S^-_{\underline{x}} \cup \{null\}$ and $N_{\underline{x}} = |S_{\underline{x}}| = N^-_{\underline{x}} + 1$. Define $S^-_{\underline{a}}$, $N^-_{\underline{a}}$, $S_{\underline{a}}$ and $N_{\underline{a}}$ analogously for node $\underline{a}$. In Table 12.1, $S^-_{\underline{x}} = \{0, 1, \ldots, N^-_{\underline{x}} - 1\}$ and a_0 is the value of node $\underline{a}$ which labels the tree branch connecting nodes $\underline{a}$ and $\underline{x}$. $\vec{p} = (p_0, p_1, \ldots, p_{N_{\underline{x}}-1})$ is a probability distribution associated with node $\underline{x}$, its TPV. TPVs can be learned from a dataset following the dtree Structure Learning (SL) algorithm discussed in Section 12.2.

Table 12.1 also applies when node $\underline{x}$ is a leaf node, except that for leaf nodes, $\vec{p}$ is one hot (i.e., all components are zero except for one component which is 1). Also, all leaf nodes $\underline{x}$ have the same $S^-_{\underline{x}}$, namely $S_{\underline{c}}$.

Adding a null state to the set of states (SOS) of each node of the image bnet is necessary because, once *null* is added to the SOS of any node, it must be added to the SOS of all descendant nodes. *null* must be added to the SOS of the children of the root node to take care of the situations when those first children don't receive the state they were expecting from their parent, i.e., the root node.

When drawing dtrees, some people put info like explanations and probabilities on the branches of the dtree. That info can all be preserved in the TPM and the node names and node state names of the image bnet nodes. One can also place info inside tool tips attached to the node name and node state names. Often, the pedagogical literature states that dtrees are more explicit and carry more info than their image bnets, but if one follows the above prescriptions, both can carry the same info.

A naive Bayes (NB) bnet (see Chapter 47) consists of a single "class node" with states $S_{\underline{c}}$ that fans out with arrows pointing to the "feature nodes". If each leaf node of a NB bnet fans out into a set of new leaf nodes, and those new leaf nodes also fan out and so on, we get a generalized NB bnet. Let's call this type of tree bnet an NB^* bnet. An NB^* bnet has the same graph structure as the image bnet of a dtree, but it's more general, because its TPMs are more general. Each TPM of a NB^* bnet can have several non-trivial columns instead of just one TPV= $\vec{p}$.

12.2 Structure Learning for Dtrees

Let

$J_0 = \{0, 1, \ldots, nj - 1\}$

$\Sigma = \{0, 1, 2, \ldots, nsam - 1\}$

$DS = \{(\sigma, x^\sigma, c^\sigma) : \sigma \in \Sigma\}$ be a dataset

$\sigma \in \Sigma$ be an individual (a sample) from a population,

$x^\sigma \in S_{\underline{x}}$ be the **feature (attributes, questions) vector**. $S_{\underline{x}} = S_{\underline{x}_0} \times S_{\underline{x}_1} \times \ldots \times S_{\underline{x}_{nj-1}}$, $x = (x_0, x_1, \ldots, x_{nj-1}) \in S_{\underline{x}}$, $x_j \in S_{\underline{x}_j}$

$c^\sigma \in S_{\underline{c}}$ be a **classification class**

We will assume $S_{\underline{x}}$ and $S_{\underline{c}}$ are finite sets.

Building a **classifier** Y (curve fit) for a dtree means finding a deterministic function $Y : S_{\underline{x}} \to S_{\underline{c}}$ such that $c^\sigma \approx Y(x^\sigma)$ for all $\sigma \in \Sigma$. If we divide the population Σ into two large disjoint sets, a **training set** Σ_{train} and a **validation set** Σ_{vali}, and if $c^\sigma \approx Y(x^\sigma)$ very closely for $\sigma \in \Sigma_{train}$ but fits poorly for $\sigma \in \Sigma_{vali}$, then we say the classifier Y suffers from **overfitting**. We can learn the structure and TPVs of a dtree from a dataset DS, by using the dtree **Structure Learning (SL)** algorithm that we will discuss in detail later. However, that algorithm is prone to produce a classifier Y that overfits. Two techniques commonly used to reduce the effects of overfitting are **pruning** and **Random Forest (RF)** (see Chapter 54). Pruning just means somehow removing nodes that are too specific. An RF is an ensemble of dtrees that one averages over. In this chapter, we will only deal with a single dtree, not an ensemble of them.

Dtree SL was invented in 1984-1986 so it is fairly old. Many in the AI community consider dtrees old fashioned compared to neural nets. But dtrees are **interpretable** whereas neural nets aren't.[2] Bnets are interpretable too.

Below, we give the standard algorithm for SL of a dtree, in the form of pseudo-code. But first, we define two quantities, Information Gain and Gini, that are used in that pseudo-code.

[2] To be precise, only plain dtrees without boosting or bagging are interpretable. Dtrees used within boosting (see Chapter 1 on AdaBoost and Chapter 76 on XGBoost) or bagging (see Chapter 54 on Random Forest) gain much accuracy but lose **interpretability**.

12.2.1 Information Gain, Gini

This section uses various Shannon Information Theory entropies. Our notation for those entropies is described in Chapter Notational Conventions and Preliminaries.

Call a **separation ability measure** (SAM) a measure used to decide, when constructing a dtree from a dataset, in what order to ask the questions about the feature vector x. The question order is decided by searching over all so far unused questions for the question with the largest SAM.[3]

$$\{N_j(c, x_j)\}_{c \in S_{\underline{c}}, x_j \in S_{\underline{x}_j}}$$

$$\sum_{c \in S_{\underline{c}}} N_j(c, x_j) = N_j(x_j)$$

$$\sum_{x_j \in S_{\underline{x}_j}} N_j(c, x_j) = N_j(c) \qquad \sum_{c \in S_{\underline{c}}} N_j(c) = \sum_{x_j \in S_{\underline{x}_j}} N_j(x_j) = N_j$$

Figure 12.4: Some population numbers associated with the nodes of a dtree. $N_j(c, x_j)$ is the number of individuals σ in the population that reaches node j, belonging to class c and having $\underline{x}_j = x_j$.

Figure 12.5: Bnet derived from population numbers in Fig.12.4

Fig.12.4 defines some population numbers associated with the nodes of a dtree. From these population numbers, we can define the bnet in Fig.12.5. The TPMs, printed in blue, for the (non-root) nodes of this bnet, are as follows

$$P(c|x_j, j) = \frac{N_j(c, x_j)}{N_j(x_j)} \tag{12.1}$$

$$P(x_j|j) = \frac{N_j(x_j)}{N_j} \tag{12.2}$$

[3]SAM is also called, somewhat confusingly, the splitting criterion and Gain.

where $j \in J_0$ $x_j \in S_{\underline{x}_j}$, and $c \in S_{\underline{c}}$ is a class node.

One can define the following information theory quantities[4] associated with the bnet Fig.12.5.

$$\begin{aligned} INFO_fin_j &= -\sum_c P(c|j) \ln P(c|j) && (12.3) \\ &= H(\underline{c}|j) && (12.4) \\ & && (12.5) \end{aligned}$$

$$\begin{aligned} INFO_init_j &= \sum_{x_j \in S_{\underline{x}_j}} P(x_j|j) H(\underline{c}|x_j, j) && (12.6) \\ &= H(\underline{c}|\underline{x}_j, j) && (12.7) \end{aligned}$$

$$\begin{aligned} IG_j &= INFO_fin_j - INFO_init_j && (12.8) \\ &= H(\underline{c}|j) - H(\underline{c}|\underline{x}_j, j) && (12.9) \\ &= H(\underline{c} : \underline{x}_j|j) && (12.10) \end{aligned}$$

IG_j is called the **information gain for node** $\underline{x}_j$. Maximizing this mutual information produces a node $\underline{x}_j$ that has a large correlation to a class c. If the goal is to reach a point where each leaf node is closely correlated to a different class, then maximizing the Information Gain of each new node is a greedy move towards that goal. Thus, Information Gain is a good SAM for dtree SL.

Note that if we approximate

$$\begin{aligned} \ln P(c|x_j) &\approx \ln[1 + P(c|x_j) - 1] && (12.11) \\ &\approx P(c|x_j) - 1 && (12.12) \end{aligned}$$

in $H(\underline{c}|x_j)$, we get what is called the **Gini (or Gini Index) for node** $\underline{x}_k$:

$$\begin{aligned} H(\underline{c}|x_j) &= -\sum_c P(c|x_j) \ln P(c|x_j) && (12.13) \\ &\approx 1 - \sum_{c \in S_{\underline{c}}} P(c|x_j)^2 \stackrel{\text{def}}{=} Gini_{x_j} && (12.14) \end{aligned}$$

[4] The average of $H(\underline{c} : b)$ over b is $H(\underline{c} : \underline{b}) = \sum_b P(b) H(\underline{c} : b)$. Likewise, the average of $H(\underline{c}|b)$ over b is $H(\underline{c}|\underline{b}) = \sum_b P(b) H(\underline{c}|b)$. $H(\underline{c} : \underline{b})$ becomes $H(\underline{c} : b)$ and $H(\underline{c}|\underline{b})$ becomes $H(\underline{c}|b)$ when there is no $b-$prior (i.e., $P(b)$ is a delta function).

$Gini_{x_j}$ is a fairly good polynomial approximation to $H(\underline{c}|x_j)$. It is computationally much less expensive than $H(\underline{c}|x_j)$, because it does not require computing a log.

We say a probability distribution $P_{\underline{x}}$, is **pure (i.e., deterministic)** if $P_{\underline{x}}(x) = \delta(x, x_0)$. $Gini_{x_j}$ and $H(\underline{c}|x_j)$ are both always non-negative. They both vanish iff $P(c|x_j)$ is pure. Thus, $Gini_{x_j}$ and $H(\underline{c}|x_j)$ are both good measures of **class impurity**.

The **average Gini of node** $\underline{x}_j$ is defined as

$$AGini_j = \sum_{x_j \in S_{\underline{x}_j}} P(x_j|j) Gini_{x_j} \ . \tag{12.15}$$

It measure the average impurity of the children of node $\underline{x}_j$.

In practice, the SL algorithm is done recursively. Each recursion step decides which feature x_j will be the root node of the current tree. For all "candidate" features (i.e., all $\underline{x}_j$ that haven't been used yet as tree nodes), one calculates IG_j, either exactly or approximately via Gini's, using the following formula:

$$IG_j = \underbrace{H(\underline{c}|j)}_{\approx Gini_j} - \sum_{x_j \in S_{\underline{x}_j}} P(x_j|j) \underbrace{H(\underline{c}|x_j)}_{\approx Gini_{x_j}} \tag{12.16}$$

One then chooses $j = \underset{j}{\operatorname{argmax}}\, IG_j$. This maximizes the $\underline{c} - \underline{x}_j$ correlation.

Alternatively, some software programs use the average Gini $AGini_j$ as their SAM. They choose as root node $j = \underset{j}{\operatorname{argmin}}\, AGini_j$. This minimizes the average impurity of the children of node j. Since IG_j differs from $AGini_j$ by $H(\underline{c}|j)$, maximizing IG_j and minimizing $AGini_j$ might lead to different results.

Example of calculation of $AGini_j$ and IG_j

Suppose we deduce from a dataset the numbers in the yellow cells in Table 12.2. These numbers are repeated in Fig.12.6. Then we can calculate the white cells as follows:

$j = temp?$	$x_j = hot$	$x_j = med$	$x_j = cold$
$c = play$	3	3	3
$c = stay$	1	2	2
$Gini$	0.375	0.48	0.48
$AGini = 0.45$			
$H(\underline{c}\|x_j)$	≈ 0.375	≈ 0.48	≈ 0.48
$IG \approx 0$			

Table 12.2: Evaluating AGini and IG for node *temp*.

- Gini(hot)$= 1 - (3/4) - (1/4) = 0.375$

Figure 12.6: Tree stump corresponding to Table 12.2.

- Gini(med)$= 1 - (3/5)^2 - (2/5)^2 = 0.48$
- Gini(cold)$= 1 - (3/5)^2 - (2/5)^2 = 0.48$
- AGini $= (4/14)(0.375) + (5/14)(0.48) + (5/14)(0.48) = 0.45$
- IG $\approx [1 - (6/9)^2 - (3/9)^2] - AGini = 0.44 - 0.45 \approx 0$

This gives the AGini and IG for just one candidate root node. We would have to calculate AGini (or IG) for all possible candidates and choose the candidate with the lowest AGini (or highest IG).

12.3 Information Gain Ratio

The Information Gain Ratio (IGR) is an alternative SAM.

$$
\begin{aligned}
IGR_j &= \frac{IG_j}{H(\underline{x}_j|j)} && (12.17)\\
&= \frac{H(\underline{c} : \underline{x}_j|j)}{H(\underline{x}_j|j)} && (12.18)\\
&= \frac{H(\underline{x}_j|j) - H(\underline{x}_j|\underline{c}, j)}{H(\underline{x}_j|j)} && (12.19)\\
&= 1 - \frac{H(\underline{x}_j|\underline{c}, j)}{H(\underline{x}_j|j)} && (12.20)
\end{aligned}
$$

$0 \leq IGR_j \leq 1$
$IGR_j = 0$ iff $H(\underline{x}_j|\underline{c}, j) = H(\underline{x}_j|j)$.

12.3.1 Pseudo-code

Below, we give the standard algorithm for SL of a dtree, in the form of pseudo-code. The strategy employed by the algo is to assume an incoming population into the current root node, then determine the feature x_j that best separates that incoming population. The feature x_j is chosen so as to maximize IG_j (or minimize $AGini_j$).

This process is repeated by nominating the end of each new branch to be the current root node. Features can appear as a node more than once, so the order in which nodes are split does not matter. In essence, what we are doing is performing a top-down, greedy search through the space of possible dtrees.

The pseudo-code below describes the following historically important software programs:

- CART (Classification and Regression Trees), invented by Breiman et al in 1984. Uses $AGini_j$ as SAM.
- ID3 (Iterative Dichotomiser 3) invented by Quinlan in 1986. Uses IG_j as SAM. C4.5/C5.0 are successors to ID3.

Thus, CART and ID were invented independently around the same time. The main difference between them is the SAM being used.

The pseudo-code below uses the majority function defined in Chapter Notational Conventions and Preliminaries.

Algorithm 1: Pseudo-code for learning a dtree from a dataset

Input :

- dataset $DS = \{(\sigma, x^\sigma, c^\sigma) : \sigma \in \Sigma\}$
- set of currently available node indices J, where $J \subset J_0$

Output:

- tree T,
- population numbers $\{(r, c, x_r, N_r(c, x_r)) : r \in J_0, c \in S_{\underline{c}}, x_r \in S_{\underline{x}_r}\}$ stored globally

From DS, calculate J_0, $S_{\underline{c}}$, $S_{\underline{x}_i}$ for each i
c^σ is called the target feature/attribute.
$J \leftarrow J_0$
Function `learn_dtree`(DS, J)**:**
  $\Sigma \leftarrow$ set of all σ in DS
  if $\{c^\sigma : \sigma \in \Sigma\} = \{c\}$ **then**
    $T \leftarrow$ one node tree with leaf node label$= c$
  else if $J = \emptyset$ **then**
    $T \leftarrow$ one node tree with leaf node label$=$ `majority`$([c^\sigma : \sigma \in \Sigma])$
  else
    $r \leftarrow \operatorname{argmax}_{j \in J} IG_j(DS)$ `// or replace` $\operatorname{argmax}_{j \in J} IG_j$ `by` $\operatorname{argmin}_{j \in J} AGini_j$
    from DS, calculate $\{(r, c, x_r, N_r(c, x_r)) : c \in S_{\underline{c}}, x_r \in S_{\underline{x}_r}\}$ and store it globally
    for $v \in S_{\underline{x}_r}$ **do**
      `/* Notice that` J `is the same every time repeat this loop, so order in which` $v \in S_{\underline{x}_r}$ `are called does not matter. Furthermore, this means that multiple tree nodes may be labeled by same feature. */`
      On current tree T, add a branch below $\underline{x}_r$ with label "$\underline{x}_r = v$"
      $DS|_{\underline{x}_r = v} \leftarrow$ subset of DS with $\underline{x}_r = v$
      if $DS|_{\underline{x}_r = v} = \emptyset$ **then**
        below the new branch add a
        leaf node labeled $=$ `majority`$([c^\sigma : \sigma \in \Sigma])$
      else
        below the new branch add
        subtree $=$`learn_dtree`$(DS|_{\underline{x}_r = v}, J - \{r\})$
  return T

Chapter 13

Difference-in-Differences

This chapter is based on Ref.[6].

The Difference-in-Differences (DID) method was first used by John Snow in an 1854 report that argued that cholera in London was being transmitted by sewage polluted water rather than, as others at the time believed, by air (in fetid vapors called miasmas). In general, one can apply DID to discover causal effects in historical data. By **historical data** (aka a **natural experiment**. See Ref.[111]) we mean data that is collected long after the treatment (rather than during it) and is thus not subject to active intervention by the experimenter.

This chapter assumes that the reader has read Chapter 52 on Potential Outcomes (PO). The DID method applies the basic single-time PO theory described in Chapter 52, to 2 well separated times in which different conditions prevail.

13.1 John Snow, DID and a cholera transmission pathway

Let

$d \in \{0, 1\}$
$t \in \{t_0, t_1\}$, $t_0 < t_1$
$y = f(d, t) \in \mathbb{R}$.
Define

$$\Delta_t f(d, t) = f(d, t_1) - f(d, t_0) \ , \tag{13.1}$$

$$\Delta_d f(d, t) = f(1, t) - f(0, t) \ , \tag{13.2}$$

$$DID = \delta = \Delta_d \Delta_t f(d, t) \ . \tag{13.3}$$

DID is illustrated in Fig.13.1.

Figure 13.1: Pictorial representation of Difference-in-differences (DID) as a difference of two differences (i.e., a difference of two slopes).

A **time series** is any function of time for which the domain is a discrete set of times.

Figure 13.2: $DID = \delta$ expressed as difference of slopes or difference of vectors.

Note that, as shown Fig.13.2, $DID = \delta$ can also be expressed as a difference of 2 slopes times $\Delta t = t_1 - t_0$. Let $\hat{y}$ be a unit in the y direction. δ can also be expressed as the dot product of a difference of 2 vectors dotted with $\hat{y}$.

A condensation of the data collected by John Snow in 1854 is given in Table 13.1. From that data, we find that

$$\delta = \Delta_d \Delta_t f(d, t) = (19 - 85) - (147 - 135) = -66 - 12 = -78 \tag{13.4}$$

	$t = t_0$ (1849)	$t = t_1$ (1854)
$d = 1$ (town 1)	85 deaths, polluted DW	19 deaths, unpolluted DW
$d = 0$ (town 0)	135 deaths, polluted DW	147 deaths, polluted DW

Table 13.1: A condensation of the data collected by John Snow in 1854, to test the hypothesis that cholera in London was being spread by polluted drinking water (DW).

13.2 PO analysis

In this section, we show how to analyze the DID method using the formalism of PO theory.

We will speak of a treatment outcome $\underline{y}^{\sigma}_{t,g^{\sigma}}(c^{\sigma}, x^{\sigma})$ for individual σ that depends, not just on the treatment dose $c^{\sigma} \in \{0, 1\}$ and the confounder state x^{σ}, but also on a group parameter (i.e., which population or town) $g^{\sigma} \in \{0, 1\}$ and on a time parameter $t \in \{t_0, t_1\}$ (note t is independent of σ). Actually, we will assume $g^{\sigma} = c^{\sigma}$, so we will just speak of $\underline{y}^{\sigma}_{t}(c^{\sigma}, x^{\sigma})$ with no explicit g^{σ} dependence. As usual for PO theory, we will consider expected values of y^{σ}_{t}:

$$E_{\sigma|d,x}[y^{\sigma}_{t}(c)] = E_{\underline{y}_t(c)|d,x}[\underline{y}_t(c)] = \mathcal{Y}_{c|d,x}(t) \tag{13.5}$$

To calculate these expected values, we need a "model" with probability distributions. In this case, the needed model and probability distributions are provided by the bnets depicted in Fig.13.3. The TPMs, printed in blue, for the bnet $G_{t,+}$ in Fig.13.3, are as follows. Note that the TPMs for the bnet $G_{t,+}$ are defined in terms of the TPMs for the bnet G_t.

Figure 13.3: $t \in \{t_0, t_1\}$. Bnet $G_{t,+}$ is obtained by adding two new nodes $\underline{y}_t(0)$ and $\underline{y}_t(1)$ to bnet G_t.

$$P(x) = P_{\underline{x}}(x) \tag{13.6}$$

$$P(d|x) = P_{\underline{d}|\underline{x}}(d|x) \tag{13.7}$$

$$P(y_t|y_t(0), y_t(1), d) = \mathbb{1}(y_t = y_t(d)) \tag{13.8}$$

$$P(y_t(c)|x) = P(y_t(c)|d, x) = \text{given} \tag{13.9}$$

Figure 13.4: Four different time-dependent expected values $\mathcal{Y}_{c|d}(t)$ of y_t^σ for bnet $G_{t,+}$ The 4 magenta stars represents the 4 DID measurements.

We define the function $c(t)$ for $t = t_0, t_1$ by

$$c(t) = \begin{cases} 0 & \text{if } t = t_0 \\ d & \text{if } t = t_1 \end{cases} \tag{13.10}$$

Now we claim that the DID δ calculated in the previous section for John Snow's data, can be expressed in PO formalism as follows:

$$\delta = \Delta_d \Delta_t \mathcal{Y}_{c(t)|d}(t) \ . \tag{13.11}$$

Fig.13.4 depicts the four functions $\mathcal{Y}_{c|d}(t)$ for t in the interval $[t_0, t_1]$ and for $c, d \in \{0, 1\}$. The $\mathcal{Y}$ coordinates of the four magenta stars in Fig.13.4 can be calculated using bnet G_t.

Define the **parallel trends** (PT) by

$$PT = \Delta_d \Delta_t \mathcal{Y}_{0|d}(t) \ . \tag{13.12}$$

We will say the **parallel trends assumption (PTA)** holds if $PT = 0$.

Next we prove that the DID δ equals the sum of an ATT[1] and PT.

$$
\begin{aligned}
\delta &= \Delta_d \Delta_t \mathcal{Y}_{c(t)|d}(t) && (13.13)\\
&= \left[\Delta_t \mathcal{Y}_{c(t)|1}(t) - \Delta_t \mathcal{Y}_{c(t)|0}(t)\right] && (13.14)\\
&= \mathcal{Y}_{1|1}(t_1) - \mathcal{Y}_{0|1}(t_0) - \{\mathcal{Y}_{0|0}(t_1) - \mathcal{Y}_{0|0}(t_0)\} && (13.15)\\
&= \mathcal{Y}_{1|1}(t_1) - \mathcal{Y}_{0|1}(t_0) - \{\mathcal{Y}_{0|0}(t_1) - \mathcal{Y}_{0|0}(t_0)\} + \underbrace{\{\mathcal{Y}_{0|1}(t_1) - \mathcal{Y}_{0|1}(t_1)\}}_{\text{zero}} && (13.16)\\
&= \underbrace{\mathcal{Y}_{1|1}(t_1) - \mathcal{Y}_{0|1}(t_1)}_{ATT(t_1)} - \mathcal{Y}_{0|1}(t_0) - \{\mathcal{Y}_{0|0}(t_1) - \mathcal{Y}_{0|0}(t_0)\} + \mathcal{Y}_{0|1}(t_1) && (13.17)\\
&= ATT(t_1) - \Delta_t \mathcal{Y}_{0|0}(t) + \Delta_t \mathcal{Y}_{0|1}(t) && (13.18)\\
&= ATT(t_1) + \underbrace{\Delta_d \Delta_t \mathcal{Y}_{0|d}(t)}_{\text{zero if PTA holds}} && (13.19)
\end{aligned}
$$

13.3 Linear Regression

In this section, we show how to apply linear regression (LR) to the PO analysis of DID.

As before, let $y_t^\sigma(c^\sigma)$ be the treatment outcome for individual σ, who receives a treatment dose c^σ at times $t \in \{t_0, t_1\}$. $y_t^\sigma(c^\sigma)$ can be fitted as follows. Here ϵ^σ is the residual for individual σ, and $b_0, m_0, b_1, m_1 \in \mathbb{R}$ are the fit parameters.

$$
y_t^\sigma = [b_0 + m_0(t - t_0)](1 - c^\sigma) + [b_1 + m_1(t - t_0)]c^\sigma + \epsilon^\sigma \ . \tag{13.20}
$$

Note that Eq.(13.20) yields a straight line in the $y_t^\sigma - t$ plane for $c^\sigma = 0$, and another straight line for $c^\sigma = 1$. We are using the standard symbols b to denote the y-intercept, and m to denote the slope of a straight line.

Taking the expected value of Eq.(13.20), we get

$$
\mathcal{Y}_{c|d}(t) = [b_0 + m_0(t - t_0)](1 - c) + [b_1 + m_1(t - t_0)]c \ . \tag{13.21}
$$

If $\Delta t = t_1 - t_0$, then

$$
\mathcal{Y}_{d|d}(t_1) = [b_0 + m_0 \Delta t](1 - d) + [b_1 + m_1 \Delta t]d \ , \tag{13.22}
$$

and

$$
\mathcal{Y}_{0|d}(t_0) = b_0 \ . \tag{13.23}
$$

[1] ATT stands for the average treatment effect of the treated. ATT is defined in Chapter 52

Thus,

$$\begin{aligned}
\delta &= \Delta_d \Delta_t \mathcal{Y}_{c(t)|d}(t) && (13.24)\\
&= \Delta_d [\mathcal{Y}_{d|d}(t_1) - \mathcal{Y}_{0|d}(t_0)] && (13.25)\\
&= (m_1 - m_0)\Delta t \ . && (13.26)
\end{aligned}$$

Figure 13.5: We use Linear Regression to fit a straight line between points U_0 and U_1, and between points T_0 and T_1. (U=untreated, T=treated, subscript refers to times t_0, t_1). U_0, T_0, U_1, T_1 are the measurement points. Point I is an image of point U_1.

	$t = t_0$	$t = t_1$
$d = 1$	$\mathcal{Y}(T_0) = \mathcal{Y}_{0\|1}(t_0)$	$\mathcal{Y}(T_1) = \mathcal{Y}_{1\|1}(t_1)$
$d = 0$	$\mathcal{Y}(U_0) = \mathcal{Y}_{0\|0}(t_0)$	$\mathcal{Y}(U_1) = \mathcal{Y}_{0\|0}(t_1)$

Table 13.2: $\mathcal{Y}$ coordinates of points U_0, T_0, U_1, T_1 in Figs.13.4 and 13.5.

Figs.13.4 and 13.5 define points U_0, T_0, U_1, T_1, I, A. The $\mathcal{Y}$ coordinates of points U_0, T_0, U_1, T_1 are given by Table 13.2. The $\mathcal{Y}$ coordinates of points A, I are given by Eqs.13.27.

$$\mathcal{Y}(A) = \mathcal{Y}_{0|1}(t_1) \tag{13.27a}$$

$$\mathcal{Y}(I) = \mathcal{Y}(U_1) + [\mathcal{Y}(T_0) - \mathcal{Y}(U_0)] \tag{13.27b}$$

We can express ATT and the δ for DID in terms of the $\mathcal{Y}$ of the points U_0, T_0, U_1, T_1, I, A. Indeed,

$$\begin{aligned} \delta &= \mathcal{Y}(T_1) - \mathcal{Y}(I) && (13.28) \\ &= \mathcal{Y}(T_1) - \mathcal{Y}(U_1) - [\mathcal{Y}(T_0) - \mathcal{Y}(U_0)] && (13.29) \end{aligned}$$

$$ATT = \mathcal{Y}(T_1) - \mathcal{Y}(A) \tag{13.30}$$

Hence,

$$\delta = ATT \iff \mathcal{Y}(I) = \mathcal{Y}(A) \iff \text{PTA holds} \tag{13.31}$$

Chapter 14

Digital Circuits

Figure 14.1: Typical digital circuit of NAND gates.

Digital (logic) gate: node with na input ports and nx output ports which represents a function

$$f : \{0,1\}^{na} \to \{0,1\}^{nx} \ . \tag{14.1}$$

Suppose
$a^{na} = (a_i)_{i=0,1,\ldots,na-1}$ where $a_i \in \{0,1\}$,
$x^{nx} = (x_i)_{i=0,1,\ldots,nx-1}$ where $x_i \in \{0,1\}$.
f maps a^{na} into x^{nx}.
Digital circuit (dcircuit) = circuit of digital gates.

Figure 14.2: 2 options for mapping dcircuit node with multiple output ports into bnet.

14.1 Mapping any dcircuit to a bnet

14.1.1 Option A of Fig.14.2

1. Replace every dcircuit gate described by Eq.(14.1) by nx bnet nodes $\underline{x}_i$ for $i = 0, 1, \ldots, nx - 1$ such that

$$P(x_i|a^{na}) = \delta(x_i, f_i(a^{na})) \tag{14.2}$$

2. Replace all connectors of the dcircuit by arrows pointing in the direction of the bit flow.

14.1.2 Option B of Fig.14.2

1. Replace every dcircuit gate described by Eq.(14.1) with one bnet node called $\underline{x}^{nx}$ and, if $nx > 0$, nx "marginalizer nodes" $\underline{m}_i$ for $i = 0, 1, \ldots, nx - 1$, such that

$$P(x^{nx}|a^{na}) = \delta(x^{nx}, f(a^{na})) \ , \tag{14.3}$$

and

$$P(m_i|x^{nx}) = \delta(m_i, x_i) \ . \tag{14.4}$$

2. Replace all connectors of the dcircuit by arrows pointing in the direction of the bit flow.

Options A and B don't work for digital circuits with feedback loops such as flip-flops. Those could probably be modeled with dynamical bnets.

Chapter 15

Do Calculus

The Do Calculus and associated ideas were invented by Judea Pearl and collaborators. This chapter is based on Judea Pearl's books (see Navigating the ocean of Judea Pearl's Books).

When doing Do Calculus, it is convenient to separate the nodes of a bnet into 2 types: **visible (observed)**, and **non-visible (unobserved, latent, hidden)**, depending on whether data describing the state of that node is available (visible) or not (non-visible). In this chapter, every hidden node will be indicated in a bnet diagram by either: (1) enclosing its random variable in a dashed circle or (2) making the arrows coming out of it dashed. Accordingly, the 3 diagrams in Fig.15.1 all mean the same thing.

A **confounder node** $\underline{c}$ for nodes $\underline{x}$ and $\underline{y}$ is a root node with arrows pointing from it to both $\underline{x}$ and $\underline{y}$. Thus, $\underline{c}$ acts as a common cause of $\underline{x}$ and $\underline{y}$. The node $\underline{c}$ in Fig.15.1) is an **unobserved confounder node**.

Figure 15.1: These 3 diagrams are equivalent. They mean that node $\underline{c}$ is hidden. Node $\underline{c}$ is implicit in the middle diagram.

Let $\mathcal{D}_{\underline{x}}$ be an operator that acts on a graph G with a node $\underline{x}$ by deleting all the arrows entering $\underline{x}$, thus converting $\underline{x}$ into a new node $\mathcal{D}\underline{x}$ that is a root node. Let $\mathcal{L}_{\underline{x}}$ be an operator that acts on a graph G with a node $\underline{x}$ by deleting all the arrows leaving $\underline{x}$, thus converting $\underline{x}$ into a new node $\mathcal{L}\underline{x}$ that is a leaf node. $\mathcal{D}_{\underline{x}}$ and $\mathcal{L}_{\underline{x}}$ are depicted in Fig.15.2.

If you don't know yet what we mean by a multi-node $\underline{a}$., see Chapter Definition of a Bayesian Network

Figure 15.2: The do operator $\mathcal{D}_{\underline{x}}$ converts node $\underline{x}$ into a root node $\mathcal{D}\underline{x}$. The leaf operator $\mathcal{L}_{\underline{x}}$ converts node $\underline{x}$ into a leaf node $\mathcal{L}\underline{x}$.

Given a bnet G, we define as follows the operators $\mathcal{D}_{\underline{a}.}$ and $\mathcal{L}_{\underline{a}.}$ for a multi-node $\underline{a}.$.

$$\mathcal{D}_{\underline{a}.}G = \left[\prod_j \mathcal{D}_{\underline{a}_j}\right] G \ , \quad \mathcal{L}_{\underline{a}.}G = \left[\prod_j \mathcal{L}_{\underline{a}_j}\right] G \ . \tag{15.1}$$

Consider a bnet whose totality of nodes is labeled $\underline{X}.$. Recall that

$$P(X.) = \prod_j P(X_j|(X_k)_{k:\underline{X}_k \in pa(\underline{X}_j)}) \ . \tag{15.2}$$

Define an operator $\mathcal{D}$ that acts as follows[1]: Let $X. - a. = (X_k)_{k:\underline{X}_k \notin \underline{a}.}$.

$$\begin{aligned} P(X. - a.|\mathcal{D}\underline{a}. = a.) &= \mathcal{N}(!(X. - a.))\frac{P(X.)}{\prod_{j:\underline{X}_j \in \underline{a}.} P(X_j|(X_k)_{k:\underline{X}_k \in pa(\underline{X}_j)})} && (15.3) \\ &= \mathcal{N}(!(X. - a.)) \prod_{j:\underline{X}_j \notin \underline{a}.} P(X_j|(X_k)_{k:\underline{X}_k \in pa(\underline{X}_j)}) && (15.4) \\ &\neq P(X. - a.|\underline{a}. = a.) \ . && (15.5) \end{aligned}$$

Also,

$$P(\mathcal{D}\underline{a}. = a.) = \delta(a'., a.) \ . \tag{15.6}$$

In words, we replace the TPM for multinode $\underline{a}.$ by a delta function.

For instance, for the bnet

$$\underline{r} \longrightarrow \underline{x} \longrightarrow \underline{y} \tag{15.7}$$

with

$$P(r, x, y) = P(y|x)P(x|r)P(r) \ , \tag{15.8}$$

[1]As usual, $\mathcal{N}(!x)$ denotes a constant that is independent of x.

one has

$$P(r, y|\mathcal{D}\underline{x} = x) = P(y|x)P(r) \tag{15.9}$$

Hence,

$$P(y|\mathcal{D}\underline{x} = x) = P(y|x) \tag{15.10}$$

For the bnet

$$\begin{tikzcd} \underline{c} \arrow[d] \arrow[dr] & \\ \underline{x} \arrow[r] & \underline{y} \end{tikzcd} \tag{15.11}$$

with

$$P(x, y, c) = P(y|x, c)P(x|c)P(c) \ , \tag{15.12}$$

one has

$$P(y, c|\mathcal{D}\underline{x} = x) = P(y|x, c)P(c) \ . \tag{15.13}$$

Hence,

$$P(y|\mathcal{D}\underline{x} = x) = \sum_c P(y|x, c)P(c) \ . \tag{15.14}$$

This is called **adjusting the parents of** $\underline{x}$.

For $\underline{b}. \subset \underline{X}. - \underline{a}.$, define

$$P(b.|\mathcal{D}\underline{a}. = a.) = \sum_{X.-a.-b.} P(X. - a.|\mathcal{D}\underline{a}. = a.) \ , \tag{15.15}$$

and for $\underline{s}. \subset \underline{X}. - \underline{a}. - \underline{b}.$, define

$$P(b.|\mathcal{D}\underline{a}. = a., s.) = \frac{P(b., s.|\mathcal{D}\underline{a}. = a.)}{P(s.|\mathcal{D}\underline{a}. = a.)} \ . \tag{15.16}$$

$P(b.|\mathcal{D}\underline{a}. = a., s.)$ is denoted by Pearl by $P(b.|do(\underline{a}. = a.), s.)$. I prefer to use $\mathcal{D}$ instead of $do()$. I will still call $\mathcal{D}$ a **do operator**.

In $P(y|\mathcal{D}\underline{x} = x)$, node $\underline{x}$ is turned into a root node. This guarantees that there is no confounding node connecting $\underline{x}$ and $\underline{y}$. Such confounding nodes are unwelcomed when calculating causal effects between the 2 variables $\underline{x}$ and $\underline{y}$ because they introduce non-causal correlations between the two. This is also what happens in a **Randomized Controlled Trial (RCT)**. In a RCT with treatment $\underline{x}$, the value of $\underline{x}$ for each patient is determined by a coin toss, effectively turning $\underline{x}$ into a root node. Hence, the do operator mimics a RCT.

$P(b.|\mathcal{D}\underline{a}. = a., s.)$ is said to be **do-identifiable** (i.e., expressible without do()) if it can be expressed in terms of probability distributions that only depend on observed variables, and that have no do operators in them.[2]

For $\underline{x}, \underline{y} \in \{0, 1\}$, the **average controlled effect (ACE)** is defined as

$$ACE = P(y = 1|\mathcal{D}\underline{x} = 1) - P(y = 1|\mathcal{D}\underline{x} = 0) \tag{15.17}$$

and the **Risk Difference (RD)** is defined as

$$RD = P(y = 1|\underline{x} = 1) - P(y = 1|\underline{x} = 0) \ . \tag{15.18}$$

15.1 Parent Adjustment Formula

Suppose that $\underline{x}., \underline{y}., \underline{z}.$ are disjoint multinodes and their union equals the totality of all nodes of a bnet. Suppose we have data available that allows us to estimate $P(x., y., z.)$. Hence, all nodes of the bnet are observable. Furthermore, suppose $\underline{z}. = pa(\underline{x}.)$. In other words, we are considering the bnet

$$\begin{tikzcd} \underline{z}. \arrow[d] \arrow[dr] & \\ \underline{x}. \arrow[r] & \underline{y}. \end{tikzcd} \ . \tag{15.19}$$

Then

$$P(y., z.|\mathcal{D}\underline{x}. = x.) = P(y.|x., z.)P(z.) \tag{15.20}$$

so

$$P(y.|\mathcal{D}\underline{x}. = x.) = \sum_{z.} P(y.|x., z.)P(z.) \ . \tag{15.21}$$

This is called **adjusting the parents** of $\underline{x}.$.

We say that we are **adjusting or controlling a node** $\underline{a}$ if we condition a probability on $\underline{a}$ and then we average that probability over $\underline{a}$. More generally, we can adjust a whole multinode $\underline{a}.$ together.

Later on, we will introduce a generalization of this parent adjustment formula called the backdoor adjustment formula. In a backdoor adjustment formula, the adjusted multinode is not necessarily the parents of $\underline{x}.$, and $P(x., y., z.)$ need not represent the whole bnet.

[2]In Statistics, one says a probability distribution $P(x; \theta)$ of x that depends on a parameter θ is **identifiable** if $P(x; \theta_1) = P(x; \theta_2)$ implies $\theta_1 = \theta_2$.

15.2 3 Rules of Do Calculus

Throughout this section, suppose $\underline{a}., \underline{b}., \underline{r}., \underline{s}.$ are disjoint multinodes in a bnet G.

Recall from Chapter 17 on d-separation, that $(\underline{b}. \perp_G \underline{a}.|\underline{r}., \underline{s}.)$ means that we have established from the d-separation rules that that all paths in G from $\underline{a}.$ to $\underline{b}.$ are blocked if we condition on $\underline{r}. \cup \underline{s}.$. Recall also that:

Rule 0: Insertion or deletion of observations, without do operators.

If $(\underline{b}. \perp \underline{a}.|\underline{r}., \underline{s}.)$ in G, then

✓ $P(b.|a., r., s.) = P(b.|r., s.)$ (i.e., $\underline{a}. = a. \leftrightarrow 1$)

$$\begin{tikzcd} & r. \arrow[dr] & s. \arrow[d] \\ a. \arrow[rr] & & b. \end{tikzcd} \quad = \quad \begin{tikzcd} & r. \arrow[dr] & s. \arrow[d] \\ a. \arrow[rr, "0"] & & b. \end{tikzcd}$$

✓ $H(\underline{b}. : \underline{a}.|\underline{r}., \underline{s}.) = 0$

Zeroing an arrow is the same as deleting it. The 3 rules of Do Calculus can be presented in the same format.

- **Rule 1:** Insertion or deletion of observations

  If $(\underline{b}. \perp \underline{a}.|\underline{r}., \underline{s}.)$ in $\mathcal{D}_{\underline{r}.}G$, then

  ✓ $P(b.|a., \mathcal{D}\underline{r}. = r., s.) = P(b.|\mathcal{D}\underline{r}. = r., s.)$ (i.e., $\underline{a}. = a. \leftrightarrow 1$)

$$\begin{tikzcd} & \mathcal{D}\underline{r}. = r. \arrow[dr] & s. \arrow[d] \\ a. \arrow[rr] & & b. \end{tikzcd} \quad = \quad \begin{tikzcd} & \mathcal{D}\underline{r}. = r. \arrow[dr] & s. \arrow[d] \\ a. \arrow[rr, "0"] & & b. \end{tikzcd}$$

  ✓ $H(\underline{b}. : \underline{a}.|\mathcal{D}\underline{r}., \underline{s}.) = 0$

- **Rule 2:** Action or observation exchange

  If $(\underline{b}. \perp \underline{a}.|\underline{r}., \underline{s}.)$ in $\mathcal{L}_{\underline{a}.}\mathcal{D}_{\underline{r}.}G$, then

  ✓ $P(b.|\mathcal{D}\underline{a}. = a., \mathcal{D}\underline{r}. = r., s.) = P(b.|a., \mathcal{D}\underline{r}. = r., s.)$ (i.e., $\mathcal{D}\underline{a}. = a. \leftrightarrow \underline{a}. = a.$)

$$\begin{tikzcd} & \mathcal{D}\underline{r}. = r. \arrow[dr] & s. \arrow[d] \\ \mathcal{D}\underline{a}. = a. \arrow[d] \arrow[rr] & & b. \\ & & \end{tikzcd} \quad = \quad \begin{tikzcd} & \mathcal{D}\underline{r}. = r. \arrow[dr] & s. \arrow[d] \\ \mathcal{D}\underline{a}. = a. \arrow[d] & a. \arrow[rr] & & b. \end{tikzcd}$$

$$\begin{tikzcd} & \mathcal{D}\underline{r}. = r. \arrow[dr] & s. \arrow[d] \\ a. \arrow[d] \arrow[rr] & & b. \\ & & \end{tikzcd} \quad = \quad \begin{tikzcd} & \mathcal{D}\underline{r}. = r. \arrow[dr] & s. \arrow[d] \\ a. \arrow[d] & \mathcal{D}\underline{a}. = a. \arrow[rr] & & b. \\ & & \end{tikzcd}$$

  In this rule, the node is split into a $\mathcal{D}\underline{a}. = a.$ node and a $a.$ node. The original node keeps the arrows, and the new node is a root node.

✓ $H(\underline{b}. : \mathcal{D}\underline{a}.|\mathcal{D}\underline{r}., \underline{s}.) = H(\underline{b}. : \underline{a}.|\mathcal{D}\underline{r}., \underline{s}.)$

- **Rule 3:** Insertion and deletion of actions

  If $(\underline{b}. \perp \underline{a}.|\underline{r}., \underline{s}.)$ in $\mathcal{D}_{\underline{a}.-an(\underline{s}.)}\mathcal{D}_{\underline{r}.}G$, then

  ✓ $P(b.|\mathcal{D}\underline{a}. = a., \mathcal{D}\underline{r}. = r., s.) = P(b.|\mathcal{D}\underline{r}. = r., s.)$ (i.e., $\mathcal{D}\underline{a}. = a. \leftrightarrow 1$)

$$\begin{tikzcd} & \mathcal{D}\underline{r}. = r. \arrow[dr] & s. \arrow[d] \\ \mathcal{D}\underline{a}. = a. \arrow[rr] & & b. \end{tikzcd} \quad = \quad \begin{tikzcd} & \mathcal{D}\underline{r}. = r. \arrow[dr] & s. \arrow[d] \\ \mathcal{D}\underline{a}. = a. \arrow[rr, "0"] & & b. \end{tikzcd}$$

  ✓ $H(\underline{b}. : \mathcal{D}\underline{a}.|\mathcal{D}\underline{r}., \underline{s}.) = 0$

See Fig.15.3 for a pictorial representation of these rules.

These rules have been proven to be sufficient for removing all do operators from an expression for which it is possible to do so.

Next we discuss two formulae that can be proven using Do Calculus: the backdoor and the frontdoor adjustment formulae.

The backdoor formula adjusts one multinode and the frontdoor formula adjusts two.

15.3 Backdoor Adjustment Formula

See Chapter 3 for examples of the use of the backdoor adjustment formula. In this section, we shall mainly be concerned with proving this theorem using Do Calculus.

For any two disjoint multinodes $\underline{x}.$ and $\underline{y}.$, we define a **backdoor path** from $\underline{x}.$ to $\underline{y}.$ as a path from $\underline{x}.$ and $\underline{y}.$ that starts with an arrow pointing into $\underline{x}.$,

Suppose that we have access to data that allows us to estimate a probability distribution $P(x., y., z.)$. Hence, the variables $\underline{x}., \underline{y}., \underline{z}.$ are ALL observed (i.e, not hidden). Then we say that the backdoor $\underline{z}.$ satisfies the **backdoor adjustment criterion** relative to $(\underline{x}., \underline{y}.)$ if

1. All backdoor paths from $\underline{x}.$ to $\underline{y}.$ are blocked by conditioning on $\underline{z}.$.
2. $\underline{z}. \cap de(\underline{x}.) = \emptyset$.

Motivation for BD criterion: Part 1 rules out paths from $\underline{x}$ to $\underline{y}$ containing a fork node (confounder) which, if not blocked by conditioning on $\underline{z}.$, would introduce a non-causal correlation (confounder bias). Part 2 rules out a directed path from $\underline{x}$ to $\underline{y}$ that has a mediator node blocked by conditioning on $\underline{z}.$ or a collider node unblocked by conditioning on $\underline{z}.$.

Claim 17 *Backdoor Adjustment Formula*

If $\underline{z}.$ satisfies the backdoor criterion relative to $(\underline{x}., \underline{y}.)$, then

Figure 15.3: Pictorial representation of the rules of Do Calculus. $an(\underline{s}.)$ stands for the ancestors of $\underline{s}.$. In Rule 2, we zero the arrows leaving $\underline{a}.$ and ascertain that there is no flow of info between $\underline{a}.$ and $\underline{b}.$ under those circumstances. In Rule 3, as shown by the dotted arrows, conditioning on $\underline{s}.$ is enough to block paths going from $\underline{b}.$ into $\underline{a}. - an(\underline{s}.)$, but it is not sufficient to block paths going from $\underline{b}.$ into $\underline{a}. \cap an(\underline{s}.)$. In the example with red dotted arrows, conditioning on $\underline{s}.$ opens a path between $\underline{a}.$ and $\underline{b}.$ in which $\underline{s}.$ is a collider. That is the reason for zeroing arrows going into $\underline{a}. - an(\underline{s}.)$, but not zeroing arrows going into $\underline{a}. \cap an(\underline{s}.)$.

$$P(y.|\mathcal{D}\underline{x}. = x.) = \sum_{z.} P(y.|x., z.)P(z.) \tag{15.22}$$

$$= \sum z. \tag{15.23}$$

$$\begin{tikzcd} & z. \arrow[dr] & \\ x. \arrow[rr] & & y. \end{tikzcd}$$

where $\sum z.$ means node $\underline{z}.$ is summed over.

proof:

For simplicity, let us omit the dots from the multinodes. If z satisfies the backdoor criterion relative to $(\underline{x}, \underline{y})$, then $\underline{x}, \underline{y}, \underline{z}$ might have the following structure.

(15.24)

See Claim 21 for a proof of this claim.

QED

Note that the backdoor adjustment formula can be written as

$$P(y.|\mathcal{D}\underline{x}. = x.) = \sum_{z.} P(y.|x., z.)P(z.) \tag{15.25}$$

$$= \sum_{z.} \frac{P(y., x., z.)}{P(x.|z.)} \tag{15.26}$$

This assumes $P(x.|z.) \neq 0$ for all $x., z.$. This assumption is referred to as **positivity**, and is violated if $P(x.|z.) = \delta(x., x.(z.))$. $P(x.|z.)$ is called the **propensity score** of $x.$ given $z.$. This equation does **inverse probability weighting**. One can approximate $P(x.|z.)$ in this equation to get an approximation to $P(y|\mathcal{D}\underline{x} = x)$.

15.4 Frontdoor Adjustment Formula

See Chapter 23 for examples of the use of the frontdoor adjustment formula. In this section, we shall mainly be concerned with proving this theorem using Do Calculus.

Suppose that we have access to data that allows us to estimate a probability distribution $P(x., m., y.)$. Hence, the variables $\underline{x}., \underline{m}., \underline{y}.$ are ALL observed (i.e, not hidden). Then we say that the frontdoor $\underline{m}.$ satisfies the **frontdoor adjustment criterion** relative to $(\underline{x}., \underline{y}.)$ if

1. All directed paths from $\underline{x}.$ to $\underline{y}.$ are intercepted by (i.e., have a node in) $\underline{m}.$.
2. All backdoor paths from $\underline{x}.$ to $\underline{m}.$ are blocked.
3. All backdoor paths from on $\underline{m}.$ to $\underline{y}.$ are blocked by conditioning on $\underline{x}.$.

Claim 18 *Frontdoor Adjustment Formula*

If $\underline{m}.$ satisfies the frontdoor criterion relative to $(\underline{x}., \underline{y}.)$, and $P(x., m.) > 0$, then

$$P(y.|\mathcal{D}\underline{x}. = x.) = \sum_{m.} \underbrace{\left[\sum_{x'.} P(y.|x'., m.)P(x'.)\right]}_{P(y.|\mathcal{D}\underline{m}.=m.)} \underbrace{P(m.|x.)}_{P(m.|\mathcal{D}\underline{x}.=x.)} \tag{15.27}$$

$$= \tag{15.28}$$

$$\begin{tikzcd} & \sum x'. \arrow[dr] & \\ x. \arrow[r] & \sum m. \arrow[r] & y. \end{tikzcd}$$

where $\sum x'.$ and $\sum m.$ means nodes $\underline{x}'.$ and $\underline{m}.$ are summed over.

proof:

For simplicity, let us omit the dots from the multinodes. If $\underline{m}$ satisfies the frontdoor criterion relative to $(\underline{x}, \underline{y})$, then $\underline{x}, \underline{m}, \underline{y}$ might have the following structure, where node $\underline{c}$ is unobserved.

$$\begin{tikzcd} & \underline{c} \arrow[dl] \arrow[dr] & \\ \underline{x} \arrow[r] & \underline{m} \arrow[r] & \underline{y} \end{tikzcd} \tag{15.29}$$

See Claim 22 for a proof of this claim.

See also Ref.[32] for a proof by Pearl of the Frontdoor Adjustment Formula without using Do Calculus.

QED

15.5 Do operator for DEN diagrams

Recall that the structural equations for a linear DEN, as given by Eq.(36.21) of Chapter 36, are:

$$\underline{x} = A\underline{x} + \underline{u} \ . \tag{15.30}$$

Therefore,

$$\underline{x} = (1 - A)^{-1}\underline{u} \tag{15.31}$$

which can be represented for both linear and non-linear DEN diagrams by:

$$\underline{x}_i = x_i(\underline{u}.) \tag{15.32}$$

If now we apply the operator $\mathcal{D}_{\underline{a}=a}$ to the diagram described by the structural equations Eqs.15.30, we get the following new structural equations:

$$\underline{x}_i^* = \begin{cases} \sum_{j<i} A_{i|j}\underline{x}_j^* + \underline{u}_i & \text{if } \underline{x}_i \neq \underline{a} \\ a & \text{if } \underline{x}_i = \underline{a} \end{cases} , \tag{15.33}$$

where we are calling $\underline{x}_i^*$ the nodes of the DEN diagram post intervention.

Eqs.(15.33) can be expressed in matrix notation as follows. Define $\pi_{\underline{a}}$ to be the $nx \times nx$ matrix with all entries equal to zero except for the (i_0, i_0) entry, which is 1. And define $e_{\underline{a}}$ to be the column vector with all entries zero except for the i_0'th one, which is 1. Here i_0 is defined so that $\underline{x}_{i_0} = \underline{a}$. In other words, $\pi_{\underline{a}}$ and $e_{\underline{a}}$ are defined by

$$(\pi_{\underline{a}})_{i,j} = \mathbb{1}(i = j, \underline{a} = \underline{x}_i) \tag{15.34}$$

and

$$(e_{\underline{a}})_i = \mathbb{1}(\underline{a} = \underline{x}_i) , \tag{15.35}$$

for $i, j \in \{0, 1, \ldots, nx - 1\}$. Next define

$$\pi_{!\underline{a}} = 1 - \pi_{\underline{a}} , \tag{15.36}$$

$$A^* = \pi_{!\underline{a}} A , \tag{15.37}$$

and

$$\underline{u}_{!\underline{a}} = \pi_{!\underline{a}} \underline{u} . \tag{15.38}$$

The effect of pre-multiplying the matrix A and the column vector $\underline{u}$ by $\pi_{!\underline{a}}$ is to leave all rows intact except for the i_0 row, which is set to zero. Here i_0 is defined by $\underline{a} = \underline{x}_{i_0}$. Finally, using all of the variables just defined, we can express the structural equations of the linear DEN diagram, post intervention, as

$$\underline{x}^* = A^* \underline{x}^* + \underline{u}_{!\underline{a}} + a e_{\underline{a}} . \tag{15.39}$$

Thus,

$$\underline{x}^* = (1 - A^*)^{-1} (\underline{u}_{!\underline{a}} + a e_{\underline{a}}) . \tag{15.40}$$

which can be represented for both linear and non-linear DEN diagrams by:

$$\underline{x}_i^* = x_i^*(\underline{u}_{!\underline{a}}, a) . \tag{15.41}$$

For any bnet,

$$P(\underline{y} = y | \underline{x} = x) = P_G(\underline{y} = y | \underline{x} = x) \tag{15.42}$$

$$P(\underline{y} = y|\mathcal{D}\underline{x} = x) = P_{\mathcal{D}_{\underline{x}=x}G}(\underline{y} = y) \tag{15.43}$$

Claim 19 *For a non-linear DEN diagram,*

$$P(y|\mathcal{D}\underline{x} = x) = E\left[\delta[y, y(\underline{u}_{!\underline{x}}, x)]\right] \ . \tag{15.44}$$

proof:

$$\begin{aligned}
P(\underline{y} = y|\mathcal{D}\underline{x} = x) &= P_{\mathcal{D}_{\underline{x}=x}G}(\underline{y} = y) && (15.45)\\
&= \sum_{u_{!\underline{x}}} P(u_{!\underline{x}}) P_{\mathcal{D}_{\underline{x}=x}G}(\underline{y} = y|u_{!\underline{x}}) && (15.46)\\
&= \sum_{u_{!\underline{x}}} P(u_{!\underline{x}}) \delta[y, y(u_{!\underline{x}}, x)] && (15.47)\\
&= E_{\underline{u}_{!\underline{x}}}[\delta[y, y(u_{!\underline{x}}, x)]] && (15.48)\\
&= E[\delta[y, y(\underline{u}_{!\underline{x}}, x)]] && (15.49)
\end{aligned}$$

QED

Claim 20 *For a nonlinear DEN diagram,*

$$E[\underline{y}|\mathcal{D}\underline{x} = x] = E[y(\underline{u}_{!\underline{x}}, x)] \ . \tag{15.50}$$

proof:

$$\begin{aligned}
E[\underline{y}|\mathcal{D}\underline{x} = x] &= \sum_y y P(\underline{y} = y|\mathcal{D}\underline{x} = x) && (15.51)\\
&= \sum_y y E[\delta[y, y(u_{!\underline{x}}, x)]] && (15.52)\\
&= E[y(\underline{u}_{!\underline{x}}, x)] && (15.53)
\end{aligned}$$

QED

For any bnet

$$P(y|\mathcal{D}\underline{x} = x, z) = \frac{P(y, z|\mathcal{D}\underline{x} = x)}{P(z|\mathcal{D}\underline{x} = x)} = P_{\mathcal{D}_{\underline{x}=x}G}(y|x, z) \tag{15.54}$$

For a nonlinear DEN diagram,

$$P(y, z|\mathcal{D}\underline{x} = x) = \sum_{u_{!\underline{x}}} P(u_{!\underline{x}}) \delta[y, y(u_{!\underline{x}}, x)] \delta[z, z(u_{!\underline{x}}, x)] \tag{15.55}$$

$$P(z|\mathcal{D}\underline{x} = x) = \sum_{u_{!\underline{x}}} P(u_{!\underline{x}}) \delta[z, z(u_{!\underline{x}}, x)] \ . \tag{15.56}$$

Chapter 16

Do Calculus proofs

In Chapter 15 of Bayesuvius, we explained Do Calculus but referred to this chapter for proofs of claims that use Do Calculus. In this chapter, we've aggregated all proofs, from throughout the book, of claims that use Do Calculus.

Note that even though the 3 rules of Do Calculus are great for proving adjustment formulae for general classes of DAGs, they are sometimes overkill for proving adjustment formulae for a single specific DAG. After all, the 3 rules of Do Calculus are a consequence of the d-separation theorem. Hence, all adjustment formulae should be provable from first principles, assuming only the d-separation theorem and the standard rules of probability theory.

In this chapter, we use the following conventions.

Random variables are underlined and their values are not. For example, $\underline{a} = a$ means the random variable $\underline{a}$ takes the value a. Diagrams with nodes that are underlined represent Bayesian Networks (bnets) and the same diagram with the letters not underlined represents a specific **instantiation** of that bnet. For example $\underline{a} \to \underline{b}$ represents the bnet with conditional probability distribution $P(b|a)$, whereas $a \to b$ represents $P(b|a)$ itself.

If $\underline{a}$ is a root node, then $\sum a$ signifies a weighted sum $\sum_a P(a)$. For example, $\sum a \to b = \sum_a P(a)P(b|a)$. If $\underline{a}$ is not a root node as in $x \to \sum a \to y = \sum_a P(y|a)P(a|x)$, then $\sum a$ signifies a simple unweighted sum $\sum_a$.

Unobserved nodes are indicated by enclosing them in a dashed circle. For example, $\overset{\frown}{\underset{\smile}{u}}$.

Selection diagrams with selection nodes are discussed in Chapter 72. In a selection diagram with a selection node $\underline{s} \in \{0, 1\}$, if a node $\underline{x}$ has parents $pa(\underline{x})$ where $\underline{s} \notin pa(\underline{x})$, then the TPM of $\underline{x}$ is $P(x|pa(x))$. If, on the other hand, $\underline{x}$ has parents $pa(\underline{x}) \cup \underline{s}$, then the TPM of $\underline{x}$ is $P(x|pa(x), s)$, where $P(x|pa(x), \underline{s} = 0) = P(x|pa(x))$ and $P(x|pa(x), \underline{s} = 1) = P^*(x|pa(x))$.

Some identities that are used in this chapter:

1.
$$P(y|x_1, x_2) = \sum_a P(y|a, x_1, x_2)P(a|x_1, x_2) \ . \tag{16.1}$$

$$\begin{tikzcd} x_1 \arrow[dr] & \\ & y \\ x_2 \arrow[ur] & \end{tikzcd} = \begin{tikzcd} x_1 \arrow[dr] \arrow[drr] & & \\ & \sum a \arrow[r] & y \\ x_2 \arrow[ur] \arrow[urr] & & \end{tikzcd} . \tag{16.2}$$

One can describe this identity as "giving $\underline{y}$ a universal backdoor", because $\sum a$ is a backdoor (i.e., input) to y, and $\sum \overline{a}$ is universal in the sense that it is entered by every arrow that enters y except $\sum a$ itself.

2.

$$\sum_a P(a|x_1, x_2) = 1 \tag{16.3}$$

$$\begin{tikzcd} x_1 \arrow[dr] & \\ & \sum a \arrow[r, "0"'] \\ x_2 \arrow[ur] & \end{tikzcd} = 1 \tag{16.4}$$

One can describe this identity as "summing over the values of a collider node which has no emerging arrows"[1]. Eq.(16.4) can be understood as an edge case (when $\underline{y} = \emptyset$) of Eq.(16.2).

3.

$$\sum_a P(x_2|a)P(a|x_1) = P(x_2|x_1) \tag{16.5}$$

$$x_1 \longrightarrow \sum a \longrightarrow x_2 \qquad = \qquad x_1 \longrightarrow x_2 \tag{16.6}$$

One can describe this identity as "summing over the values of a mediator node".

4.

$$P(x) = \sum_a P(x|a)P(a) = \sum_b P(x|b)P(b) \tag{16.7}$$

$$P(x) \qquad = \qquad \xrightarrow[0]{} \sum a \longrightarrow x \qquad = \qquad \xrightarrow[0]{} \sum b \longrightarrow x \tag{16.8}$$

One can describe this identity as "averaging over different priors". Eq.(16.8) can be understood as an edge case of Eq.(16.6).

[1] A zeroed arrow means the same as no arrow.

A **do-adjustment formula** expresses a **do-query** (i.e., a conditional probability with do operators in its condition) by an equivalent expression without do operators. If a do-adjustment formula exists for a particular do-query, then we say the do-query is **do-identifiable**.[2] The following is the simplest of do-adjustment formulae:

$$P(y|\mathcal{D}\underline{x} = x) = P(y|x) \tag{16.9}$$

Many problems (e.g., Backdoor, Frontdoor, etc.) satisfy this simplest of adjustment formulae, but in general we aim to find an adjustment formula that utilizes the full distribution of the observed nodes. Adjustment formulae that don't utilize the full distribution of the observed nodes are throwing away useful info from the dataset, and are less sensitive to deviations from the DAG model being hypothesized.

A **do-transport formula** expresses a do-query in terms of an equivalent do-query.

This chapter deals with both do-adjustment and do-transport formulae.

Claim 21 *(Backdoor Adjustment Formula)*

If $\begin{tikzcd} \underline{z} \arrow[d] \arrow[dr] & \\ \underline{x} \arrow[r] & \underline{y} \end{tikzcd}$ *then*

$$P(y|\mathcal{D}\underline{x} = x) = \sum_z P(y|x,z)P(z) \tag{16.10}$$

$$\mathcal{D}\underline{x} = x \longrightarrow y \quad \begin{tikzcd} \sum z \arrow[dr] & \\ x \arrow[r] & y \end{tikzcd} \tag{16.11}$$

$$=$$

proof:
*** proof 1:**

$P(y|\mathcal{D}\underline{x} = x) = \sum_z P(y|x,z)P(z)$

$$\begin{tikzcd} \sum z \arrow[dr] & \\ \mathcal{D}\underline{x} = x \arrow[r] & y \end{tikzcd} = \begin{tikzcd} \sum z \arrow[dr] & \\ x \arrow[r] & y \end{tikzcd}$$

We can replace $\mathcal{D}\underline{x} = x$ by x because $\mathcal{D}\underline{x} \perp \underline{y}$ in $\mathcal{L}_{\mathcal{D}\underline{x}}G$ so Rule 2 premise is satisfied.

[2] To prove that a do-query $P(y|\mathcal{D}\underline{x} = x, z)$ is do-identifiable for a graph G, just prove that $\underline{y} \perp \underline{x}|\underline{z}$ in $\mathcal{L}_{\underline{x}}G$. This is called Rule 2 of Do Calculus, but it is easy to understand just from the d-separation theorem. Info can be transmitted between $\underline{y}$ and $\underline{x}$ by either (1) paths in $\mathcal{D}_{\underline{x}}G$ or (2) paths in $\mathcal{L}_{\underline{x}}G$. $P(y|\mathcal{D}\underline{x} = x, z) = P(y|x, z)$ means the info is being transmitted only by (1). So the Rule 2 premise is checking that no info is being transmitted by (2).

*** proof 2:**

$P(y|\mathcal{D}\underline{x} = x) = \sum_z P(y|\mathcal{D}\underline{x} = x, z)P(z|\mathcal{D}\underline{x} = x)$

by Probability Axioms

$= \sum_z P(y|x, z)P(z|\mathcal{D}\underline{x} = x)$

$P(y|\mathcal{D}\underline{x} = x, z) \to P(y|x, z)$

by Rule 2: If $(\underline{b}. \perp \underline{a}.|\underline{r}., \underline{s}.)$ in $\mathcal{L}_{\underline{a}.}\mathcal{D}_{\underline{r}.}G$, then $\mathcal{D}\underline{a}. = a. \leftrightarrow \underline{a}. = a.$

$\underline{y} \perp \underline{x}|\underline{z}$ in $\mathcal{L}_{\underline{x}}\mathcal{D}_{\emptyset}G$:

$$\begin{tikzcd} \underline{z} \arrow[d] \arrow[dr] & \\ \underline{x} & \underline{y} \end{tikzcd}$$

$= \sum_z P(y|x, z)P(z)$

$P(z|\mathcal{D}\underline{x} = x) \to P(z)$

by Rule 3: If $(\underline{b}. \perp \underline{a}.|\underline{r}., \underline{s}.)$ in $\mathcal{D}_{\underline{a}.-an(\underline{s}.)}\mathcal{D}_{\underline{r}.}G$, then $\mathcal{D}\underline{a}. = a. \leftrightarrow 1$

$\underline{z} \perp \underline{x}$ in $\mathcal{D}_{\underline{x}}\mathcal{D}_{\emptyset}G$:

$$\begin{tikzcd} \underline{z} \arrow[dr] & \\ \underline{x} \arrow[r] & \underline{y} \end{tikzcd}$$

QED

Claim 22 *(Frontdoor Adjustment Formula)*

If

$$\begin{tikzcd} & \underline{c} \arrow[dl] \arrow[dr] & \\ \underline{x} \arrow[r] & \underline{m} \arrow[r] & \underline{y} \end{tikzcd}$$

then

$$P(y|\mathcal{D}\underline{x} = x) = \sum_m \left[\sum_{x'} P(y|x', m)P(x') \right] P(m|x) \tag{16.12}$$

$$\mathcal{D}\underline{x} = x \longrightarrow y \qquad = \begin{tikzcd} & \sum x' \arrow[dr] & \\ x \arrow[r] & \sum m \arrow[r] & y \end{tikzcd} \tag{16.13}$$

proof:

*** proof 1:**

$P(y|\mathcal{D}\underline{x} = x) = \sum_{m,c,x'} P(y|m, c)P(c|x')P(x')P(m|\mathcal{D}\underline{x} = x)$

$$\begin{tikzcd}
 & \sum c \arrow[dr] & \\
\mathcal{D}\underline{x} = x \arrow[r] & \sum m \arrow[r] & y
\end{tikzcd}
=
\begin{tikzcd}
\sum x' \arrow[r] & \sum c \arrow[dr] & \\
\mathcal{D}\underline{x} = x \arrow[r] & \sum m \arrow[r] & y
\end{tikzcd}$$

$= \sum_{m,x'} P(y|m,x')P(x')P(m|x)$

We can replace $\mathcal{D}\underline{x} = x$ by x because $\mathcal{D}\underline{x} \perp \underline{m}$ in $\mathcal{L}_{\mathcal{D}\underline{x}}G$ so Rule 2 premise is satisfied.

$$= \begin{tikzcd}
 & \sum x' \arrow[dr] & \\
x \arrow[r] & \sum m \arrow[r] & y
\end{tikzcd}$$

*** proof 2:**

$P(y|\mathcal{D}\underline{x} = x) = \sum_m P(y|\mathcal{D}\underline{x} = x, m)P(m|\mathcal{D}\underline{x} = x)$

by Probability Axioms

$= \sum_m P(y|\mathcal{D}\underline{x} = x, \mathcal{D}\underline{m} = m)P(m|\mathcal{D}\underline{x} = x)$

$P(y|\mathcal{D}\underline{x} = x, m) \to P(y|\mathcal{D}\underline{x} = x, \mathcal{D}m = m)$

by Rule 2: If $(\underline{b}. \perp \underline{a}.|\underline{r}., \underline{s}.)$ in $\mathcal{L}_{\underline{a}.}\mathcal{D}_{\underline{r}.}G$, then $\mathcal{D}\underline{a}. = a. \leftrightarrow \underline{a}. = a.$

$\underline{y} \perp \underline{m}|\underline{x}$ in $\mathcal{L}_{\underline{m}}\mathcal{D}_{\underline{x}}G$:

$$\begin{tikzcd}
 & \underline{c} \arrow[dr] & \\
\underline{x} \arrow[r] & \underline{m} & \underline{y}
\end{tikzcd}$$

$= \sum_m P(y|\mathcal{D}\underline{x} = x, \mathcal{D}\underline{m} = m)P(m|x)$

$P(m|\mathcal{D}\underline{x} = x) \to P(m|x)$

by Rule 2: If $(\underline{b}. \perp \underline{a}.|\underline{r}., \underline{s}.)$ in $\mathcal{L}_{\underline{a}.}\mathcal{D}_{\underline{r}.}G$, then $\mathcal{D}\underline{a}. = a. \leftrightarrow \underline{a}. = a.$

$\underline{m} \perp \underline{x}$ in $\mathcal{L}_{\underline{x}}\mathcal{D}_{\emptyset}G$:

$$\begin{tikzcd}
 & \underline{c} \arrow[dl] \arrow[dr] & \\
\underline{x} & \underline{m} \arrow[r] & \underline{y}
\end{tikzcd}$$

$= \sum_m P(y|\mathcal{D}\underline{m} = m)P(m|x)$

$P(y|\mathcal{D}\underline{x} = x, \mathcal{D}\underline{m} = m) \to P(y|\mathcal{D}\underline{m} = m)$

by Rule 3: If $(\underline{b}. \perp \underline{a}.|\underline{r}., \underline{s}.)$ in $\mathcal{D}_{\underline{a}.-an(\underline{s}.)}\mathcal{D}_{\underline{r}.}G$, then $\mathcal{D}\underline{a}. = a. \leftrightarrow 1$

$\underline{y} \perp \underline{x}|\underline{m}$ in $\mathcal{D}_{\underline{x}}\mathcal{D}_{\underline{m}}G$:

$$\begin{tikzcd}
 & \underline{c} \arrow[dr] & \\
\underline{x} & \underline{m} \arrow[r] & \underline{y}
\end{tikzcd}$$

$= \sum_{x'} \sum_m P(y|\mathcal{D}\underline{m} = m, x')P(x'|\mathcal{D}\underline{m} = m)P(m|x)$

by Probability Axioms

$= \sum_{x'} \sum_{m} P(y|m,x')P(x'|\mathcal{D}\underline{m}=m)P(m|x)$

$P(y|\mathcal{D}\underline{m}=m,x') \rightarrow \quad P(y|m,x')$

by Rule 2: If $(\underline{b}. \perp \underline{a}.|\underline{r}., \underline{s}.)$ in $\mathcal{L}_{\underline{a}.}\mathcal{D}_{\underline{r}.}G$, then $\mathcal{D}\underline{a}. = a. \leftrightarrow \underline{a}. = a.$

$\underline{y} \perp \underline{m}|\underline{x}$ in $\mathcal{L}_{\underline{m}}\mathcal{D}_{\emptyset}G$:

$$\begin{tikzcd} & \underline{c} \arrow[dl] \arrow[dr] & \\ \underline{x} \arrow[r] & \underline{m} & \underline{y} \end{tikzcd}$$

$= \sum_{x'} \sum_{m} P(y|m,x')P(x')P(m|x)$

$P(x'|\mathcal{D}\underline{m}=m) \rightarrow P(x')$

by Rule 3: If $(\underline{b}. \perp \underline{a}.|\underline{r}., \underline{s}.)$ in $\mathcal{D}_{\underline{a}.-an(\underline{s}.)}\mathcal{D}_{\underline{r}.}G$, then $\mathcal{D}\underline{a}. = a. \leftrightarrow 1$

$\underline{x} \perp \underline{m}$ in $\mathcal{D}_{\underline{m}}\mathcal{D}_{\emptyset}G$:

$$\begin{tikzcd} & \underline{c} \arrow[dl] \arrow[dr] & \\ \underline{x} & \underline{m} \arrow[r] & \underline{y} \end{tikzcd}$$

QED

Claim 23 *(Napkin problem from Ref.[41])*

If

$$\begin{tikzcd} & \underline{u}_1 \arrow[dl] \arrow[ddrr] & & \\ & \underline{u}_2 \arrow[dl] \arrow[dr] & & \\ \underline{w} \arrow[r] & \underline{z} \arrow[r] & \underline{x} \arrow[r] & \underline{y} \end{tikzcd}$$

then

$$P(y|\mathcal{D}\underline{x}=x) = \sum_{w,z} P(y|x,w,z)P(w,z) \tag{16.14}$$

$$\mathcal{D}\underline{x} = x \longrightarrow y \quad = \quad \begin{tikzcd} & \sum w,z \arrow[dr] & \\ x \arrow[rr] & & y \end{tikzcd} \tag{16.15}$$

proof:

$P(y|\mathcal{D}\underline{x}=x) = \sum_{u_1} P(y|\mathcal{D}\underline{x}=x,u_1)P(u_1)$

$$
\begin{tikzcd}
 & \sum u_1 \arrow[dl] \arrow[drr] & & \\
 & \sum u_2 \arrow[dl] & & \\
\sum w \arrow[r] & \sum z & \mathcal{D}\underline{x} = x \arrow[r] & \underline{y}
\end{tikzcd}
\quad = \quad
\begin{tikzcd}
\sum u_1 \arrow[dr] & \\
\mathcal{D}\underline{x} = x \arrow[r] & y
\end{tikzcd}
$$

$$= \sum_{w,z} \sum_{u_1} P(y|\mathcal{D}\underline{x} = x, u_1) P(u_1|w,z) P(w,z)$$

$$
\begin{tikzcd}
\sum w, z \arrow[r] & \sum u_1 \arrow[dr] & \\
 & & \mathcal{D}\underline{x} = x \arrow[r] & y
\end{tikzcd}
$$

$$= \sum_{w,z} P(y|x,w,z) P(w,z)$$

We can replace $\mathcal{D}\underline{x} = x$ by x because $\mathcal{D}\underline{x} \perp \underline{y}$ in $\mathcal{L}_{\mathcal{D}\underline{x}} G$ so Rule 2 premise is satisfied.

$$
\begin{tikzcd}
\sum w, z \arrow[dr] & \\
x \arrow[r] & y
\end{tikzcd}
$$

QED

Claim 24 *(from Ref.[41])*

If

$$
\begin{tikzcd}
\underline{x} \arrow[d] \arrow[drr] & & \\
\underline{w} \arrow[r] & \underline{z} \arrow[r] & \underline{y} \\
\underline{u} \arrow[u] \arrow[urr] & &
\end{tikzcd}
$$

then

$$P(y|\mathcal{D}\underline{z} = z, x) = \sum_{w} P(y|z,x,w) P(w) \tag{16.16}$$

$$
\begin{tikzcd}
x \arrow[drr] & & \\
 & \mathcal{D}\underline{z} = z \arrow[r] & y
\end{tikzcd}
\quad = \quad
\begin{tikzcd}
x \arrow[drr] & & \\
\sum w \arrow[rr, bend right] & z \arrow[r] & y
\end{tikzcd}
\tag{16.17}
$$

proof:

$P(y|\mathcal{D}\underline{z}=z,x) = \sum_u P(y|\mathcal{D}\underline{z}=z,x,u)P(u)$

$= \sum_w \sum_u P(y|\mathcal{D}\underline{z}=z,x,u)P(u|w)P(w)$

$= \sum_w P(y|z,x,w)P(w)$

We can replace $\mathcal{D}\underline{z}=z$ by z because $\mathcal{D}\underline{z} \perp \underline{y}$ in $\mathcal{L}_{\mathcal{D}\underline{z}}G$ so Rule 2 premise is satisfied.

QED

Claim 25 *(Trivial Memoryless Transportability, from Ref.[39])*
If

where $\underline{s} \in \{0,1\}$ *is a selection node, then*

$$P^*(y|\mathcal{D}\underline{x}=x,z) = P^*(y|x,z) \quad \textit{(replace } \mathcal{D} \textit{ by } 1\textit{, keep } P^*\textit{)} \tag{16.18}$$

(16.19)

proof:

$$P(y|\mathcal{D}\underline{x} = x, z, \underline{s} = 1) = P(y|x, z, \underline{s} = 1)$$

$$\begin{tikzcd}
 & z \arrow[dr] & \underline{s} = 1 \arrow[d] \\
\mathcal{D}\underline{x} = x \arrow[rr] & & y
\end{tikzcd}
=
\begin{tikzcd}
 & z \arrow[dr] & \underline{s} = 1 \arrow[d] \\
x \arrow[rr] & & y
\end{tikzcd}$$

QED

Claim 26 *(Direct Transportability, a.k.a. External Validity, from Ref.[39])*
If $\begin{tikzcd}[rr, bend left] \underline{s} \arrow[r] \arrow[rr, dashed, bend left] & \underline{z} \arrow[dl] \arrow[dr] & \\ \underline{x} \arrow[rr] & & \underline{y} \arrow[ll, dashed, bend left] \end{tikzcd}$ *where* $\underline{s} \in \{0, 1\}$ *is a selection node, then*

$$P^*(y|\mathcal{D}\underline{x} = x, z) = P(y|\mathcal{D}\underline{x} = x, z) \quad \textit{(replace } P^* \textit{ by } P\textit{, keep } \mathcal{D}\textit{)} \tag{16.20}$$

$$\begin{tikzcd}
\underline{s} = 1 \arrow[r] & z \arrow[dr] & \\
\mathcal{D}\underline{x} = x \arrow[rr] & & y
\end{tikzcd}
=
\begin{tikzcd}
 & z \arrow[dr] & \\
\mathcal{D}\underline{x} = x \arrow[rr] & & y
\end{tikzcd} \tag{16.21}$$

Furthermore,

$$P^*(y|\mathcal{D}\underline{x} = x) = \sum_z P(y|\mathcal{D}\underline{x} = x, z) P^*(z) \tag{16.22}$$

$$\begin{tikzcd}
\underline{s} = 1 \arrow[drr] & & \\
\mathcal{D}\underline{x} = x \arrow[rr] & & y
\end{tikzcd}
=
\begin{tikzcd}
\underline{s} = 1 \arrow[r] & \sum z \arrow[dr] & \\
\mathcal{D}\underline{x} = x \arrow[rr] & & y
\end{tikzcd} \tag{16.23}$$

proof:

$$P(y|\mathcal{D}\underline{x} = x, z, \underline{s} = 1) = P(y|\mathcal{D}\underline{x} = x, z)$$

$$\begin{tikzcd}
\underline{s} = 1 \arrow[r] & z \arrow[dr] & \\
\mathcal{D}\underline{x} = x \arrow[rr] & & y
\end{tikzcd}
=
\begin{tikzcd}
 & z \arrow[dr] & \\
\mathcal{D}\underline{x} = x \arrow[rr] & & y
\end{tikzcd}$$

Because $\underline{s} \perp \underline{y}|\underline{z}$

Furthermore,

$$P(y|\mathcal{D}\underline{x} = x, \underline{s} = 1) = \textstyle\sum_z P(y|\mathcal{D}\underline{x} = x, z) P(z|\underline{s} = 1)$$

$$\begin{tikzcd}
\underline{s} = 1 \arrow[r] & \sum z \arrow[dr] & \\
\mathcal{D}\underline{x} = x \arrow[rr] & & y
\end{tikzcd}$$

QED

Claim 27 *(S-Admisssible Transportability, from Ref.[39])*

If $\begin{tikzcd}
\underline{s} \arrow[r] & \underline{z} \arrow[r] \arrow[d] & \underline{a} \arrow[d] \\
\underline{x} \arrow[rr] & & \underline{y} \arrow[ll, bend left, dashed]
\end{tikzcd}$ *where* $\underline{s} \in \{0, 1\}$ *is a selection node, then*

$$P^*(y|\mathcal{D}\underline{x} = x) = \sum_a P(y|\mathcal{D}\underline{x} = x, a)P^*(a) \tag{16.24}$$

$$\begin{tikzcd}
\underline{s} = 1 \arrow[dr] & \\
\mathcal{D}\underline{x} = x \arrow[r] & y
\end{tikzcd}
=
\begin{tikzcd}
\underline{s} = 1 \arrow[r] & \sum a \arrow[d] \\
\mathcal{D}\underline{x} = x \arrow[r] & y
\end{tikzcd} \tag{16.25}$$

proof:

$$P(y|\mathcal{D}\underline{x} = x, \underline{s} = 1) = \textstyle\sum_a P(y|\mathcal{D}\underline{x} = x, a)P(a|\underline{s} = 1)$$

$$\begin{tikzcd}
\underline{s} = 1 \arrow[r] & \sum z \arrow[r] & \sum a \arrow[d] \\
 & \mathcal{D}\underline{x} = x \arrow[r] & y
\end{tikzcd}
=
\begin{tikzcd}
\underline{s} = 1 \arrow[r] & \sum a \arrow[d] \\
\mathcal{D}\underline{x} = x \arrow[r] & y
\end{tikzcd}$$

QED

Claim 28 *(Non-transportability, from Ref.[39])*

If $\begin{tikzcd}
 & \underline{h} \arrow[dl] \arrow[dr] & \underline{s} \arrow[d] \\
\underline{x} \arrow[rr] & & \underline{y}
\end{tikzcd}$ *where* $\underline{s} \in \{0, 1\}$ *is a selection node, then*

$$P^*(y|\mathcal{D}\underline{x} = x) = P^*(y|\mathcal{D}\underline{x} = x) \tag{16.26}$$

$$\begin{tikzcd} & \underline{s} = 1 \arrow[d] \\ \mathcal{D}\underline{x} = x \arrow[r] & y \end{tikzcd} \quad = \textit{same} \tag{16.27}$$

proof:

$$P^*(y|\mathcal{D}\underline{x} = x) = P^*(y|\mathcal{D}\underline{x} = x)$$

$$\begin{tikzcd} & \sum h \arrow[dr] & \underline{s} = 1 \arrow[d] \\ \mathcal{D}\underline{x} = x \arrow[rr] & & y \end{tikzcd} \quad = \quad \begin{tikzcd} & & \underline{s} = 1 \arrow[d] \\ \mathcal{D}\underline{x} = x \arrow[rr] & & y \end{tikzcd}$$

Can't replace $\mathcal{D}\underline{x} = x$ by x because $\underline{y} \not\perp \underline{x}$ in $\mathcal{L}_{\underline{x}}G$. Hence, Rule 2 not satisfied.
QED

Claim 29 *(from Ref.[39])*

If $\begin{tikzcd} \underline{s} \arrow[rr, bend left] \arrow[dr, dashed] & & \underline{h} \arrow[ll, dashed] \arrow[r] \arrow[dl] \arrow[dr] & \underline{z} \\ & \underline{x} \arrow[rr] & & \underline{y} \arrow[ll, dashed, bend left] \end{tikzcd}$ *where* $\underline{s} \in \{0, 1\}$ *is a selection node, then*

$$P^*(y|\mathcal{D}\underline{x} = x) = P(y|\mathcal{D}\underline{x} = x) \tag{16.28}$$

$$\begin{tikzcd} & \underline{s} = 1 \arrow[dr] & \\ \mathcal{D}\underline{x} = x \arrow[rr] & & y \end{tikzcd} \quad = \quad \mathcal{D}\underline{x} = x \longrightarrow y \tag{16.29}$$

proof:

$$P(y|\mathcal{D}\underline{x} = x, \underline{s} = 1) = \textstyle\sum_h P(y|\mathcal{D}\underline{x} = x, h)P(h)$$

$$\begin{tikzcd} \underline{s} = 1 \arrow[rr, bend left] & \sum h \arrow[r] \arrow[dr] & \sum z \\ \mathcal{D}\underline{x} = x \arrow[rr] & & y \end{tikzcd} \quad = \quad \begin{tikzcd} & \sum h \arrow[dr] & \\ \mathcal{D}\underline{x} = x \arrow[rr] & & y \end{tikzcd}$$

$$= P(y|\mathcal{D}\underline{x} = x)$$

$$= \mathcal{D}\underline{x} = x \longrightarrow y$$

QED

Claim 30 *(from Ref.[39])*

If $\underline{s}$ *where* $\underline{s} \in \{0,1\}$ *is a selection node, then*

$$P^*(y|\mathcal{D}\underline{x}=x) = \sum_z P(y|\mathcal{D}\underline{x}=x,z)P^*(z|x) \tag{16.30}$$

(16.31)

proof:

$P(y|\mathcal{D}\underline{x}=x,\underline{s}=1) = \sum_h \sum_z P(y|h,z)P(h)P(z|\mathcal{D}\underline{x}=x,\underline{s}=1)$

$= \sum_h \sum_z P(y|h,z)P(h|\mathcal{D}\underline{x}=x)P(z|x,\underline{s}=1)$

$= \sum_z P(y|\mathcal{D}\underline{x}=x,z)P(z|x,\underline{s}=1)$

QED

Claim 31 *(Unconfounded Mediation, from Ref.[38])*

If *then*

$$\begin{tikzcd} & \underline{m} \arrow[dr] & \\ \underline{d} \arrow[ur] \arrow[rr] & & \underline{y} \end{tikzcd}$$

$$P(y|\mathcal{D}\underline{d} = d, \mathcal{I}_{\underline{m}}\underline{d} = d') = \sum_m P(y|d,m)P(m|d') \tag{16.32}$$

$$\begin{tikzcd} \mathcal{I}\underline{d} = d' \arrow[dr] & \\ \mathcal{D}\underline{d} = d \arrow[r] & y \end{tikzcd} = \begin{tikzcd} \mathcal{I}\underline{d} = d' \arrow[r] & \sum m \arrow[dr] & \\ \mathcal{D}\underline{d} = d \arrow[rr] & & y \end{tikzcd} \tag{16.33}$$

proof:

$$P(y|\mathcal{D}\underline{d} = d, \mathcal{I}\underline{d} = d') = \textstyle\sum_m P(y|d,m)P(m|d')$$

$$\begin{tikzcd} \mathcal{I}\underline{d} = d' \arrow[r] & \sum m \arrow[dr] & \\ \mathcal{D}\underline{d} = d \arrow[rr] & & y \end{tikzcd}$$

QED

Claim 32 *(Mediation with universal prior $\underline{\xi}$ and universal confounder $\underline{u}$, from Ref.[38])*

If *then*

$$\begin{tikzcd} & \underline{u} \arrow[dl] \arrow[d] \arrow[ddr] \arrow[ddrr] & & \\ \underline{\xi} \arrow[r] \arrow[d] \arrow[drr] & \underline{m} \arrow[dr] & & \\ \underline{d} \arrow[ur] \arrow[rr] & & \underline{y} & \end{tikzcd}$$

$$P(y|\mathcal{D}\underline{d} = d, \mathcal{I}_{\underline{m}}\underline{d} = d') = \sum_\xi \sum_m P(y|d,m,\xi)P(m|d',\xi)P(\xi) \tag{16.34}$$

$$\begin{tikzcd} \mathcal{I}\underline{d} = d' \arrow[dr] & \\ \mathcal{D}\underline{d} = d \arrow[r] & y \end{tikzcd} = \begin{tikzcd} \mathcal{I}\underline{d} = d' \arrow[rr, bend right] & & \sum \xi \arrow[r] \arrow[ddr] & \sum m \arrow[dr] & \\ \mathcal{D}\underline{d} = d \arrow[rrr] & & & & y \end{tikzcd} \tag{16.35}$$

proof:

$$P(y|\mathcal{D}\underline{d}=d,\mathcal{I}\underline{d}=d') = \sum_{\xi,u}\sum_m P(y|d,m,\xi,u)P(m|d',\xi,u)\underbrace{P(\xi|u)P(u)}_{P(\xi,u)}$$

$$\begin{tikzcd}
 & & \left(\sum u\right) \arrow[dl] \arrow[d] \arrow[ddr] & \\
\mathcal{I}\underline{d}=d' \arrow[rr, bend right] & \sum \xi \arrow[r] \arrow[drr] & \sum m \arrow[dr] & \\
\mathcal{D}\underline{d}=d \arrow[rrr] & & & y
\end{tikzcd}$$

$$= \sum_\xi \sum_m P(y|d,m,\xi)P(m|d',\xi)P(\xi)$$

$$=\begin{tikzcd}
\mathcal{I}\underline{d}=d' \arrow[rr, bend right] & \sum \xi \arrow[r] \arrow[drr] & \sum m \arrow[dr] & \\
\mathcal{D}\underline{d}=d \arrow[rrr] & & & y
\end{tikzcd}$$

We switch from averaging over the prior of ξ, u to averaging over the prior of ξ.

QED

Chapter 17

D-Separation

Before reading this chapter, I recommend that you read Chapter Definition of a Bayesian Network.

A path γ that isn't a loop can have 3 types of intermediate nodes $\underline{x}$ (an intermediate node of γ is a node in γ that isn't one of the two end nodes). Suppose $\underline{a}, \underline{b} \in \gamma$ are the two neighbors of $\underline{x}$. Then the 3 possible cases are:

1. $\underline{x}$ **is a mediator node:** $(\underline{a} \leftarrow \underline{x} \leftarrow \underline{b})$ or $(\underline{a} \rightarrow \underline{x} \rightarrow \underline{b})$
2. $\underline{x}$ **is a fork node:** $(\underline{a} \leftarrow \underline{x} \rightarrow \underline{b})$
3. $\underline{x}$ **is a collider node:** $(\underline{a} \rightarrow \underline{x} \leftarrow \underline{b})$

We say that a non-loop path γ from $\underline{a}$ to $\underline{b}$ (i.e., with end nodes $\underline{a}, \underline{b}$) is **blocked** by conditioning on a multinode $\underline{Z}.$ if one or more of the following statements is true:

1. There is a node $\underline{x} \in \underline{Z}.$ which is a mediator or a fork of γ.
2. γ contains a collider node $\underline{c}$ and $(\underline{c} \cup de(\underline{c})) \cap \underline{Z}. = \emptyset$ (i.e., neither $\underline{c}$ nor any of the descendants of $\underline{c}$ is contained in $\underline{Z}.$)

This definition of a blocked path[1] is easy to remember if one thinks of the following analogy with pipes carrying a fluid. Think of path γ as if it were a pipe carrying a fluid. Think of the nodes of γ as junctions in the pipe. If $\underline{Z}.$ intersects γ at either a mediator or a fork junction, that blocks the pipe flow. A collider junction $\underline{c}$ is like a blackhole or a huge leak. Its presence blocks passage of the fluid as long as neither $\underline{c}$ nor any of the descendants of $\underline{c}$ are in $\underline{Z}.$. If, on the other hand, $\underline{c} \in \underline{Z}.$, or $\underline{c}' \in \underline{Z}.$ where $\underline{c}' \in de(\underline{c})$, then that acts as a complete (in the case of $\underline{c} \in \underline{Z}.$) or a partial (in the case of $\underline{c}' \in \underline{Z}.$) bridge across the blackhole.

See Fig.17.1 for some examples of paths that are blocked or not blocked by conditioning on a multinode $\underline{Z}.$.

[1] Note that we speak of blocked paths or info, not of blocked nodes. Nodes are not blocked; rather they are either conditioned upon or not.

Not Blocked	(17.1)
Blocked	(17.2)
Not Blocked	(17.3)
Blocked	(17.4)
Blocked	(17.5)
Not Blocked	(17.6)
Not Blocked	(17.7)

Figure 17.1: Examples of paths that are blocked or not blocked by conditioning on a multinode $\underline{Z}.$. Nodes belonging to $\underline{Z}.$ are colored yellow.

Given 3 disjoint multinodes $\underline{A}., \underline{B}., \underline{Z}.$ of a graph G, we write " $\underline{A}. \perp_G \underline{B}.|\underline{Z}.$" or say " $\underline{A}.$ **and** $\underline{B}.$ **are d-separated by** $\underline{Z}.$ **in** G" iff there exists no path γ from $\underline{a} \in \underline{A}.$, to $\underline{b} \in \underline{B}.$ which is not blocked by conditioning on $\underline{Z}.$.[2]

The minimal Markov blanket (see Chapter 37) of a node $\underline{a}$ is the smallest multinode $\underline{Z}.$ such that $\underline{a} \perp_G \underline{b}|\underline{Z}.$ for all $\underline{b} \notin \underline{a} \cup \underline{Z}.$.

We are finally ready to state the d-separation theorem, without proof.

A probability distribution P **is compatible with a DAG** G if P and G have the same random variables, and they can be combined to form a bnet without contradictions; i.e., one can calculate all the TPMs from P and multiply them together to obtain P again.

Claim 33 *(d-separation Theorem)*

Suppose $\underline{A}., \underline{B}., \underline{Z}.$ are disjoint multinodes of a DAG G.

If $\underline{A}. \perp_G \underline{B}.|\underline{Z}.$, then $P(B.|A., Z.) = P(B.|Z.)$ for all $B., A., Z.$, for all P compatible with G.

The full converse of the theorem can also be proven, but we won't be using it in this book.

Often, the right hand side of this theorem is stated as "$\underline{A}. \perp_P \underline{B}.|\underline{Z}.$ for all P". Then the theorem is stated: "If $\underline{A}. \perp_G \underline{B}.|\underline{Z}.$, then $\underline{A}. \perp_P \underline{B}.|\underline{Z}.$ for all P."

Note that the following are equivalent:

- $P(B.|A., Z.) = P(B.|Z.)$ for all $B., A., Z.$.
- $\underline{A}. \perp_P \underline{B}.|\underline{Z}.$

[2] $\underline{Z}.$ are the nodes we are "conditioning on". Unmeasured (i.e., hidden, unobserved) nodes cannot be conditioned on, because that would entail measuring them.

- $H(\underline{A}. : \underline{B}.|\underline{Z}.) = 0$ (see Chapter Notational Conventions and Preliminaries for definition of conditional mutual information (CMI))

Extra stuff: mostly only for pure mathematicians

Below, we will use the notation $nde(\underline{a})$ to denote all non-descendants, including $\underline{a}$ itself, of a node $\underline{a}$ in a DAG G; i.e., all nodes of G that are not in $de(\underline{a}) \cup \underline{a}$, where $de(\underline{a})$ is defined in Chapter Definition of a Bayesian Network.

Given a DAG G, define the following sets of d-separations:[3]

$$DS(G) = \{(\underline{A}. \perp_G \underline{B}. \mid \underline{Z}.) : \ \underline{A}., \underline{B}., \underline{Z}. \text{ are multinodes of } G\} \ . \tag{17.8}$$

$$DS_{min}(G) = \{(\underline{A}. \perp_G nde(\underline{A}.) \mid pa(\underline{A}.)) : \ \underline{A}. \text{ is a multinode of } G\} \ . \tag{17.9}$$

See Chapter 51 for an example where set $DS_{min}(G)$ is calculated for a particular DAG G.

Claim 34 *For all DAGs G, $DS(G) = DS_{min}(G)$.*

Given a probability distribution P, define the following set of conditional independencies:

$$CI(P) = \{(\underline{A}. \perp_P \underline{B}. \mid \underline{Z}.) : \ \underline{A}., \underline{B}., \underline{Z}. \text{ are multinodes of } P\} \ , \tag{17.10}$$

For a DAG G and a probability distribution P compatible with G, define a map ϕ by

$$\phi : DS_{min}(G) \ \rightarrow \ CI(P) \tag{17.11}$$

$$\phi : \underline{A}. \perp_G nde(\underline{A}.) \mid pa(\underline{A}.) \ \mapsto \ \underline{A}. \perp_P nde(\underline{A}.) \mid pa(\underline{A}.) \tag{17.12}$$

In general, this map is 1-1 but not onto.

Claim 35 *For a bnet with a DAG G and a total probability distribution P, the map ϕ is a bijection.*

$DS(G)$ does not fully specify a DAG. DAGs with the same $DS(G)$ are said to be **d-separation equivalent**. See Chapter 51 for more info about d-separation equivalence.

---

[3] Note that $(\underline{A}. \perp_G nde(\underline{A}.) \mid pa(\underline{A}.))$ and $(\underline{A}. \perp_G nde(\underline{A}.) - pa(\underline{A}.) \mid pa(\underline{A}.))$ are equivalent because $H(\underline{a} : \underline{b}, \underline{c}|\underline{c}) = H(\underline{a} : \underline{b}|\underline{c})$.

Chapter 18

D-Separation and Do Calculus Rules for LDEN: COMING SOON

In this chapter, we will prove the D-Separation Theorem (see Chapter 17) and the Do Calculus Rules (see Chapter 15) for the special bnets that we call LDEN(see Chapter 36). Proving these theorems for LDEN rather than for general bnets is much simpler and quite instructive.

Chapter 19

D-Separation in Quantum Mechanics

See Ref.[56].

Chapter 20

Dynamic Bayesian Networks

Figure 20.1: Example of a dynamic bnet. The pattern of red arrows is repeated $T-1$ times.

A dynamic bnet is simply a time homogeneous Markov chain (see Chapter 39) for which each node is called a **time slice**, and each time slice represents at finer resolution a sub-DAG which is the same between any 2 successive time slices. Fig.20.1 gives an example of a dynamic bnet. In Fig.20.1, we've drawn the 3 nodes of each time slice vertically, and labeled them with a superscript $.^{(t)}$, where $t \in \{0, 1 \ldots, T-1\}$ is the time of the slice. To fully specify the dynamic bnet of Fig.20.1, we would also have to specify the TPMs

$P(c^{(0)})$,
$P(b^{(0)})$
$P(c^{(1)}|c^{(0)})$,
$P(b^{(1)}|b^{(0)}, a^{(1)})$
$P(a^{(1)}|b^{(0)})$.

Dynamic bnets are very common in AI and Data Science. Kalman filters (Chapter 34), Hidden Markov Models (Chapter 28) and Recurrent Neural Networks (Chapter 55) are famous examples of dynamic bnets.

Bnets are acyclic; they can't have cycles (i.e, closed directed paths). Yet feedback loops are an important concept in Science. So what is the equivalent of feedback loops in the bnet world? Dynamic bnets are. Fig.20.2 represents Fig.20.1

more compactly using feedback loops. Any bnet with feedback loops can be "unrolled" into a dynamic bnet.

Figure 20.2: Dynamic bnet Fig.20.1 represented more compactly using feedback loops. Arrows labelled + point from nodes of the t time slice to nodes of the $t+1$ time slice.

Chapter 21

Expectation Maximization

This chapter is based on Refs.[81] and [132].

The Expectation Maximization (EM) algorithm is commonly used in Data Science to find the maximum over an **unknown parameter** θ of a likelihood function

$$P(\vec{x}|\theta) = \sum_{\vec{h}} P(\vec{x}, \vec{h}|\theta) \ , \tag{21.1}$$

where $\vec{x}$ denotes the **observed variables**, and $\vec{h}$ denotes the **latent variables**. Both θ and $\vec{h}$ are hidden (i.e., unobserved).[1]

Figure 21.1: bnet for EM with $nsam = 3$.

The bnet for the EM algorithm is given by Fig.21.1 for $nsam = 3$. Later on in this chapter, we will give the node TPMs for this bnet for the special case in which $P(x[\sigma] \mid \theta)$ is a mixture (i.e., weighted sum) of Gaussians.

[1] The term "unknown parameter" is mainly of frequentist origin. For Bayesians, θ is a random variable with a delta function prior, whereas for frequentists, it is not a random variable at all, just an unknown parameter with no randomness.

Note that if we erase the $\underline{h}[\sigma]$ nodes from Fig.21.1, we get the bnet for naive Bayes, which is used for classification into the states of $\underline{\theta}$. However, there is one big difference. With naive Bayes, the leaf nodes have different TPMs. Here, we will assume they are i.i.d. Naive Bayes is used for classification: i.e., given the states of the leaf nodes, we infer the state of the root node. EM is used for clustering; i.e., given many i.i.d. samples, we fit their distribution by a weighted sum of prob distributions, usually Gaussians.

Let

$\mathcal{L}$ =likelihood function.

$nsam$ = number of samples.

$\vec{x} = (x[0], x[1], \ldots, x[nsam-1])$ $x[\sigma] \in S_{\underline{x}}$ for all σ.

$\vec{h} = (h[0], h[1], \ldots, h[nsam-1])$ $h[\sigma] \in S_{\underline{h}}$ for all σ.

We assume that the samples $(x[\sigma], h[\sigma])$ are i.i.d. for different σ at fixed θ. What this means is that there are probability distributions $P_{\underline{x}|\underline{h},\underline{\theta}}$ and $P_{\underline{h}|\underline{\theta}}$ such that

$$P(\vec{x}, \vec{h}|\theta) = \prod_{\sigma} \left[P_{\underline{x}|\underline{h},\underline{\theta}}(x[\sigma] \mid h[\sigma], \theta) P_{\underline{h}|\underline{\theta}}(h[\sigma] \mid \theta) \right] \ . \tag{21.2}$$

Definition of likelihood functions:

$$\underbrace{P(\vec{x}|\theta)}_{\mathcal{L}(\theta;\vec{x})} \ = \ \sum_{\vec{h}} \underbrace{P(\vec{x}, \vec{h}|\theta)}_{\mathcal{L}(\theta;\vec{x},\vec{h})} \tag{21.3}$$

θ^* = maximum likelihood estimate of θ (no prior $P(\theta)$ assumed):

$$\theta^* = \operatorname*{argmax}_{\theta} \mathcal{L}(\theta; \vec{x}) \tag{21.4}$$

21.1 The EM algorithm:

1. **Expectation step:**[2]

$$Q(\theta|\theta^{(t)}) = E_{\vec{h}|\vec{x},\theta^{(t)}} \ln P(\vec{x}, \vec{h}|\theta) \tag{21.5}$$

2. **Maximization step:**

$$\theta^{(t+1)} = \operatorname*{argmax}_{\theta} Q(\theta|\theta^{(t)}) \ . \tag{21.6}$$

Claim: $\lim_{t\to\infty} \theta^{(t)} = \theta^*$.

[2] Note that that the right hand side of Eq.(21.5) is expressible in the form $\sum_\sigma \sum_{h[\sigma]} f(x[\sigma], h[\sigma])$.

Figure 21.2: The EM algo generates a sequence of parameter estimates $(\theta^{(t)})_{t=0,1,2,\ldots}$ that converges to the optimum (i.e., best-fit) parameter θ^*.

Fig.21.2 portrays the recursive nature of the EM algo as a dynamical, recurrent bnet. For Fig.21.2, the TPMs, printed in blue, for the $\underline{\theta}^{(t)}$ nodes for $t = 1, 2, \ldots$, are as follows:

$$P(\theta^{(t+1)}|\vec{x}, \theta^{(t)}) = \delta(\theta^{(t+1)}, \operatorname*{argmax}_{\theta} Q(\theta|\theta^{(t)}))\ . \tag{21.7}$$

21.1.1 Motivation

$$\begin{aligned} Q(\theta|\theta^{(t)}) &= E_{\vec{h}|\vec{x},\theta^{(t)}} \ln P(\vec{x}, \vec{h}|\theta) && (21.8)\\ &= E_{\vec{h}|\vec{x},\theta^{(t)}} [\ln P(\vec{h}|\vec{x}, \theta) + \ln P(\vec{x}|\theta)] && (21.9)\\ &= -D_{KL}\left(P(\vec{h}|\vec{x}, \theta^{(t)}) \parallel P(\vec{h}|\vec{x}, \theta)\right) - H[P(\underline{\vec{h}}|\vec{x}, \theta^{(t)})] + \ln P(\vec{x}|\theta) && (21.10) \end{aligned}$$

When $\theta^{(t)} = \theta$, this becomes

$$Q(\theta|\theta) = -H[P(\underline{\vec{h}}|\vec{x}, \theta)] + \ln P(\vec{x}|\theta)\ . \tag{21.11}$$

Hence,

$$\begin{aligned} \partial_\theta Q(\theta|\theta) &= -\sum_{\vec{h}} \partial_\theta P(\underline{\vec{h}}|\vec{x}, \theta) + \partial_\theta \ln P(\vec{x}|\theta) && (21.12)\\ &= \partial_\theta \ln P(\vec{x}|\theta) && (21.13) \end{aligned}$$

So if $\theta^{(t)} \to \theta$ and $Q(\theta|\theta)$ is max at $\theta = \theta^*$, then $\ln P(\vec{x}|\theta)$ is max at $\theta = \theta^*$ too.

For a more rigorous proof that $\lim_{t\to\infty} \theta^{(t)} = \theta^*$, see Wikipedia article Ref.[81] and references therein.

Figure 21.3: Function $\mu(\theta|\theta^{(t)})$ minorizes the function $\mathcal{L}(\theta)$. Note that $\mu(\theta|\theta^{(t)})$ is always below $\mathcal{L}(\theta)$. "max" indicates $\theta^{(t+1)} = \operatorname*{argmax}_\theta \mu(\theta|\theta^{(t)})$. "kiss" indicates $\mu(\theta^{(t)}|\theta^{(t)}) = \mathcal{L}(\theta^{(t)})$.

21.2 Minorize-Maximize (MM) algorithms

A function $\mu(\theta|\theta^{(t)})$ **is said to minorize a target function** $\mathcal{L}(\theta)$ iff for all θ at fixed $\theta^{(t)}$, it satisfies the "$\mu \leq \mathcal{L}$ property"

$$\mu(\theta|\theta^{(t)}) \leq \mathcal{L}(\theta) \ , \tag{21.14}$$

and the "$\mu = \mathcal{L}$ property"

$$\mu(\theta^{(t)}|\theta^{(t)}) = \mathcal{L}(\theta^{(t)}) \ . \tag{21.15}$$

We **recursively maximize a minorizing function** $\mu(\theta|\theta^{(t)})$ if we define a sequence $(\theta^{(t)})_{t=0,1,\ldots}$ as follows:

$$\theta^{(t+1)} = \operatorname*{argmax}_\theta \mu(\theta|\theta^{(t)}) \ . \tag{21.16}$$

The sequence $(\mathcal{L}(\theta^{(t)}))_{t=0,1,2,\ldots}$ generated by recursively maximizing a minorizing function must be nondecreasing:

$$\mathcal{L}(\theta^{(t+1)}) \geq \mu(\theta^{(t+1)}|\theta^{(t)}) \geq \mu(\theta^{(t)}|\theta^{(t)}) = \mathcal{L}(\theta^{(t)}) \ . \tag{21.17}$$

A **minorize-maximize (MM) algorithm** is any algo that specifies a minorizing function $\mu(\theta|\theta^{(t)})$ for a particular target function $\mathcal{L}(\theta)$. One can also define a

majorize-minimize algo (also called MM) by inverting the inequalities throughout.

The EM algo is an MM algo. Indeed, if we define

$$\mathcal{L}(\theta) = \ln P(\vec{x}|\theta) \tag{21.18}$$

and

$$\mu(\theta|\theta^{(t)}) = Q(\theta|\theta^{(t)}) + H(P(\underline{\vec{h}}|\vec{x}, \theta^{(t)}) \ , \tag{21.19}$$

then Eq.(21.10) establishes the $\mu \leq \mathcal{L}$ and $\mu = \mathcal{L}$ properties required of a minorizing function.

How an MM algo works is portrayed in Fig.21.3.

21.3 Examples

21.3.1 Gaussian mixture

$x[\sigma] \in \mathbb{R}^d = S_{\underline{x}}$. $S_{\underline{h}}$ discrete and not too large. $n_{\underline{h}} = |S_{\underline{h}}|$ is number of Gaussians that we are going to fit the samples with.

Let

$$\theta = [w_h, \mu_h, \Sigma_h]_{h \in S_{\underline{h}}} \ , \tag{21.20}$$

where $[w_h]_{h \in S_{\underline{h}}}$ is a probability distribution of weights, and where $\mu_h \in \mathbb{R}^d$ and $\Sigma_h \in \mathbb{R}^{d \times d}$ are the mean value vector and covariance matrix of a d-dimensional Gaussian distribution.

The TPMs, printed in blue, for the nodes of Fig.21.1, for the special case of a mixture of Gaussians, are as follows:

$$P(x[\sigma] \mid h[\sigma] \mid \theta) = \mathcal{N}_d(x[\sigma]; \mu_{h[\sigma]}, \Sigma_{h[\sigma]}) \tag{21.21}$$

$$P(h[\sigma] \mid \theta) = w_{h[\sigma]} \tag{21.22}$$

Note that

$$P(x[\sigma] \mid \theta) = \sum_h P(x[\sigma] \mid h[\sigma] = h, \theta) P(h[\sigma] = h \mid \theta) \tag{21.23}$$

$$= \sum_h w_h \mathcal{N}_d(x[\sigma]; \mu_h, \Sigma_h) \tag{21.24}$$

$$P(\vec{x}, \vec{h}|\theta) = \prod_{\sigma} \left[w_{h[\sigma]} \mathcal{N}_d(x[\sigma]; \mu_{h[\sigma]}, \Sigma_{h[\sigma]}) \right] \tag{21.25}$$

$$= \prod_{\sigma} \prod_{h} \left[w_h \mathcal{N}_d(x[\sigma]; \mu_h, \Sigma_h) \right]^{\mathbb{1}(h=h[\sigma])} \tag{21.26}$$

Old Faithful: See Wikipedia Ref.[81] for an animated gif of a classic example of using EM to fit samples with a Gaussian mixture. Unfortunately, could not include it here because pdflatex does not support animated gifs. The gif shows samples in a 2 dimensional space (eruption time, delay time) from the Old Faithful geyser. In that example, $d = 2$ and $n_{\underline{h}} = 2$. Two clusters of points in a plane are fitted by a mixture of 2 Gaussians.

K-means clustering is often presented as the main competitor to EM for doing **clustering (non-supervised learning)**. In K-means clustering, the sample points are split into K mutually disjoint sets $S_0, S_1, \ldots, S_{K-1}$. The algorithm is easy to describe:

1. Initialize by choosing at random K data points $(\mu_k)_{k=0}^{K-1}$ called means or centroids and placing μ_k in S_k for all k.

2. **STEP 1:** For each data point, add it to the S_k whose centroid μ_k is closest to it.

3. **STEP 2:** Recalculate the centroids. Set μ_k equal to the mean value of set S_k.

4. Repeat steps 1 and 2 until the centroids stop changing by much.

Step 1 is analogous to the expectation step in EM, and Step 2 to the maximization step in EM (θ estimation versus μ_k estimation). We won't say anything further about K-means clustering because it isn't related to bnets in any way, and this is a book about bnets. For more info about K-means clustering, see Ref.[95].

21.3.2 Blood Genotypes and Phenotypes

Notation: $\underline{\vec{a}} = (\underline{a}[\sigma])_{\sigma=0,1,\ldots,nsam-1}$, where $nsam$ is the number of samples. Will sometimes denote $a[\sigma]$ by $a^{[\sigma]}$.

Suppose $\underline{\vec{x}} = (\underline{\vec{x}}_0)$ (i.e., just one component)
$\underline{\vec{h}} = (\underline{\vec{h}}_0)$ (i.e., just one component)
$\underline{h}[\sigma] \in S_{\underline{h}} = \{AA, AO, BB, BO, OO, AB\}$ (the 6 blood genotypes)
$\underline{x}[\sigma] \in S_{\underline{x}} = \{A, B, O, AB\}$ (the 4 blood phenotypes)

Figure 21.4: bnet for blood phenotypes $x[\sigma]$ and genotypes $h[\sigma]$.

For the bnet of Fig.21.4, the TPMs, printed in blue, are:

$$P(h^{[\sigma]}|\theta) = \begin{array}{c|c} & \\ \hline AA & p_A^2 \\ AO & 2p_Ap_O \\ BB & p_B^2 \\ BO & 2p_Bp_O \\ OO & p_O^2 \\ AB & 2p_Ap_B \end{array} , \tag{21.27}$$

where $p_A + p_B + p_O = 1$.

$$P(x^{[\sigma]} \mid h^{[\sigma]}, \theta) = \begin{array}{c|cccccc} & AA & AO & BB & BO & OO & AB \\ \hline A & 1 & 1 & 0 & 0 & 0 & 0 \\ B & 0 & 0 & 1 & 1 & 0 & 0 \\ O & 0 & 0 & 0 & 0 & 1 & 0 \\ AB & 0 & 0 & 0 & 0 & 0 & 1 \end{array} \tag{21.28}$$

$$\theta = (p_A, p_B) \tag{21.29}$$

Multiplying the TPMs in Eqs.(21.27 and (21.28), we get

$$P(x^{[\sigma]} \mid \theta) = \begin{array}{c|l} & \\ \hline A & p_A^2 + 2p_Ap_O (= \pi_A) \\ B & p_B^2 + 2p_Bp_O (= \pi_B) \\ O & p_O^2 (= \pi_O) \\ AB & 2p_Ap_B (= \pi_{AB}) \end{array} \tag{21.30}$$

Note that

$$P(\vec{x}|\theta) = \prod_\sigma P(x^{[\sigma]}|\theta) \tag{21.31}$$

$$= (\pi_A)^{N_A}(\pi_B)^{N_B}(\pi_O)^{N_O}(\pi_{AB})^{N_{AB}} , \tag{21.32}$$

where N_x for $x \in S_{\underline{x}} = \{A, B, O, AB\}$ are the counts from the data. We can get estimates for the parameters p_A and p_B right here without doing EM. Just note that

$$\hat{\pi}_x = \frac{N_x}{N_+} \tag{21.33}$$

for $x \in S_{\underline{x}}$, where $N_+ = \sum_x N_x$. Eqs.(21.33) give 4 quadratic equations that can be solved for the parameters p_A, p_B in terms of the observed counts N_x for $x \in S_{\underline{x}}$.

If, instead, you want to find the optimum parameters p_A, p_B using EM, note that

$$
\begin{aligned}
Q(\theta|\theta^{(t)}) &= \sum_{\vec{h}} P(\vec{h}|\theta^{(t)}) \ln P(\vec{x}, \vec{h}|\theta) && (21.34)\\
&= \sum_{\vec{h}} \left[\prod_{\sigma} P(h^{[\sigma]}|\theta^{(t)})\right] \ln \left[\prod_{\sigma} P(x^{[\sigma]}, h^{[\sigma]}|\theta)\right] && (21.35)\\
&= \sum_{\sigma} \sum_{h^{[\sigma]}} P(h^{[\sigma]}|\theta^{(t)}) \ln P(x^{[\sigma]}, h^{[\sigma]}|\theta) && (21.36)\\
&= \sum_{\sigma} \sum_{h^{[\sigma]}} P(h^{[\sigma]}|\theta^{(t)}) [\ln P(x^{[\sigma]}|h^{[\sigma]}, \theta) + \ln P(h^{[\sigma]}|\theta)] && (21.37)\\
&= nsam \sum_{h^{[\sigma]}} P(h^{[\sigma]}|\theta^{(t)}) \ln P(h^{[\sigma]}|\theta) \ . && (21.38)
\end{aligned}
$$

21.3.3 Missing Data/Imputation

The previous example on blood genotypes and phenotypes assumed no missing data in compiling the counts N_x. But what if there is missing data? Can one still apply the EM algo in that case? Yes! See Chapter 44.

Chapter 22

Frisch-Waugh-Lovell (FWL) theorem

The Frisch-Waugh-Lovell (FWL) theorem (see Ref.[82]) (mnemonic: FoWL Theorem) is a method used in Linear Regression (LR). It allows us to calculate a regression coefficient by doing LR with two residuals. Let's call them the **independent residual** and the **dependent or target residual**.[1] These two residuals are calculated with the help of two previously performed LR steps.

As in the section Linear regression, Ordinary Least Squares (OLS) on LR, we will consider two cases: x^σ non-random, and x^σ random i.i.d..

22.1 FWL, assuming x^σ are non-random

Suppose

$$y = X_1\beta_1 + X_2\beta_2 + \epsilon \tag{22.1}$$

where

$y, \epsilon \in \mathbb{R}^{nsam}$
$X_a \in \mathbb{R}^{nsam\times k_a}$, $\beta_a \in \mathbb{R}^{k_a}$ for $a = 1, 2$
Define the matrices U_1 and A_1 by

$$U_1 = X_1(X_1^T X_1)^{-1} X_1^T \tag{22.2}$$

and

$$A_1 = 1 - U_1 \ . \tag{22.3}$$

Note that

$$U_1 X_1 = X_1 \ , \quad A_1 X_1 = 0 \tag{22.4}$$

[1]Standard LR is performed with two features, the independent feature and the dependent or target feature.

(mnemonic: A_1 Annihilates X_1, and U_1 acts like Unity on X_1).

Applying A_1 to Eq.(22.1) gives

$$A_1 y = A_1 X_2 \beta_2 + A_1 \epsilon \tag{22.5}$$

so we can estimate β_2 by

$$\boxed{\hat{\beta}_2 = (A_1 X_2)^{-1} A_1 y} \, . \tag{22.6}$$

22.2 FWL, assuming x^σ are random

Assume for simplicity that $k_1 = k_2 = 1$ in Eq.(22.1). Let $\beta_1 = \alpha \in \mathbb{R}$, $\beta_2 = \beta \in \mathbb{R}$. When the x^σ are random and i.i.d., and X_1, X_2 are replaced by the random variables $\underline{x}, \underline{d} \in \mathbb{R}$, Eq.(22.1) is equivalent to

$$\underline{y} = \alpha \underline{x} + \beta \underline{d} + \underline{u}_{\underline{y}} \, . \tag{22.7}$$

As usual, we will denote the linear operator $\langle \underline{x}, \underline{x} \rangle^{-1} \langle \underline{x}, \cdot \rangle$ by a derivative operator:

$$\frac{\langle \underline{x}, \cdot \rangle}{\langle \underline{x}, \underline{x} \rangle} = \frac{d\cdot}{d\underline{x}} \, . \tag{22.8}$$

Define operators $U_{\underline{x}}$ and $A_{\underline{x}}$ by

$$U_{\underline{x}}(\cdot) = \underline{x} \frac{d\cdot}{d\underline{x}} \tag{22.9}$$

and

$$A_{\underline{x}} = 1 - U_{\underline{x}} \, . \tag{22.10}$$

Note that

$$U_{\underline{x}} \underline{x} = \underline{x} \, , \quad A_{\underline{x}} \underline{x} = 0 \, . \tag{22.11}$$

If we apply $A_{\underline{x}}$ to Eq.(22.7), we get

$$A_{\underline{x}} \underline{y} = \beta A_{\underline{x}} \underline{d} + A_{\underline{x}} \underline{u}_{\underline{y}} \, . \tag{22.12}$$

Hence,

$$\left\langle A_{\underline{x}} \underline{d}, A_{\underline{x}} \underline{y} \right\rangle = \beta \left\langle A_{\underline{x}} \underline{d}, A_{\underline{x}} \underline{d} \right\rangle \, . \tag{22.13}$$

Thus,

$$\boxed{\beta = \frac{d(A_{\underline{x}}\underline{y})}{d(A_{\underline{x}}\underline{d})}} \; . \tag{22.14}$$

Eqs.(22.6) and (22.14) constitute our statement of the FWL theorem. They are the main result of this chapter. But before ending this chapter, let us show how the FWL theorem for random x^σ can be interpreted graphical using LDEN diagrams (LDEN diagrams are discussed in Chapter 36).

Figure 22.1: LDEN diagrams for discussing the FWL theorem.

Fig.22.1(a) is a graphical representation of the following equation:

$$\underline{y} = \alpha\underline{x} + \beta\underline{d} + \underline{u}_{\underline{y}} \; . \tag{22.15a}$$

Likewise, 22.1(b) represents

$$\begin{cases} \underline{d} = \gamma\underline{x} + \underline{u}_{\underline{d}} \\ \underline{y} = \alpha'\underline{x} + \underline{u}'_{\underline{y}} \end{cases} \tag{22.15b}$$

and 22.1(c) represents

$$A_{\underline{x}}\underline{y} = \beta A_{\underline{x}}\underline{d} \; . \tag{22.15c}$$

Applying $d/d\underline{x}$ to Eq.(22.15a yields

$$\frac{d\underline{y}}{d\underline{x}} = \alpha + \beta\frac{d\underline{d}}{d\underline{x}} \; . \tag{22.16}$$

Applying $d/d\underline{x}$ to Eq.(22.15b yields

$$\gamma = \frac{d\underline{d}}{d\underline{x}} \; , \quad \alpha' = \frac{d\underline{y}}{d\underline{x}} \; . \tag{22.17}$$

Therefore,

$$\alpha' = \alpha + \beta\gamma \; . \tag{22.18}$$

Applying $A_{\underline{x}}$ to Eq.(22.15b) yields

$$A_{\underline{x}}\underline{y} = (1 - \underline{x}\frac{d}{d\underline{x}})\underline{y} = \underline{y} - \alpha'\underline{x} \tag{22.19}$$

and

$$A_{\underline{x}}\underline{d} = (1 - \underline{x}\frac{d}{d\underline{x}})\underline{d} = \underline{d} - \gamma\underline{x} \ . \tag{22.20}$$

Therefore, Eq.(22.15c) is equivalent to

$$\underline{y} - \alpha'\underline{x} = \beta(\underline{d} - \gamma\underline{x}) \ . \tag{22.21}$$

Chapter 23

Frontdoor Adjustment Formula

The frontdoor (FD) adjustment formula is proven in Chapter 15 from the rules of Do Calculus. The goal of this chapter is to give examples of the use of that theorem. We will restate the theorem in this chapter, sans proof. There is no need to understand the theorem's proof in order to use it. However, you will need to skim Chapter 15 in order to familiarize yourself with the notation used to state the theorem. This chapter also assumes that you are comfortable with the rules for checking for d-separation. Those rules are covered in Chapter 17.

Suppose that we have access to data that allows us to estimate a probability distribution $P(x., m., y.)$. Hence, the variables $\underline{x}., \underline{m}., \underline{y}.$ are ALL observed (i.e, not hidden). Then we say that the frontdoor $\underline{m}.$ satisfies the **frontdoor adjustment criterion** relative to $(\underline{x}., \underline{y}.)$ if

1. All directed paths from $\underline{x}.$ to $\underline{y}.$ are intercepted by (i.e., have a node in) $\underline{m}.$.
2. All backdoor paths from $\underline{x}.$ to $\underline{m}.$ are blocked.
3. All backdoor paths from on $\underline{m}.$ to $\underline{y}.$ are blocked by conditioning on $\underline{x}.$.

Claim 36 *Frontdoor Adjustment Formula*

If $\underline{m}.$ satisfies the frontdoor criterion relative to $(\underline{x}., \underline{y}.)$, and $P(x., m.) > 0$, then

$$P(y.|\mathcal{D}\underline{x}. = x.) = \sum_{m.} \underbrace{\left[\sum_{x'.} P(y.|x'., m.)P(x'.)\right]}_{P(y.|\mathcal{D}\underline{m}.=m.)} \underbrace{P(m.|x.)}_{P(m.|\mathcal{D}\underline{x}.=x.)} \tag{23.1}$$

$$= \tag{23.2}$$

$$\begin{tikzcd} & & \sum x'. \arrow[dr] & \\ x. \arrow[r] & \sum m. \arrow[r] & & y. \end{tikzcd}$$

where $\sum x'.$ and $\sum m.$ means nodes $\underline{x}'.$ and $\underline{m}.$ are summed over.

proof: See Chapter 15.
QED

23.1 Examples

1.

$$\begin{tikzcd} & \underline{c} \arrow[dl] \arrow[dr] & \\ \underline{x} \arrow[r] & \underline{m} \arrow[r] & \underline{y} \end{tikzcd} \tag{23.3}$$

If $\underline{x}. = \underline{x}, \underline{m}. = \underline{m}$ and $\underline{y}. = \underline{y}$, then the FD criterion is satisfied. Can't satisfy backdoor criterion because $\underline{c}$ unobserved so can't condition on it to block backdoor path $\underline{x} - \underline{c} - \underline{y}$.

2.

$$\begin{tikzcd} \underline{z}_1 \arrow[d] \arrow[dr] & & \underline{z}_2 \arrow[dl] \arrow[d] \\ \underline{w}_1 \arrow[d] & \underline{c} \arrow[dl] \arrow[dr] & \underline{w}_2 \arrow[d] \\ \underline{x} \arrow[r] & \underline{m} \arrow[r] & \underline{y} \end{tikzcd} \tag{23.4}$$

If $\underline{x}. = \underline{x}, \underline{m}. = \underline{m}$ and $\underline{y}. = \underline{y}$, then the FD criterion is satisfied. Can't satisfy backdoor criterion because to block backdoor path $\underline{x} - \underline{c} - \underline{y}$, need to condition on $\underline{c}$ but if this is true, then long path $\underline{x} - \underline{w}_1 - \underline{z}_1 - \underline{c} - \underline{z}_2 - \underline{w}_2 - \underline{y}$ becomes unblocked.

Chapter 24

Gaussian Nodes with Linear Dependence on Parents

Bnet nodes that have a Gaussian TPM with a linear dependence on their parent nodes (GLP) are a very popular way of modeling continuous nodes of bnets. A convenient aspect of them is that their parents can be discrete or continuous nodes, and their children can be discrete or continuous nodes too. Also, they can be learned easily from the data because their parameters can be expressed in terms of two node covariances. For these reasons, they are commonly used when doing structure learning of bnets with continuous nodes (see Chapter 66).

Figure 24.1: GLP node $\underline{y}$ with 3 parent nodes $\underline{x}^3 = (\underline{x}_1, \underline{x}_2, \underline{x}_3)$.

Recall our notation for a Gaussian distribution:

$$\mathcal{N}(x; \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}} \ , \tag{24.1}$$

where $x, \mu \in \mathbb{R}$ and $\sigma > 0$.

A GLP node $\underline{y}$ with n parents $\underline{x}^n = (\underline{x}_1, \underline{x}_2, \ldots, \underline{x}_n)$ has the following TPM:

$$P(y|x^n) = \mathcal{N}(y; \beta_0 + \beta^{nT} x^n, \sigma^2) \tag{24.2}$$

where $\underline{y}, \beta_0, \in \mathbb{R}$ and $\sigma^2 > 0$, and where $\underline{x}^n, \beta^n \in \mathbb{R}^n$ are **column vectors**. The T in β^{nT} stands for transpose. Any $\underline{x}_i$ can have a discrete set of states as long as

they are real valued and ordinal (ordered by size). Fig.24.1 shows a diagrammatic representation of a GLP node with 3 parents.

Note that as $\sigma \to 0$, a GLP node becomes deterministic. In fact, it becomes a neural net node with a linear activation function.

An equivalent way of defining a GLP node $\underline{y}$ is in terms of a random variable equation expressing $\underline{y}$ as a hyperplane function of the parents $\underline{x}^n$ plus a Gaussian noise variable. Define an estimator $\hat{\underline{y}}$ of a "true value" $\underline{y}$ by

$$\hat{\underline{y}} = \beta_0 + \beta^{nT} \underline{x}^n \tag{24.3a}$$

and

$$\underline{y} = \hat{\underline{y}} + \underline{\epsilon} \tag{24.3b}$$

where the residual $\underline{\epsilon}$ satisfies

$$P(\epsilon) = \mathcal{N}(\epsilon; 0, \sigma^2) \tag{24.3c}$$

and

$$\langle \underline{x}^n, \underline{\epsilon} \rangle = 0 \ . \tag{24.3d}$$

The notation $\langle \underline{x}, \underline{y} \rangle$ for the covariance of random variables $\underline{x}$ and $\underline{y}$ is explained in Chapter Notational Conventions and Preliminaries.

Claim 37 *The parameters of a GLP node can be expressed in terms of 2-node covariances. Specifically,*

$$\beta^n = \left\langle \underline{x}^n, \underline{x}^{nT} \right\rangle^{-1} \left\langle \underline{y}, \underline{x}^n \right\rangle \tag{24.4}$$

$$\beta_0 = \left\langle \underline{y} \right\rangle - \beta^{nT} \left\langle \underline{x}^n \right\rangle \tag{24.5}$$

$$\sigma^2 = \left\langle \underline{y}, \underline{y} \right\rangle - \beta^{nT} \left\langle \underline{x}^n, \underline{y} \right\rangle \tag{24.6}$$

proof:

Note that $\left\langle \underline{x}^n, \underline{x}^{nT} \right\rangle^T = \left\langle \underline{x}^n, \underline{x}^{nT} \right\rangle$ and $\left\langle \underline{y}, \underline{x}^{nT} \right\rangle^T = \left\langle \underline{y}, \underline{x}^n \right\rangle$.

$$\left\langle \underline{y}, \underline{x}^{nT} \right\rangle = \beta^{nT} \left\langle \underline{x}^n, \underline{x}^{nT} \right\rangle \tag{24.7}$$

$$\left\langle \underline{y}, \underline{x}^n \right\rangle = \left\langle \underline{x}^n, \underline{x}^{nT} \right\rangle \beta^n \tag{24.8}$$

$$\beta^n = \left\langle \underline{x}^n, \underline{x}^{nT} \right\rangle^{-1} \left\langle \underline{y}, \underline{x}^n \right\rangle \tag{24.9}$$

$$\langle \underline{y} \rangle = \beta_0 + \beta^{nT} \langle \underline{x}^n \rangle \tag{24.10}$$

$$\begin{aligned} \langle \underline{y}, \underline{y} \rangle &= \langle \beta_0 + \beta^{nT} \underline{x}^n + \underline{\epsilon}, \underline{y} \rangle & (24.11) \\ &= \beta^{nT} \langle \underline{x}^n, \underline{y} \rangle + \sigma^2 & (24.12) \end{aligned}$$

QED

Let D=Discrete, GLP=Gaussian with Linear dependence in Parents
The following arrows are possible in a bnet.

- $GLP \leftarrow GLP$
- $GLP \leftarrow D$

  Pass to GLP a separate set of regression coefficients β_0, β^n and variance σ^2 for each state of D. If D is called $\underline{d}$, let

  $$P(y|(x^n)_d, d) = \mathcal{N}(y; (\beta_0)_d + (\beta^{nT})_d (x^n)_d, \sigma_d^2) \tag{24.13}$$

  for each $d \in S_{\underline{d}}$.

- $D \leftarrow GLP$

  If D expects a continuous parent, no need to preprocess GLP output. If D expects a discrete parent, break the interval $[a, b]$ that contains most of the range of the GLP node into sub-intervals and assign a discrete label to each subinterval.

- $D \leftarrow D$

Chapter 25

Generative Adversarial Networks (GANs)

Figure 25.1: Generative Adversarial Network (GAN)

Original GAN, Ref.[12](2014).

Generator G (counterfeiter) generates samples $\vec{f}$ of fake money and submits them to Discriminator D (Treasury agent). D also gets samples $\vec{r}$ of real money. D submits verdict $V \in [0, 1]$. G depends on parameter θ_G and D on parameter θ_D. Verdict V and initial θ_G, θ_D are used to get new parameters θ'_G, θ'_D.Process is repeated

Figure 25.2: Discriminator node $\underline{V}$ in Fig.25.1 can be split into 3 nodes $\vec{\underline{c}}$, $\vec{\underline{d}}$ and $\underline{V}$.

(Dynamical Bayesian Network) until saddle point in $V(\theta_G, \theta_D)$ is reached. D makes G better and vice versa. Zero-sum game between D and G.

Let $\mathcal{D}$ be the domain of $D(\cdot, \theta_D)$. Assume that for any $x \in \mathcal{D}$,

$$0 \leq D(x, \theta_D) \leq 1 \ . \tag{25.1}$$

For any $S \subset \mathcal{D}$, define

$$\sum_{x \in S} D(x, \theta_D) = \lambda(S, \theta_D) \ . \tag{25.2}$$

In general, $G(\cdot, \theta_G)$ need not be real valued.

Assume that for every $u \in S_{\underline{u}}$, $G(u, \theta_G) = f \in S_{\underline{f}} \subset \mathcal{D}$. Define

$$\overline{D}(f, \theta_D) = 1 - D(f, \theta_D) \ . \tag{25.3}$$

Note that

$$0 \leq \overline{D}(f, \theta_D) \leq 1 \ . \tag{25.4}$$

Define:

$$V(\theta_G, \theta_D) = \sum_r P(r) \ln D(r, \theta_D) + \sum_u P(u) \ln \overline{D}(G(u, \theta_G), \theta_D) \ . \tag{25.5}$$

We want the first variation of $V(\theta_G, \theta_D)$ to vanish.

$$\delta V(\theta_G, \theta_D) = 0 \ . \tag{25.6}$$

This implies

$$\partial_{\theta_G} V(\theta_G, \theta_D) = \partial_{\theta_D} V(\theta_G, \theta_D) = 0 \tag{25.7}$$

and

$$V_{opt} = \min_{\theta_G} \max_{\theta_D} V(\theta_G, \theta_D) \ . \tag{25.8}$$

Node TPMs for Figs.25.1 and 25.2 are given next in blue:

$$P(\theta_G) = \text{ given} \tag{25.9}$$

$$P(\theta_D) = \text{ given} \tag{25.10}$$

$$P(\vec{u}) = \prod_i P(u[i]) \ \ \text{(usually uniform distribution)} \tag{25.11}$$

$$P(\vec{r}) = \prod_i P(r[i]) \tag{25.12}$$

$$P(f[i] \mid \vec{u}, \theta_G) = \delta[f[i], G(u[i], \theta_G)] \tag{25.13}$$

$$P(c[i] \mid \vec{f}, \theta_D) = \delta(c[i], \overline{D}(f[i], \theta_D)) \tag{25.14}$$

$$P(d[j] \mid \vec{r}, \theta_D) = \delta(d[j], D(r[j], \theta_D)) \tag{25.15}$$

$$P(V|\vec{d}, \vec{c}) = \delta(V, \frac{1}{N} \ln \prod_{i,j} (c[i]d[j])) \tag{25.16}$$

where $N = nsam(\vec{r})nsam(\vec{u})$.

Let $\eta_G, \eta_D > 0$. Maximize V wrt θ_D, and minimize it wrt θ_G.

$$P(\theta'_G|V, \theta_G) = \delta(\theta'_G, \theta_G - \eta_G \partial_{\theta_G} V) \tag{25.17}$$

$$P(\theta'_D|V, \theta_D) = \delta(\theta'_D, \theta_D + \eta_D \partial_{\theta_D} V) \tag{25.18}$$

Figure 25.3: GAN, Constraining Bayesian Network

Constraining Bnet given in Fig.25.3. It adds 2 new nodes, namely $\underline{\vec{U}}$ and $\underline{\vec{R}}$, to the bnet of Fig.25.1. The purpose of these 2 barren (childrenless) nodes is to constrain certain functions to be probability distributions.

Node TPMs for the 2 new nodes given next in blue.

$$P(U[i] \mid \theta_G) = \frac{\overline{D}(G(U[i], \theta_G), \theta_D))}{\overline{\lambda}(\theta_G, \theta_D)} \tag{25.19}$$

where $S_{\underline{U[i]}} = S_{\underline{u}}$ and $\overline{\lambda}(\theta_G, \theta_D) = \sum_u \overline{D}(G(u, \theta_G), \theta_D))$.

$$P(R[i] \mid \theta_G, \theta_D) = \frac{D(R[i], \theta_D)}{\lambda(\theta_D)} \tag{25.20}$$

where $S_{\underline{R[i]}} = S_{\underline{r}}$ and $\lambda(\theta_D) = \sum_r D(r, \theta_D)$.

$$P(V|\vec{u}, \vec{r}) = \delta(V, \frac{1}{N} \ln \prod_{i,j} (P(\underline{R[i]} = r[i] \mid \theta_G, \theta_D) P(\underline{U[i]} = u[j] \mid \theta_G))) \tag{25.21}$$

where $N = nsam(\vec{r}) nsam(\vec{u})$.

$\mathcal{L}$ = likelihood

$$\begin{aligned} \mathcal{L} &= P(\vec{r}, \vec{u}|\theta_G, \theta_D) && (25.22) \\ &= \prod_{i,j} \left[\frac{D(r[i], \theta_D)}{\lambda(\theta_D)} \frac{\overline{D}(G(u[j], \theta_G), \theta_D))}{\overline{\lambda}(\theta_G, \theta_D)} \right] && (25.23) \end{aligned}$$

$$\ln \mathcal{L} = N[V(\theta_G, \theta_D) - \ln \lambda(\theta_D) - \ln \overline{\lambda}(\theta_G, \theta_D)] \tag{25.24}$$

Chapter 26

Goodness of Causal Fit

This chapter is an almost verbatim copy of a paper of mine, Ref.[55].

26.1 Abstract

We propose a Goodness of Causal Fit (GCF) measure which depends on Pearl "do" interventions. This is different from a measure of Goodness of Fit (GF), which does not use interventions. Given a DAG set $\mathcal{G}$, to find a good $G \in \mathcal{G}$, we propose plotting $GCF(G)$ versus $GF(G)$ for all $G \in \mathcal{G}$, and finding a graph $G \in \mathcal{G}$ with a large amount of both types of goodness.

26.2 Introduction

Frequently, when students first encounter Bayesian Networks (bnets) and Causal Inference (CI), they experience serious doubts about the usefulness of this theory, because they believe finding the underlying model (i.e., DAG) for most realistic physical situations is too difficult or impossible. I think that part of the problem is that these students are assuming, perhaps unconsciously, that there exists a unique DAG that fits Nature perfectly, and a mind-boggling number of possibilities to sift through to find that DAG. Rather than looking for a unique DAG, I think a better strategy is to write down a set $\mathcal{G}$ of likely DAGs, and to calculate for each DAG in $\mathcal{G}$, a measure called Goodness of Causal Fit (GCF). Then use the DAGs with the highest GCF scores.

The goal of this paper is to propose a GCF measure. Such a measure is of course not unique, and someone may propose in the future a measure that is better than ours.

When designing a GCF measure, it is important to keep in mind the First Dictum[1] of CI: The data is model-less. In the First Dictum, when we say "data", we are referring to what is commonly called a dataset. A dataset is a table of data,

[1] This is just my whimsical name for it.

where all the entries of each column have the same units, and measure a single feature, and each row refers to one particular sample or individual. Datasets are particularly useful for estimating probability distributions and for training neural nets. In the First Dictum, when we say "model", we are referring to a DAG (directed acyclic graph) or a bnet (Bayesian Network= DAG + probability table for each node of DAG).

You can try to derive a model from a dataset, but you'll soon find out that you can only go so far. The process of finding a *partial* model from a dataset is called structure learning (SL). SL can be done quite nicely with Marco Scutari's open source program `bnlearn` (Ref[46]). The problem is that SL often cannot narrow down the model to a single one. It finds an undirected graph (UG), and it can determine the direction of some of the arrows in the UG, but it is often incapable, for well understood fundamental —not just technical— reasons, of finding the direction of *all* the arrows of the UG. So it often fails to fully specify a DAG model.

Let's call the ordered pair (dataset, model) a **data SetMo**. Then what the First Dictum is saying is that a dataset is model-free or model-less (although sometimes one can find a partial model hidden in there). A dataset is not a data SetMo.

Graphs which contain both directed and undirected edges are called **partially directed (PD) graphs**. `bnlearn` takes a dataset as input and returns a PD graph G_{pd}. Given a PD graph G_{pd}, let $\mathcal{G}(G_{pd})$ be the DAG set $\mathcal{G}$ which is generated by giving directions to all undirected edges of G_{pd} in all possible ways. We will refer to $\mathcal{G}(G_{pd})$ as the **DAG set generated by** G_{pd}. and to any $\mathcal{G}' \subset \mathcal{G}(G_{pd})$ as a **DAG set partially generated by** G_{pd}**.** Once we define below our GCF measure, we will apply it to sets of the type $\mathcal{G}(G_{pd})$ as an example.

It's clear that any measure of GCF will have to involve interventions such as the "do" intervention invented by Pearl et al for CI. Without interventions like "do", it is impossible to distinguish causally the DAGs in a set $\mathcal{G}(G_{pd})$.

Henceforth, random variables will be indicated by underlining.

26.3 Goodness of Fit

Before trying to define a GCF measure, it is instructive to review the closely related, well established, measures of Goodness of Fit (GF).

Consider two probability distributions $PO(x)$ and $PE(x)$, where $x \in S_{\underline{x}}$. By a GF measure, we mean a measure of the difference between PO and PE. Usually PO is the observed probability distribution and PE is the expected, theoretical one.

Three popular measures of the difference between PO and PE are:

1. The **Kullback-Liebler divergence**:

$$D_{KL}(PO \parallel PE) = \sum_{x \in S_{\underline{x}}} PO(x) \ln \frac{PO(x)}{PE(x)} \ . \tag{26.1}$$

2. The **Pearson divergence** (aka **Pearson Chi-squared test statistic**):

$$D_{\chi^2}(PO \parallel PE) = \sum_{x \in S_{\underline{x}}} \frac{[PO(x) - PE(x)]^2}{PE(x)} = \sum_{x \in S_{\underline{x}}} \frac{PO^2(x)}{PE(x)} - 1 \ . \tag{26.2}$$

It's easy to show using $\ln(1+\delta) = \delta + \mathcal{O}(\delta^2)$ that if $\left|\frac{PO(x)}{PE(x)} - 1\right| << 1$ for all $x \in S_{\underline{x}}$, then

$$D_{KL}(PO \parallel PE) \approx D_{\chi^2}(PO \parallel PE) \tag{26.3}$$

3. The **Euclidean distance squared**:

$$D_E(PO, PE) = \sum_{x \in S_{\underline{x}}} [PO(x) - PE(x)]^2 \tag{26.4}$$

Note that of these 3 measures, only $D_E(PO, PE)$ is symmetric in PO and PE.

Given any bnet G with full probability distribution [2] $P_G(x.)$ and a probability distribution[3] $\tilde{P}(x.)$ derived empirically from a dataset, let

$$\begin{aligned} D(G) &= \sum_{x.} \tilde{P}(x.) \ln \frac{\tilde{P}(x.)}{P_G(x.)} && (26.5) \\ &= D_{KL}(\tilde{P}(\underline{x}.) \parallel P_G(\underline{x}.)) && (26.6) \end{aligned}$$

We define **Goodness of Fit (GF)** of the bnet G by

$$GF(G) = \ln \frac{1}{D(G)} \tag{26.7}$$

26.4 GCF example 1

For the first example of our measure of GCF, we consider $\mathcal{G}(G_{pd}) = \{G_1, G_2\}$ given by Fig.26.1. We will assume the following:

- First, we assume that we have collected a dataset from which we have extracted a full empirical distribution $\tilde{P}(z, a, b)$. From $\tilde{P}(z, a, b)$, we assume that the following have been calculated. $\tilde{P}(z, b|a)$ $\tilde{P}(z, a|b)$ $\tilde{P}(a)$, $\tilde{P}(b)$.

- Second, we assume that a dataset has been collected for which $\underline{a}$ was held fixed to each of the possible values $a \in S_{\underline{a}}$ of $\underline{a}$. Furthermore, we assume that the distribution $\tilde{P}(z, b|do(\underline{a}) = a)$ has been extracted from that dataset.

[2] We define $x.$ to be a vector with components x_i

[3] Empirical distributions will be denoted by P with a tilde over it.

Figure 26.1: $\mathcal{G}(G_{pd}) = \{G_1, G_2\}$. The partially directed graph G_{pd} generates the DAGs G_1 and G_2 by giving directions to all undirected edges of G_{pd} in all possible ways. (In this case, there is only one undirected edge in G_{pd}.)

- Third, we assume that a dataset has been collected for which $\underline{b}$ was held fixed to each of the possible values $b \in S_{\underline{b}}$ of $\underline{b}$. Furthermore, we assume that the distribution $\tilde{P}(z, a|do(\underline{b}) = b)$ has been extracted from that dataset.

We will refer to $\tilde{P}(z, b|do(\underline{a}) = a)$ and $\tilde{P}(z, a|do(\underline{b}) = b)$ as **empirical do-probability distributions**.

Now define

$$D_a = \sum_{z,b} \tilde{P}(z, b|a) \ln \frac{\tilde{P}(z, b|a)}{\tilde{P}(z, b|do(\underline{a}) = a)} \tag{26.8}$$

$$= D_{KL}(\tilde{P}(\underline{z}, \underline{b}|a) \parallel \tilde{P}(\underline{z}, \underline{b}|do(\underline{a}) = a)) \tag{26.9}$$

$$D_{\underline{a}} = \sum_a \tilde{P}(a) D_a = E_a[D_a] \tag{26.10}$$

and

$$D_b = D_{KL}(\tilde{P}(\underline{z}, \underline{a}|b) \parallel \tilde{P}(\underline{z}, \underline{a}|do(\underline{b}) = b)) \tag{26.11}$$

$$D_{\underline{b}} = \sum_a \tilde{P}(b) D_b = E_b[D_b] \ . \tag{26.12}$$

We will refer to $D_{\underline{a}}$ and $D_{\underline{b}}$ as **do-divergences**.

Note that

$$D_a(G_2) = 0 \text{ for all } a \text{ so } \underbrace{D_{\underline{a}}(G_2)}_{0} \leq D_{\underline{b}}(G_2) \tag{26.13}$$

and

$$D_b(G_1) = 0 \text{ for all } b \text{ so } D_{\underline{a}}(G_1) \geq \underbrace{D_{\underline{b}}(G_1)}_{0} \ . \tag{26.14}$$

If $D_{\underline{a}} \leq D_{\underline{b}}$, then $\underline{a} \to \underline{b}$, and if $D_{\underline{a}} \geq D_{\underline{b}}$ then $\underline{a} \leftarrow \underline{b}$. Thus, the arrow and the inequality sign point in opposite directions. Alternatively, just remember that the arrow points to the larger of the two D's.

If $D_{\underline{a}} \leq D_{\underline{b}}$, then define $GCF(G_1) = -1$ and $GCF(G_2) = 1$.

If $D_{\underline{b}} \leq D_{\underline{a}}$, then define $GCF(G_1) = 1$, $GCF(G_2) = -1$.

26.5 GCF example 2

Figure 26.2: $\mathcal{G} = \{G_1, G_2, G_3\}$. $\mathcal{G}$ is a set of observationally equivalent (OE) graphs. These are graphs that have the same value for GF, and are therefore indistinguishable from GF alone. For more info about OE graphs, see Chapter 51. Note that $\mathcal{G}(G_{pd})$ includes one more DAG, the one in which node $\underline{x}_1$ is a collider.

For the second example of our measure of GCF, consider $\mathcal{G} = \{G_1, G_2, G_3\}$ given by Fig.26.2.

Which of the do-divergences $D_{\underline{x}_2}$, $D_{\underline{x}_1}$ and $D_{\underline{x}_3}$, is smallest, middle and highest, depends on the empirical do-probability distributions. For definiteness, suppose the sizes of these do-divergences are related as follows:

$$D_{\underline{x}_2} \leq D_{\underline{x}_1} \leq D_{\underline{x}_3} \ . \tag{26.15}$$

For any two do-divergences $D_{\underline{a}}$ and $D_{\underline{b}}$, define the distance

$$d_{\underline{b},\underline{a}} = |D_{\underline{b}} - D_{\underline{a}}| \tag{26.16}$$

If we abbreviate $D_{\underline{x}_j}$ by D_j, we can define the GCF for each of the graphs in $\mathcal{G}$ by:

$$GCF(G_1) = \frac{-d_{2,1} + d_{1,3}}{d_{2,1} + d_{1,3}} \tag{26.17a}$$

$$GCF(G_2) = \frac{d_{2,1} + d_{1,3}}{d_{2,1} + d_{1,3}} = 1 \tag{26.17b}$$

$$GCF(G_3) = \frac{-d_{2,1} - d_{1,3}}{d_{2,1} + d_{1,3}} = -1 \tag{26.17c}$$

26.6 GCF in general

Eqs.(26.17) are a special case of the following formulae.

For any two do-divergences $D_{\underline{a}}$ and $D_{\underline{b}}$, define the **do-divergence distance** by

$$d_{\underline{b},\underline{a}} = |D_{\underline{b}} - D_{\underline{a}}| \ . \tag{26.18}$$

For any $G_i \in \mathcal{G}$, define the **edge-sign function** by

$$\sigma_{G_i}(\underline{a}—\underline{b}) = \begin{cases} +1 & \text{if edge } \underline{a}—\underline{b} \text{ in } G_i \text{ points towards larger of } D_{\underline{a}} \text{ and } D_{\underline{b}}. \\ -1 & \text{otherwise} \end{cases} \tag{26.19}$$

Finally, suppose that $\mathcal{G}$ is either partially or fully generated by a PD graph G_{pd} with undirected edges $\{\underline{a}_k—\underline{b}_k\}_{k=0,1,\ldots,nk-1}$. Then define the GCF of graph $G_i \in \mathcal{G}$ by

$$GCF(G_i) = \frac{\sum_{k=0}^{nk-1} \sigma_{G_i}(\underline{a}_k—\underline{b}_k) d_{\underline{a}_k,\underline{b}_k}}{\sum_{k=0}^{nk-1} d_{\underline{a}_k,\underline{b}_k}} \ . \tag{26.20}$$

Note that $-1 \leq GCF(G_i) \leq 1$.

If the DAG set $\mathcal{G}$ contains only one DAG G, define $GCF(G) = 1$, because all arrows in G are in the correct direction, as far as we know.

Call the **skeleton** of a DAG, the undirected graph that one obtains by turning all its edges from directed to undirected ones.

So far, we have applied our measure of GCF to a DAG set $\mathcal{G}$ which is either fully or partially generated by a PD graph G_{pd}, or is a singleton set. But what if we want a GCF that can score every DAG in a DAG set $\mathcal{G}$ that contains DAGs with different skeletons? In that case, let $\{\underline{a}_k—\underline{b}_k\}_{k=0,1,\ldots,nk-1}$ be *all* the edges of graph $G_i \in \mathcal{G}$, and use the relative GCF given by Eq.(26.20), or use an absolute GCF defined by

$$GCF_a(G_i) = \sum_{k=0}^{nk-1} \sigma_{G_i}(\underline{a}_k—\underline{b}_k) d_{\underline{a}_k,\underline{b}_k} \ . \tag{26.21}$$

So let $\mathcal{G}$ be an arbitrary DAG set. Our GCF (or GCF_a) measure is not enough to decide the best possible G in $\mathcal{G}$, because there might be several graphs with $GCF \approx 1$. For this reason, we recommend plotting $GCF(G)$ (or $GCF_a(G)$) versus $GF(G)$ for all $G \in \mathcal{G}$. Then choose a G with a large amount of both types of goodness. It might even be advantageous to average over a small subset of DAGs in $\mathcal{G}$ that have large amounts of both types of goodness. This would be similar to averaging over an ensemble of decision trees to get a random forest.

A plot of $GCF(G)$ versus $GF(G)$ agrees with the spirit of the First Dictum and data setmos, because also in those, we acknowledge a separation between the dataset (GF) and model (GCF) degrees of freedom.

Chapter 27

Granger Causality

This chapter is based on the Wikipedia article Ref.[87] and Scholarpedia article Ref.[45] on Granger causality and on the book Ref.[14] on time series analysis by Hamilton.

This chapter assumes the reader has read Chapter 69 on the stationary time series $ARMA(p, q)$ and $VAR(p)$.

Figure 27.1: When t-series $\mathcal{X}$ Granger-causes t-series $\mathcal{Y}$, the patterns in $\mathcal{X}$ are approximately repeated in $\mathcal{Y}$ after some time lag (two examples are indicated with arrows). Thus, past values of $\mathcal{X}$ can be used for the prediction of future values of $\mathcal{Y}$. (image and caption from Ref.[87])

Let $\underline{\vec{x}}_t = [\underline{x}_t^A]_{\forall A} \in \mathbb{R}^{nr_1 \times 1}$ and $\underline{\vec{y}}_t = [\underline{y}_t^B]_{\forall B} \in \mathbb{R}^{nr_2 \times 1}$. Thus, $\underline{\vec{x}}_t$ for each t is a column vector with nr_1 rows, and $\underline{\vec{y}}_t$ for each t is a column vector with nr_2 rows. Let $nr_1 + nr_2 = nr$, the total number of rows. Consider a vector t-series $\{\underline{\vec{x}}_t, \underline{\vec{y}}_t\}_{\forall t}$ of type $VAR(p)$, as defined by Eq.(69.183). To simplify the notation of Eq.(69.183), we are

replacing x^1 by x, x^2 by y, n^1 by n, and n^2 by w.

$$\begin{bmatrix} \underline{x}_t^A \\ \underline{y}_t^B \end{bmatrix} = \sum_{j=1}^{p} \begin{bmatrix} \alpha_j^{\underline{x}|\underline{x};A,A'} & \alpha_j^{\underline{x}|\underline{y};A,B'} \\ \alpha_j^{\underline{y}|\underline{x};B,A'} & \alpha_j^{\underline{y}|\underline{y};B,B'} \end{bmatrix} \begin{bmatrix} \underline{x}_{t-j}^{A'} \\ \underline{y}_{t-j}^{B'} \end{bmatrix} + \begin{bmatrix} \underline{n}_t^A \\ \underline{w}_t^B \end{bmatrix} \tag{27.1}$$

Hence,

$$E_{|\vec{x}_{<t},\vec{y}_{<t}}\left[\underline{y}_t^B\right] = \sum_{j=1}^{p} \alpha_j^{\underline{y}|\underline{x};B,A'} \underline{x}_{t-j}^{A'} + \sum_{j=1}^{p} \alpha_j^{\underline{y}|\underline{y};B,B'} \underline{y}_{t-j}^{B'} \tag{27.2}$$

Let $\mathcal{X} = \{\underline{\vec{x}}_t\}_{\forall t}$ and $\mathcal{Y} = \{\underline{\vec{y}}_t\}_{\forall t}$.

We say $\mathcal{X}$ **does not G-cause (or does not G-forecast)** $\mathcal{Y}$, and symbolize this by $\mathcal{X} \underset{G}{\not\longrightarrow} \mathcal{Y}$, if

$$\alpha_j^{\underline{y}|\underline{x};B,A'} = 0 \;\; \forall (j, B, A') \tag{27.3}$$

or, equivalently,

$$E_{|\vec{x}_{<t},\vec{y}_{<t}}\left[\underline{y}_t^B\right] = E_{|\vec{y}_{<t}}\left[\underline{y}_t^B\right] \;\; \forall (B, t) \tag{27.4}$$

We say $\mathcal{X}$ **G-causes (or G-forecasts)** $\mathcal{Y}$ and symbolize this by $\mathcal{X} \underset{G}{\longrightarrow} \mathcal{Y}$, if $\mathcal{X} \underset{G}{\not\longrightarrow} \mathcal{Y}$ is false.

Figure 27.2: Bnet for $VAR(2)$. For clarity, we only show the arrows entering nodes $\underline{\vec{x}}_t$ and $\underline{\vec{y}}_t$. Remove red arrows if $\mathcal{X}$ does not G-cause $\mathcal{Y}$, and keep them if it does.

Eq.(27) describing a bipartite $VAR(p)$ t-series can be represented by the bnet

Fig.27.2. The TPMs, printed in blue, for the two nodes $\vec{x}_t$ and $\vec{y}_t$, are as follows:

$$P(\underline{\vec{x}}_t|\underline{\vec{x}}_{[t-p,t-1]},\underline{\vec{y}}_{[t-p,t-1]},\vec{n}_t) = \prod_A \mathbb{1}\left(\underline{x}_t^A = \sum_{j=1}^{p} \alpha_j^{\underline{x}|\underline{x};A,A'} \underline{x}_{t-j}^{A'} + \sum_{j=1}^{p} \alpha_j^{\underline{x}|\underline{y};A,B'} \underline{y}_{t-j}^{B'} + n_t^A\right) \tag{27.5}$$

$$P(\underline{\vec{y}}_t|\underline{\vec{x}}_{[t-p,t-1]},\underline{\vec{y}}_{[t-p,t-1]},\vec{w}_t) = \prod_B \mathbb{1}\left(\underline{y}_t^B = \sum_{j=1}^{p} \underbrace{\alpha_j^{\underline{y}|\underline{x};B,A'}}_{0?} \underline{x}_{t-j}^{A'} + \sum_{j=1}^{p} \alpha_j^{\underline{y}|\underline{y};B,B'} \underline{y}_{t-j}^{B'} + w_t^B\right) \tag{27.6}$$

Testing for GC

- Consider the datasets

$$\mathcal{D}_{\underline{x}} = \{(t, \vec{x}_{[t-p,t-1]}, \vec{y}_{[t-p,t-1]}, \boxed{\vec{x}_t}) : t\} \tag{27.7}$$

$$\mathcal{D}_{\underline{y}} = \{(t, \vec{x}_{[t-p,t-1]}, \vec{y}_{[t-p,t-1]}, \boxed{\vec{y}_t}) : t\} \tag{27.8}$$

  One can do Linear Regression on $\mathcal{D}_{\underline{x}}$ with x-variables $(\vec{x}_{[t-p,t-1]}, \vec{y}_{[t-p,t-1]})$, y-variable $\vec{x}_t$, and regression coefficients $\alpha_{[1,p]}^{\underline{x}|\underline{x}}$, $\alpha_{[1,p]}^{\underline{x}|\underline{y}}$. One can also do Linear Regression on $\mathcal{D}_{\underline{y}}$ with x-variables $(\vec{x}_{[t-p,t-1]}, \vec{y}_{[t-p,t-1]})$, y-variable $\vec{y}_t$, and regression coefficients $\alpha_{[1,p]}^{\underline{y}|\underline{x}}$, $\alpha_{[1,p]}^{\underline{y}|\underline{y}}$. The LR software yields confidence intervals for the regression coefficients.

- One can do hypothesis testing using the Likelihood Ratio Test[1]
  Null hypothesis $H_0 : \mathcal{X} \xrightarrow[G]{\|} \mathcal{Y}$, Alternative hypothesis $H_1 : \mathcal{X} \xrightarrow[G]{} \mathcal{Y}$

- Test for both $\mathcal{X} \xrightarrow[G]{} \mathcal{Y}$ and $\mathcal{Y} \xrightarrow[G]{} \mathcal{X}$.

Limitations

It has been remarked that G-causality is not true causality because, even though an event A must precede an event B in order to cause it, that does not necessarily mean that A causes B. I think the problem with G-causality is that it uses a bnet that is a good fit for the dataset, but not necessarily also a good *causal* fit for the experiment. One can measure the goodness of causal fit of a bnet by doing do-intervention experiments (See Chapter 26).

[1]The Likelihood Ratio Test is discussed in Section Maximum Likelihood Estimate, Likelihood Ratio Test.

Chapter 28

Hidden Markov Model

A dynamical Bayesian network (DBN) (see Chapter 20) is a generalization of a Hidden Markov Model (HMM), which in turn is a generalization of a Kalman Filter (KF) (see Chapter 34).

See Wikipedia article Ref.[88] to learn about the history and many uses of HMMs. This chapter is based on Refs.[30], [88], [130], [63].

In this chapter, we use the following conventions.

Random variables are underlined and their values are not. For example, $\underline{a} = a$ means the random variable $\underline{a}$ takes the value a. Diagrams with nodes that are underlined represent Bayesian Networks (bnets) and the same diagram with the letters not underlined represents a specific **instantiation** of that bnet. For example $\underline{a} \to \underline{b}$ represents the bnet with conditional probability distribution $P(b|a)$, whereas $a \to b$ represents $P(b|a)$ itself.

If $\underline{a}$ is a root node, then $\sum a$ signifies a weighted sum $\sum_a P(a)$. For example, $\sum a \to b = \sum_a P(a)P(b|a)$. If $\underline{a}$ is not a root node as in $x \to \sum a \to y = \sum_a P(y|a)P(a|x)$, then $\sum a$ signifies a simple unweighted sum $\sum_a$.

Unobserved nodes are indicated by enclosing them in a dashed circle. For example, $\overset{\frown}{\underset{\smile}{(\underline{u})}}$.

Figure 28.1: HMM bnet with $n = 4$.

Suppose

$\underline{v}^n = (\underline{v}_0, \underline{v}_1, \ldots, \underline{v}_{n-1})$ are n visible nodes that are measured, and

$\underline{x}^n = (\underline{x}_0, \underline{x}_1, \ldots, \underline{x}_{n-1})$ are the n hidden, unmeasurable state nodes of a system that is being monitored.

For the bnet of Fig.28.1, one has

$$P(x^n, v^n) = \prod_{t=0}^{n-1} P(x_t|x_{t-1})P(v_t|x_t) \ , \tag{28.1}$$

where $x_{-1} = \emptyset$.

We assume that this bnet is stationary. The following notation emphasizes that fact:

$\pi(x)$ = prior probability

$$\pi(x) = P(\underline{x}_0 = x) \tag{28.2}$$

$A(x|x')$ = transition matrix

$$A(x|x') = P(\underline{x}_t = x|\underline{x}_{t-1} = x') \tag{28.3}$$

$B(v|x)$ = emission probability

$$B(v|x) = P(\underline{v}_t = v|\underline{x}_t = x) \tag{28.4}$$

Let $x_{<t} = (x_0, x_1, \ldots, x_{t-1})$.
For $t = 0, 1, \ldots, n-1$, define
$\mathcal{F}_t(x_t)$=future measurements probability

$$\begin{aligned} \mathcal{F}_t(x_t) &= P(v_{>i}|x_t) && (28.5) \\ &= \begin{tikzcd} x_t \arrow[r] & \sum x_{>t} \arrow[d] \\ & v_{>t} \end{tikzcd} && (28.6) \end{aligned}$$

$\overline{\mathcal{F}}_t(x_t)$= past and present measurements probability

$$\begin{aligned} \overline{\mathcal{F}}_t(x_t) &= P(v_{<t}, v_t, x_t) && (28.7) \\ &= \begin{tikzcd} \sum x_{<t} \arrow[r] \arrow[d] & x_t \arrow[d] \\ v_{<t} & v_t \end{tikzcd} && (28.8) \end{aligned}$$

$\lambda_t(x_t)$= present measurement probability, (a.k.a. emission probability $B(v_t|x_t)$). $\lambda_t(x_t)$ is the likelihood of x_t.

$$\begin{aligned} \lambda_t(x_t) &= P(v_t|x_t) && (28.9) \\ &= \begin{tikzcd} x_t \arrow[d] \\ v_t \end{tikzcd} && (28.10) \end{aligned}$$

28.1 Calculating $P(x_t, v^n)$ and $P(x_t, x_{t+1}, v^n)$

Claim 38 *For $t \geq 0$,*

$$
\begin{aligned}
P(x_t, v^n) &= \overline{\mathcal{F}}_t(x_t)\mathcal{F}_t(x_t) && (28.11)\\
&= \begin{tikzcd}[column sep=small]
\sum x_{<t} \arrow[r] \arrow[d] & x_t \arrow[d] & x_t \arrow[r] & \sum x_{>t} \arrow[d] \\
v_{<t} & v_t & & v_{>t}
\end{tikzcd} . && (28.12)
\end{aligned}
$$

For $t > 0$,

$$
\begin{aligned}
P(x_{t-1}, x_t, v^n) &= \overline{\mathcal{F}}_{t-1}(x_{t-1})P(x_t|x_{t-1})\lambda_t(x_t)\mathcal{F}_t(x_t) && (28.13)\\
&= \begin{tikzcd}[column sep=small]
\sum x_{<t-1} \arrow[r] \arrow[d] & x_{t-1} \arrow[d] & x_{t-1} \arrow[r] & x_t & x_t \arrow[d] & x_t \arrow[r] & \sum x_{>t} \arrow[d] \\
v_{<t-1} & v_{t-1} & & & v_t & & v_{>t}
\end{tikzcd} && (28.14)
\end{aligned}
$$

proof:

$$
\begin{aligned}
P(x_t, v^n) &= \sum_{x_{<i}} \sum_{x_{>i}} P(x^n, v^n) && (28.15)\\
&= \sum_{x_{<i}} \sum_{x_{>i}} P(x^n, v^n|x_t)P(x_t) && (28.16)\\
&= \sum_{x_{<i}} \sum_{x_{>i}} P(x_{<i}, v_{<i}, v_t|x_t)P(x_{>t}, v_{>i}|x_t)P(x_t) && (28.17)\\
&= P(v_{<i}, v_t|x_t)P(v_{>i}|x_t)P(x_t) && (28.18)\\
&= \overline{\mathcal{F}}_t(x_t)\mathcal{F}_t(x_t) && (28.19)
\end{aligned}
$$

$$
\begin{aligned}
P(x_{t-1}, x_t, v^n) &= \sum_{x_{<t-1}} \sum_{x_{>t}} P(x^n, v^n) && (28.20)\\
&= \sum_{x_{<t-1}} \sum_{x_{>t}} P(x^n, v^n|x_{t-1}, x_t)P(x_{t-1}, x_t) && (28.21)\\
&= \sum_{x_{<t-1}} \sum_{x_{>t}} P(x_{<t-1}, v_{<t-1}, v_{t-1}|x_{t-1})P(v_t|x_t)P(x_{t-1}, x_t)P(x_{>i}, v_{>i}|x_t) && (28.22)\\
&= P(v_{<t-1}, v_{t-1}|x_{t-1})P(v_t|x_t)P(x_{t-1}, x_t)P(v_{>i}|x_t) && (28.23)\\
&= \overline{\mathcal{F}}_{t-1}(x_{t-1})\lambda_t(x_t)P(x_t|x_{t-1})\mathcal{F}_t(x_t) && (28.24)
\end{aligned}
$$

QED

28.2 Calculating $\mathcal{F}_t$ and $\overline{\mathcal{F}}_t$

Claim 39 *For $t > 0$, $\mathcal{F}_t$ and $\overline{\mathcal{F}}_t$ can be calculated recursively as follows:*

$$\overline{\mathcal{F}}_t(x_t) = \sum_{x_{t-1}} \overline{\mathcal{F}}_{t-1}(x_{t-1}) P(x_t|x_{t-1}) \lambda_t(x_t) \tag{28.25}$$

$$= \sum_{x_{t-1}} \begin{tikzcd} \sum x_{<t-1} \arrow[r] \arrow[d] & x_{t-1} \arrow[d] & x_{t-1} \arrow[r] & x_t \arrow[d] & x_t \\ v_{<t-1} & v_{t-1} & & v_t & \end{tikzcd} \tag{28.26}$$

and

$$\mathcal{F}_{t-1}(x_{t-1}) = \sum_{x_t} P(x_t|x_{t-1}) \lambda_t(x_t) \mathcal{F}_t(x_t) \tag{28.27}$$

$$= \sum_{x_t} \begin{tikzcd} x_{t-1} \arrow[r] & x_t \arrow[d] & x_t \arrow[r] & \sum x_{>t} \arrow[d] \\ & v_t & & v_{>t} \end{tikzcd} \tag{28.28}$$

proof:

$$\overline{\mathcal{F}}_t(x_t)\mathcal{F}_t(x_t) = P(x_t, v^n) \tag{28.29}$$

$$= \sum_{x_{t-1}} P(x_{t-1}, x_t, v^n) \tag{28.30}$$

$$= \sum_{x_{t-1}} \overline{\mathcal{F}}_{t-1}(x_{t-1}) \lambda_t(x_t) P(x_t|x_{t-1}) \mathcal{F}_t(x_t) \tag{28.31}$$

$$\overline{\mathcal{F}}_{t-1}(x_{t-1})\mathcal{F}_{t-1}(x_{t-1}) = P(x_{t-1}, v^n) \tag{28.32}$$

$$= \sum_{x_t} P(x_{t-1}, x_t, v^n) \tag{28.33}$$

$$= \sum_{x_t} \overline{\mathcal{F}}_{t-1}(x_{t-1}) \lambda_t(x_t) P(x_t|x_{t-1}) \mathcal{F}_t(x_t) \tag{28.34}$$

QED

28.3 Calculating $P(x^n|v^n)$

Claim 40

$$P(x_t|x_{t-1}, v^n) = \frac{P(x_t|x_{t-1})\lambda_t(x_t)\mathcal{F}_t(x_t)}{\mathcal{F}_{t-1}(x_{t-1})} \tag{28.35}$$

$$= \frac{\begin{tikzcd} x_{t-1} \arrow[r] & x_t \arrow[d] & x_t \arrow[r] & \sum x_{>t} \arrow[d] \\ & v_t & & v_{>t} \end{tikzcd}}{\begin{tikzcd} x_{t-1} \arrow[r] & \sum x_t \arrow[r] \arrow[d] & \sum x_{>t} \arrow[d] \\ & v_t & v_{>t} \end{tikzcd}} \tag{28.36}$$

Note that actually, $P(x_t|x_{t-1}, v^n) = P(x_t|x_{t-1}, v_{\geq t})$ by d-separation, but we won't use this fact.

$$P(x_{t-1}|x_t, v^n) = \frac{\overline{\mathcal{F}}_{t-1}(x_{t-1})P(x_t|x_{t-1})\lambda_t(x_t)}{\overline{\mathcal{F}}_t(x_t)} \tag{28.37}$$

$$= \frac{\begin{tikzcd} \sum x_{<t-1} \arrow[r] \arrow[d] & x_{t-1} \arrow[d] & x_{t-1} \arrow[r] & x_t & x_t \arrow[d] \\ v_{<t-1} & v_{t-1} & & & v_t \end{tikzcd}}{\begin{tikzcd} \sum x_{<t-1} \arrow[r] \arrow[d] & \sum x_{t-1} \arrow[r] \arrow[d] & x_t \arrow[d] \\ v_{<t-1} & v_{t-1} & v_t \end{tikzcd}} \tag{28.38}$$

Note that actually $P(x_{t-1}|x_t, v^n) = P(x_{t-1}|x_t, v_{\leq t-1})$ by d-separation, but we won't use this fact.

proof:

$$P(x_t|x_{t-1}, v^n) = \frac{P(x_{t-1}, x_t, v^n)}{P(x_{t-1}, v^n)} \tag{28.39}$$

$$= \frac{\overline{\mathcal{F}}_{t-1}(x_{t-1})\lambda_t(x_t)P(x_t|x_{t-1})\mathcal{F}_t(x_t)}{\overline{\mathcal{F}}_{t-1}(x_{t-1})\mathcal{F}_{t-1}(x_{t-1})} \tag{28.40}$$

Analogous proof for Eq.(28.38).
QED

$$
\begin{aligned}
P(x^n|v^n) &= \prod_{t=1,\ldots,n-1,n} P(x_t|x_{t-1}, v^n) \quad \text{(forward propagation)} && (28.41)\\
&= \prod_{t=n+1,n,\ldots 3,2} P(x_{t-1}|x_t, v^n) \quad \text{(backward propagation)} && (28.42)
\end{aligned}
$$

28.4 Calculating $P(v^n|A, B, \pi)$

$P(v^n|A, B, \pi)$ can be calculated

- from $\overline{\mathcal{F}}_{n-1}(x_{n-1})$ (past of $\underline{x}_{n-1}$)

$$
\begin{aligned}
P(v^n|A, B, \pi) &= \sum_{x_{n-1}} \underbrace{P(v_{<n-1}, v_{n-1}, x_{n-1})}_{\overline{\mathcal{F}}_{n-1}(x_{n-1})} && (28.43)\\
&= \begin{array}{ccc} \sum x_{<n-1} & \longrightarrow & \sum x_{n-1} \\ \downarrow & & \downarrow \\ v_{<n-1} & & v_{n-1} \end{array} && (28.44)
\end{aligned}
$$

- from $\mathcal{F}_0(x_0)$ (future of $\underline{x}_0$)

$$
\begin{aligned}
P(v^n|A, B, \pi) &= \sum_{x_0} P(x_0)P(v_0|x_0)\underbrace{P(v_{>0}|x_0)}_{\mathcal{F}_0(x_0)} && (28.45)\\
&= \begin{array}{ccc} \sum x_0 & \longrightarrow & \sum x_{>0} \\ \downarrow & & \downarrow \\ v_0 & & v_{>0} \end{array} && (28.46)
\end{aligned}
$$

- from $\overline{\mathcal{F}}_t(x_t)$ and $\mathcal{F}_t(x_t)$ for some t (past and future of $\underline{x}_t$)

$$
\begin{aligned}
P(v^n|A, B, \pi) &= \sum_{x_t} \underbrace{P(v_{<t}, v_t, x_t)}_{\overline{\mathcal{F}}_t(x_t)}\underbrace{P(v_{>t}|x_t)}_{\mathcal{F}_t(x_t)} && (28.47)\\
&= \begin{array}{ccccc} \sum x_{<t} & \longrightarrow & \sum x_t & \longrightarrow & \sum x_{>t} \\ \downarrow & & \downarrow & & \downarrow \\ v_{<t} & & v_t & & v_{>t} \end{array} . && (28.48)
\end{aligned}
$$

28.5 Calculating $\hat{x}^n$ (Viterbi algorithm)

x^n is not visible, only v^n is. Here is how to find an estimate $\hat{x}^n$ of x^n.

Define

$$\overline{\mathcal{F}}_t^{max}(x_t) = \max_{x_{<t}} P(v_{<t}, v_t, x_{<t}, x_t) \tag{28.49}$$

$$= \begin{array}{ccc} \max x_{<t} & \longrightarrow & x_t \\ \downarrow & & \downarrow \\ v_{<t} & & v_t \end{array} \tag{28.50}$$

$$\hat{x}_{t-1}(x_t) = \operatorname*{argmax}_{x_{t-1}} \overline{\mathcal{F}}_{t-1}^{max}(x_{t-1}) P(x_t|x_{t-1}) \lambda_t(x_t) \tag{28.51}$$

$$= \begin{array}{ccccc} \max x_{<t-1} & \longrightarrow & \operatorname{argmax} x_{t-1} & \longrightarrow & x_t \\ \downarrow & & \downarrow & & \downarrow \\ v_{<t-1} & & v_{t-1} & & v_t \end{array} \tag{28.52}$$

For $t > 0$, $\overline{\mathcal{F}}_t^{max}$ and $\hat{x}_{t-1}(x_t)$ can be calculated recursively as follows:

$$\overline{\mathcal{F}}_t^{max}(x_t) = \max_{x_{t-1}} \overline{\mathcal{F}}_{t-1}^{max}(x_{t-1}) P(x_t|x_{t-1}) \lambda_t(x_t) \tag{28.53}$$

$$= \max_{x_{t-1}} \quad \begin{array}{ccccccc} \max x_{<t-1} & \longrightarrow & x_{t-1} & x_{t-1} & \longrightarrow & x_t & x_t \\ \downarrow & & \downarrow & & & & \downarrow \\ v_{<t-1} & & v_{t-1} & & & & v_t \end{array} \tag{28.54}$$

$$\hat{x}_{t-1}(x_t) = \operatorname*{argmax}_{x_{t-1}} \overline{\mathcal{F}}_{t-1}^{max}(x_{t-1}) P(x_t|x_{t-1}) \lambda_t(x_t) \tag{28.55}$$

$$= \operatorname*{argmax}_{x_{t-1}} \quad \begin{array}{ccccccc} \max x_{<t-1} & \longrightarrow & x_{t-1} & x_{t-1} & \longrightarrow & x_t & x_t \\ \downarrow & & \downarrow & & & & \downarrow \\ v_{<t-1} & & v_{t-1} & & & & v_t \end{array} \tag{28.56}$$

Claim 41 *(Viterbi Algorithm)*

If

$$\hat{x}^n = \operatorname*{argmax}_{x^n} P(x^n|v^n) \ , \tag{28.57}$$

then the components of $\hat{x}^n$ can be calculated recursively from the last one $\hat{x}_{n-1}$ to the first one $\hat{x}_0$, as follows. Let

$$\hat{x}_{n-1} = \underset{x_{n-1}}{\operatorname{argmax}} \overline{\mathcal{F}}^{max}_{n-1}(x_{n-1}) \tag{28.58}$$

$$= \begin{array}{ccc} \max x_{<n-1} & \longrightarrow & \operatorname{argmax} x_{n-1} \\ \downarrow & & \downarrow \\ v_{<n-1} & & v_{n-1} \end{array} \tag{28.59}$$

and for $t < n-1$, use

$$\hat{x}_{t-1} = \hat{x}_{t-1}(\hat{x}_t) \tag{28.60}$$

$$= \begin{array}{ccccc} \max x_{<t-1} & \longrightarrow & \operatorname{argmax} x_{t-1} & \longrightarrow & \hat{x}_t \\ \downarrow & & \downarrow & & \downarrow \\ v_{<t-1} & & v_{t-1} & & v_t \end{array} \tag{28.61}$$

proof:
QED

28.6 Calculating $\hat{A}$, $\hat{B}$, $\hat{\pi}$ (Baum-Welch algorithm)

Let $\theta = (A, B, \pi)$. θ is a set of hidden parameters. Here is how to find an estimate $\hat{\theta}$ of θ.

If x^n and v^n were visible, we could use

$$\hat{\pi}(x) = \mathbb{1}(\underline{x}_0 = x) \tag{28.62}$$

$$\hat{A}(x'|x) = \frac{\sum_{t=0}^{n-2} \mathbb{1}(\underline{x}_t = x, \underline{x}_{t+1} = x')}{\sum_x numerator} \tag{28.63}$$

$$\hat{B}(v|x) = \frac{\sum_{t=0}^{n-1} \mathbb{1}(\underline{v}_t = v, \underline{x}_t = x)}{\sum_v numerator} \tag{28.64}$$

But x^n is not visible. So how can we estimate θ under those circumstances?

Define

$$
\begin{aligned}
\gamma_t(x_t) = \; &= \; P(x_t|v^n) && (28.65)\\
&= \; \frac{P(x_t, v^n)}{P(v^n)} && (28.66)\\
&= \; \frac{\overline{\mathcal{F}}_t(x_t)\mathcal{F}_t(x_t)}{\sum_{x_t} numerator} && (28.67)\\
&= \; \frac{\begin{array}{ccccc} \sum x_{<t} & \longrightarrow & x_t & x_t & \longrightarrow \sum x_{>t} \\ \downarrow & & \downarrow & & \downarrow \\ v_{<t} & & v_t & & v_{>t} \end{array}}{\sum_{x_t} numerator} && (28.68)
\end{aligned}
$$

$$
\begin{aligned}
\xi_t(x_t, x_{t+1}) \; &= \; \frac{P(x_t, x_{t+1}, v^n)}{\sum_{x_t}\sum_{x_{t+1}} numerator} && (28.69)\\
&= \; \frac{\overline{\mathcal{F}}_{t-1}(x_{t-1})P(x_t|x_{t-1})\lambda_t(x_t)\mathcal{F}_t(x_t)}{\sum_{x_t}\sum_{x_{t+1}} numerator} && (28.70)\\
&= \; \frac{\begin{array}{ccccccc} \sum x_{<t-1} & \longrightarrow & x_{t-1} & x_{t-1} & \longrightarrow & x_t \;\; x_t & x_t \longrightarrow \sum x_{>t} \\ \downarrow & & \downarrow & & & \downarrow & \downarrow \\ v_{<t-1} & & v_{t-1} & & & v_t & v_{>t} \end{array}}{\sum_{x_t}\sum_{x_{t+1}} numerator} && (28.71)
\end{aligned}
$$

Claim 42 *(Baum-Welch algorithm)*
If

$$
\hat{\theta} = \operatorname*{argmax}_{\theta} P(v^n|\theta) \; , \tag{28.72}
$$

then we can find $\hat{\theta}$ using the following formulae:

$$
\hat{\pi}(x) = \overbrace{\gamma_0(x)}^{\sim P(x)} \tag{28.73}
$$

$$
\hat{A}(x'|x) = \frac{\sum_{t=0}^{n-2} \overbrace{\xi_t(x, x')}^{\sim P(x,x')}}{\sum_{t=0}^{n-2} \underbrace{\gamma_t(x)}_{P(x)}} \tag{28.74}
$$

$$
\hat{B}(v|x) = \frac{\sum_{t=0}^{n-1} \overbrace{\mathbb{1}(\underline{v}_t = v)}^{\sim P(v|x)} \overbrace{\gamma_t(x)}^{\sim P(x)}}{\sum_{t=0}^{n-1} \underbrace{\gamma_t(x)}_{\sim P(x)}} \tag{28.75}
$$

proof:
QED

Chapter 29

Influence Diagrams & Utility Nodes

Influence diagrams are just arbitrary bnets enhanced with a new kind of node called an utility node. The rest of this brief chapter will be devoted to discussing utility nodes.

Suppose $U(x)$ is a deterministic function $U : S_{\underline{x}} \to \mathbb{R}$ called the **utility function**. Then the **expected utility** is defined as

$$
\begin{aligned}
E_{\underline{U}}[\underline{U}] &= \sum_{U} P(U)U && (29.1)\\
&= \sum_{x}\sum_{U} \underbrace{P(U|x)}_{\delta[U,U(x)]} P(x)U && (29.2)\\
&= \sum_{x} P(x)U(x)\ . && (29.3)
\end{aligned}
$$

An **utility node** can be understood as a node composed of 3 simpler bnet nodes. This is illustrated in Fig.29.1.

Figure 29.1: An utility node can be understood as a node composed of 3 simpler bnet nodes.

The TPMs, printed in blue, for the nodes of Fig.29.1, are as follows:

$$P(U|pa(U)) = \delta[U, U(pa(U))] \ , \tag{29.4}$$

where if $U : S_{\underline{x}} \to \mathbb{R}$, then $\underline{x} = pa(\underline{U})$.

$$P(u|pa(U)) = \delta[u, U(pa(U))] \tag{29.5}$$

Node $\underline{\mu}_u$ calculates the expected value (mean value) of $\underline{u}$:

$$P(\mu_u) = \delta(\mu_u, E_{\underline{u}}[\underline{u}]) \tag{29.6}$$

Node $\underline{\sigma}_u$ calculates the standard deviation of $\underline{u}$:

$$P(\sigma_u) = \delta(\sigma_U, \sqrt{E_{\underline{u}}[(\underline{u} - E_{\underline{u}}[\underline{u}])^2]}) \tag{29.7}$$

Note that in order to calculate expected values, it is necessary that $\underline{U}, \underline{u} \in \mathbb{R}$. Note that nodes $\underline{u}$, $\underline{\mu}_u$, $\underline{\sigma}_u$ must all 3 have access to the TPM $P(U|pa(U))$ of node $\underline{U}$. In fact, in order to calculate $E_{\underline{u}}[\cdot]$, it is necessary for nodes $\underline{\mu}_u$ and $\underline{\sigma}_u$ to have access not just to $P(U|pa(U))$ but also to $P(pa(U))$.

See Fig.29.2. An influence diagram may have multiple utility nodes ($\underline{U}_1$ and $\underline{U}_2$ in Fig.29.2). Then one can define a merging utility node $\underline{U}$ that sums the values of all the other utility nodes.

Figure 29.2: An influence diagram may have multiple utility nodes, say $\underline{U}_1$ and $\underline{U}_2$. Then one can define an utility node $\underline{U} = \underline{U}_1 + \underline{U}_2$.

For the node $\underline{U}$ of Fig.29.2,

$$P(U|U_1, U_2) = \delta(U, U_1 + U_2) \tag{29.8}$$

Chapter 30

Instrumental Inequality and beyond

This chapter is based on Refs. [7] and [33].

Instrumental Variables (IVs) are discussed in Chapter 31. This chapter will discuss the original Instrumental inequality (I-inequality) discovered by Pearl, and other related inequalities. The I-inequality arises in bnets that use an IV. The I-inequality bounds the effect that an IV $\underline{z}$ can have on the outcome $\underline{y}$ of a treatment $\underline{d} \to \underline{y}$. Since there is a path $\underline{z} \to \underline{d} \to \underline{y}$, the treatment dose $\underline{d}$ acts as a mediator between the IV $\underline{z}$ and the treatment outcome $\underline{y}$. The I-inequality is reminiscent of the data processing inequality $H(\underline{z} : \underline{y}) \leq H(\underline{d} : \underline{y})$ which is valid for a simple Markov chain bnet $\underline{z} \to \underline{d} \to \underline{y}$. The data processing inequality is saying that the endpoint $\underline{y}$ receives more information from $\underline{d}$ than from $\underline{z}$. This is reasonable, since $\underline{y}$ is "closer" to $\underline{d}$ than to $\underline{z}$.

30.1 I-inequality

Figure 30.1: In bnet G, an IV $\underline{z}$ acts on a treatment $\underline{d} \to \underline{y}$. Bnet $\tilde{G}$ is obtained by applying an imagine operator to arrow $\underline{d} \to \underline{y}$ of bnet G.

Claim 43 *The TPMs for the bnet G in Fig.30.1 satisfy*

$$\max_d \sum_y \max_z P(d, y|z) \leq 1 \tag{30.1}$$

proof:

Below, any probability that alludes to a value $\tilde{d}$ refers to bnet $\tilde{G}$. Otherwise, if it doesn't allude to $\tilde{d}$, then it refers to G (or to $\tilde{G}$, since the TPMs of $\tilde{G}$ are defined from those of G in a consistent manner.)

G satisfies

$$P(d, y|z) = \sum_u P(u)P(y|u, d)P(d|u, z) \ , \tag{30.2}$$

and $\tilde{G}$ satisfies

$$P(d, y|z, \tilde{d}) = \sum_u P(u)P(y|u, \tilde{d})P(d|u, z) \ . \tag{30.3}$$

Note that Eqs.(30.2) and (30.3) imply that

$$P(d, y|z, d) = P(d, y|z) \tag{30.4}$$

and that

$$\boxed{P(\tilde{d}, y|z, \tilde{d}) \leq \sum_d P(d, y|z, \tilde{d}) = P(y|\tilde{d})} \ . \tag{30.5}$$

Thus,

$$\begin{aligned}
\max_{\tilde{d}} \sum_y \max_z P(\tilde{d}, y|z, \tilde{d}) &\leq \max_{\tilde{d}} \sum_y \max_z P(y|\tilde{d}) && (30.6)\\
&\leq \max_{\tilde{d}} \sum_y P(y|\tilde{d}) && (30.7)\\
&\leq \max_{\tilde{d}} 1 && (30.8)\\
&\leq 1 && (30.9)
\end{aligned}$$

QED

As pointed out in Ref.[7] from which I learned the above proof, the above proof is highly generalizable.

Fig.30.2 gives a graphical representation of the boxed Eq.(30.5) which is crucial to the proof.

And here is a meta-description of the steps in the proof:

Figure 30.2: Graphical representation of the boxed equation Eq.(30.5).

1. Use imagine operator to create a non-negative matrix $M_{d,\tilde{d}}$.
2. Use fact that row or column sum of $M_{d,\tilde{d}}$ is larger than diagonal element in sum: $\sum_d M_{d,\tilde{d}} \geq M_{\tilde{d},\tilde{d}}$.

30.1.1 I-inequality for binary z,d,y

It is enlightening to write down the I-inequality for the special case that $\underline{z}, \underline{d}, \underline{y}$ are binary.

$$P(d=1, y|z) = \begin{array}{c|c|c} & z=0 & z=1 \\ \hline y=0 & A & B \\ \hline y=1 & C & D \end{array}$$

$$A + D \leq 1$$
$$B + C \leq 1$$

Figure 30.3: I-inequality for binary $\underline{z}, \underline{d}, \underline{y}$. The same picture except with $d = 0$ is also true.

In the binary case, the I-inequality implies 4 different inequalities. These are as follows. One gets two inequalities by setting $d = 1$ in the next 2 equations.

$$\sum_{y=0}^{1} \sum_{z=0}^{1} \mathbb{1}(y = z) P(d, y|z) \ , \tag{30.10a}$$

$$\sum_{y=0}^{1} \sum_{z=0}^{1} \mathbb{1}(y \neq z) P(d, y|z) \ . \tag{30.10b}$$

One gets an additional 2 inequalities by setting $d = 0$ in Eqs.(30.10). These 4 inequalities are illustrated in Fig.30.3.

What do they mean? That at fixed $\underline{d}$, the correlation between $\underline{z}$ and $\underline{y}$ is limited.

30.2 Bounds on Effect of IV on treatment outcome y

Figure 30.4: Bnet $\mathcal{G}$ is obtained from the bnet G in Fig.30.1 by adding to G an arrow from the IV $\underline{z}$ to the treatment outcome $\underline{y}$. Bnet $\mathcal{G}_{do}$ is obtained by applying a do operator to node $\underline{d}$ of $\mathcal{G}$. Bnet $\mathcal{G}_{im}$ is obtained by applying an imagine operator to arrow $\underline{d} \to \underline{y}$ of $\mathcal{G}$.

In this section, we will assume that random variables $\underline{z}, \underline{d}, \underline{y}$ are binary. Just as with the binary case of the I-inequality, we will find an inequality for each value of $\underline{d} \in \{0, 1\}$.

Below, we will use the following 3 shorthand notations:

$$P_{y|z}(d) = P(d, y|z) \ , \tag{30.11}$$

$$P_{|z}(d) = \sum_{y} P(d, y|z) \ , \tag{30.12}$$

and

$$\pi_{|z}(d) = 1 - P_{|z}(d) \ . \tag{30.13}$$

For the bnet $\mathcal{G}_{do}$ in Fig.30.4, define the IV effect at fixed $\mathcal{D}\underline{d} = \tilde{d}$ by

$$IVE(\tilde{d}) = P(y = 1|z = 1, \mathcal{D}\underline{d} = \tilde{d}) - P(y = 1|z = 0, \mathcal{D}\underline{d} = \tilde{d}) \ . \tag{30.14}$$

Claim 44 *The TPMs for the bnet $\mathcal{G}_{do}$ in Fig.30.4 satisfy*

$$\pi_{|0}(d) \leq \left[IVE(d) - \{P_{1|1}(d) - P_{1|0}(d)\}\right] \leq \pi_{|1}(d) \tag{30.15}$$

proof:

$$
\begin{aligned}
P(y|z, \mathcal{D}\underline{d} = \tilde{d}) &= \sum_u P(u) P(y|u, z, \tilde{d}) && (30.16)\\
&= \sum_u P(u) \sum_d P(d, y|u, z, \tilde{d}) && (30.17)\\
&\geq \sum_u P(u) P(\tilde{d}, y|u, z, \tilde{d}) && (30.18)\\
&= \sum_u P(u) P(\tilde{d}, y|u, z) && (30.19)\\
&= P_{y|z}(\tilde{d}) && (30.20)
\end{aligned}
$$

Next note that $P(d, y|z, \tilde{d}) \geq 0$, and $\sum_{d,y} P(d, y|z, \tilde{d}) = 1$. If we write a table for $P(d, y|z, \tilde{d})$ at fixed $z, \tilde{d}$ with row and column indices (d, y), then a partial sum of the entries of that table must be ≤ 1:

$$
\sum_{d \neq \tilde{d}} P(d, y|z, \tilde{d}) + \underbrace{\sum_{y'} P(\tilde{d}, y'|z, \tilde{d})}_{P_{|z}(\tilde{d})} \leq 1 \ . \tag{30.21}
$$

Using the definitions of $P_{|z}$ and $\pi_{|z}$, we can rewrite the last equation as

$$
\sum_{d \neq \tilde{d}} P(d, y|z, \tilde{d}) \leq \pi_{|z}(\tilde{d}) \ . \tag{30.22}
$$

Next note that

$$
\begin{aligned}
P(y|z, \mathcal{D}\underline{d} = \tilde{d}) &= \sum_u P(u) P(y|u, z, \tilde{d}) && (30.23)\\
&= \sum_u P(u) \sum_d P(d, y|u, z, \tilde{d}) && (30.24)\\
&= P(\tilde{d}, y|z, \tilde{d}) + \sum_{d \neq \tilde{d}} P(d, y|z, \tilde{d}) && (30.25)\\
&= P_{y|z}(\tilde{d}) + \sum_{d \neq \tilde{d}} P(d, y|z, \tilde{d}) && (30.26)\\
&\leq P_{y|z}(\tilde{d}) + \pi_{|z}(\tilde{d}) \ . && (30.27)
\end{aligned}
$$

Hence,

$$
P_{y|z}(\tilde{d}) \leq P(y|z, \mathcal{D}\underline{d} = \tilde{d}) \leq P_{y|z}(\tilde{d}) + \pi_{|z} \tag{30.28}
$$

$$P_{1|1}(\tilde{d}) \leq P(y=1|z=1, \mathcal{D}\underline{d}=\tilde{d}) \leq P_{1|1}(\tilde{d}) + \pi_{|1} \tag{30.29}$$

$$-P_{1|0}(\tilde{d}) - \pi_{|0} \leq -P(y=1|z=0, \mathcal{D}\underline{d}=\tilde{d}) \leq -P_{1|0}(\tilde{d}) \tag{30.30}$$

QED

Chapter 31

Instrumental Variables

This chapter is based on Refs.[6] and [91].

The theory of potential outcomes (PO) discussed in Chapter 52 assumes that confounders can be ignored by conditioning on them. However, there are cases when that is not possible, as when there are some unmeasured (i.e., unobserved, hidden) confounder nodes in the bnet, because one can only condition on observed random variables, by definition. So what if confounders can't be ignored? Are we then precluded from using PO theory? Not necessarily. It might still be possible to use PO theory if one can find a suitable instrumental variable (IV) for the problem.

IVs were actually invented by Sewall Wright and his father Philip Wright long before PO theory was invented by Rubin. The reason why IVs save PO theory is greatly clarified by using Pearl causal DAGs and his d-separation theorem (see Chapter 17).

Most of the discussion in this chapter is limited to LDEN (linear deterministic bnets with external noise). These are discussed in Chapter 36. However, as will become obvious to the reader, IVs are also applicable and useful in general bnet modeling.

In this chapter, for any random variable $\underline{x}$, we will represent the linear operator $\langle \underline{x}, \underline{x} \rangle^{-1} \langle \underline{x}, \cdot \rangle$ as a derivative:

$$\frac{d\,\cdot}{d\underline{x}} = \frac{\langle \underline{x}, \cdot \rangle}{\langle \underline{x}, \underline{x} \rangle} \ . \tag{31.1}$$

31.1 δ with unmeasured confounder

In this section, we explain using LDENs why unmeasured confounders prejudice PO calculations.

Consider the LDEN bnet of Fig.31.1, For some $\delta, \mu \in \mathbb{R}$, we have

$$\underline{y} = \delta \underline{d} + \underbrace{\mu \underline{h} + \underline{u}_{\underline{y}}}_{\underline{n}_y} \ . \tag{31.2}$$

Figure 31.1: An LDEN bnet. The direct path $\underline{d} \to \underline{y}$ is confounded by a hidden variable $\underline{h}$.

Note that

$$\left\langle \underline{y}, \underline{d} \right\rangle = \delta \left\langle \underline{d}, \underline{d} \right\rangle + \left\langle \underline{n}_{\underline{y}}, \underline{d} \right\rangle \ . \tag{31.3}$$

Therefore,

$$\delta = \frac{d\underline{y}}{d\underline{d}} - \frac{d\underline{n}_{\underline{y}}}{d\underline{d}} \ . \tag{31.4}$$

If the confounder $\underline{h}$ is measured, then we calculate the covariances at fixed $\underline{n}_{\underline{y}}$, and the conditional covariance $\left\langle \underline{n}_{\underline{y}}, \underline{d} \right\rangle_{|\underline{n}_{\underline{y}}} = 0$.

31.2 δ (with unmeasured confounder) can be inferred via IV

Figure 31.2: Two LDEN bnets. The direct path $\underline{d} \to \underline{y}$ is confounded by a hidden variable $\underline{h}$, but by using the IV $\underline{A}$, we are still able to identify (i.e. calculate) δ.

Now consider the two LDEN bnets shown in Fig.31.2. Note that there are no arrows $\underline{A} \to \underline{y}$ or $\underline{A} \to \underline{h}$. Note that node $\underline{d}$ is a collider in the path $\underline{A} - \underline{d} - \underline{h} - \underline{y}$, Therefore, the only unblocked path from $\underline{A}$ to $\underline{y}$ in G is $\underline{A} \to \underline{d} \to \underline{y}$ and that path has been removed in G_{im+}. These observations are encapsulated in the following statements.

$$\underline{d} \perp_G \underline{y} = \text{false}, \quad \underline{A} \perp_G \underline{y} = \text{false} \ . \tag{31.5}$$

$$\underline{d} \perp_{G_{im+}} \underline{y} = \text{false}, \quad \underline{A} \perp_{G_{im+}} \underline{y} = \text{true} \ . \tag{31.6}$$

The following is true for G:

$$\underline{y} = \delta \underline{d} + \underbrace{\mu \underline{h} + \underline{u}_{\underline{y}}}_{\underline{n}_{\underline{y}}} \tag{31.7}$$

$$\underline{d} = \alpha \underline{A} + \underbrace{\nu \underline{h} + \underline{u}_{\underline{d}}}_{\underline{n}_{\underline{d}}} \ . \tag{31.8}$$

Since $\left\langle \underline{n}_{\underline{y}}, \underline{A} \right\rangle = \langle \underline{n}_{\underline{d}}, \underline{A} \rangle = 0$ is true in G, we have

$$\left\langle \underline{y}, \underline{A} \right\rangle = \delta \left\langle \underline{d}, \underline{A} \right\rangle \tag{31.9}$$

and

$$\langle \underline{d}, \underline{A} \rangle = \alpha \left\langle \underline{A}, \underline{A} \right\rangle \ . \tag{31.10}$$

Note that $\left\langle \underline{y}, \underline{A} \right\rangle = \delta = 0$ for G_{im+} but not for G, so we are speaking about G from here on. It follows that

$$\alpha = \frac{d\underline{d}}{d\underline{A}} \tag{31.11}$$

and

$$\begin{aligned}
\delta &= \frac{\left\langle \underline{y}, \underline{A} \right\rangle}{\langle \underline{d}, \underline{A} \rangle} && (31.12)\\
&= \frac{\left\langle \underline{y}, \underline{A} \right\rangle}{\langle \underline{A}, \underline{A} \rangle} \frac{\langle \underline{A}, \underline{A} \rangle}{\langle \underline{d}, \underline{A} \rangle} && (31.13)\\
&= \frac{\frac{d\underline{y}}{d\underline{A}}}{\frac{d\underline{d}}{d\underline{A}}} \quad (\text{equal to } \frac{d\underline{y}}{d\underline{d}}) && (31.14)\\
&= \frac{d\underline{y}}{d(\alpha \underline{A})} \ . && (31.15)
\end{aligned}$$

Eq.(31.14) is illustrated in Fig.31.3.

31.3 More general bnets with IVs

Figs.31.4 and 31.5 are examples of other bnets for which the effect δ is identifiable thanks to the IV $\underline{A}$.

Figure 31.3: Effect $\delta = \frac{d\underline{y}}{d\underline{d}}$ as slope of line.

Figure 31.4: The 2 paths in G_{im+} from IV variable $\underline{A}$ to $\underline{y}$ are blocked by not conditioning on colliders $\underline{v}$ and $\underline{d}$. Thus, $\underline{d} \perp_{G_{im+}} \underline{y} =$ false, $\underline{A} \perp_{G_{im+}} \underline{y} =$ true

Figure 31.5: There are 2 paths in G_{im+} from IV variable $\underline{A}$ to $\underline{y}$. One is blocked by not conditioning on the collider $\underline{d}$ and the other can be blocked by conditioning on $\underline{v}$. Thus, $\underline{d} \perp_{G_{im+}} \underline{y}|\underline{v} =$ false, $\underline{A} \perp_{G_{im+}} \underline{y}|\underline{v} =$ true

31.4 Instrumental Inequality

Pearl's instrumental inequality and related inequalities are discussed in Chapter 30.

Chapter 32

Jackknife Resampling

This paper is based on Ref. [93].

Before reading this chapter, we recommend that you read the section entitled "Demystifying Population and Sample Variances", in Chapter Notational Conventions and Preliminaries.

Jackknife Resampling (JR) is a way of generating from an original list of n samples, a new list of n synthetic (i.e., man made, not occurring physically in Nature) samples obtained by deleting one of the samples from the original list and averaging over the rest.

Let $\Sigma = \{0, 1, 2, \ldots, n-1\}$. Let us consider the list of samples

$$\vec{x} = (x^\sigma)_{\sigma \in \Sigma} \ , \tag{32.1}$$

where the x^σ are assumed to be i.i.d. with

$$E[\underline{x}^\sigma] = \mu \tag{32.2}$$

and

$$\left\langle \underline{x}^\sigma, \underline{x}^{\sigma'} \right\rangle = V_1 \delta(\sigma, \sigma') \ . \tag{32.3}$$

If we define μ and V_1 estimators by

$$\hat{\mu} = \frac{1}{n} \sum_\sigma x^\sigma \tag{32.4a}$$

$$\hat{V}_1 = \frac{1}{n-1} \sum_\sigma (x^\sigma - \hat{\mu})^2 \ , \tag{32.4b}$$

then one can show that:

$$E[\underline{\hat{\mu}}] = \mu \ , \quad E[\underline{\hat{V}}_1] = V_1 \tag{32.5}$$

so both of these estimators are unbiased.

Now define lists of samples with one of the items in $\vec{x}$ deleted:

$$\vec{x}_\xi = (x^\sigma)_{\sigma\in\Sigma-\{\xi\}} \ . \tag{32.6}$$

Suppose we are given functions of $\vec{x}$ and $\vec{x}_\xi$

$$A = A^n(\vec{x}) \ , \tag{32.7}$$

$$A_\xi = A^{n-1}(\vec{x}_\xi) \ . \tag{32.8}$$

Then define a list $\vec{A}$ by

$$\vec{A} = (A_\xi)_{\xi\in\Sigma} \ . \tag{32.9}$$

One can also define a list $\vec{B}$ by:

$$\vec{B} = (B_\xi)_{\xi\in\Sigma} \tag{32.10}$$

where

$$\begin{aligned} B_\xi &= nA - (n-1)A_\xi & (32.11)\\ &= A_\xi - n[A_\xi - A] \ . & (32.12)\end{aligned}$$

Later on, we will see why the list $\vec{B}$ just defined is of interest.

Figure 32.1: Bnet for jackknife resampling (JR).

Fig.32.1 is a bnet that encapsulates JR. The TPMs, printed in blue, for the nodes of that bnet, are as follows:

$$P(\vec{x}_\xi|\vec{x}) = \mathbb{1}(\quad \vec{x}_\xi = \text{ defined by Eq.(32.6)} \quad) \tag{32.13}$$

$$P(A_\xi|\vec{x}_\xi) = \mathbb{1}(\quad A_\xi = \text{ defined by Eq.(32.8)} \quad) \tag{32.14}$$

$$P(B_\xi|\vec{A}_\xi, A) = \mathbb{1}(\quad B_\xi = \text{ defined by Eq.(32.12)} \quad) \tag{32.15}$$

32.1 Case $A = A^n(\vec{x}) = \frac{1}{n}\sum_\sigma x^\sigma$

Suppose

$$A = \underbrace{\frac{1}{n}\sum_\sigma x^\sigma}_{E_\sigma[x^\sigma]} \tag{32.16}$$

and

$$A_\xi = \underbrace{\frac{1}{n-1}\sum_{\sigma\in\Sigma-\{\xi\}} x^\sigma}_{E_{\sigma|\xi}[x^\sigma] \text{ where } P(\sigma|\xi)=\frac{\mathbb{1}(\sigma\neq\xi)}{n-1}} \quad . \tag{32.17}$$

Then

$$\frac{1}{n}\sum_\xi A_\xi = E_\xi E_{\sigma|\xi}[x^\sigma] = E_\sigma[x^\sigma] = A \tag{32.18}$$

Claim 45

$$E[\underline{A}_\xi] = \mu \tag{32.19}$$

$$\left\langle \underline{A}_\xi, \underline{A}_{\xi'} \right\rangle = V_1 \left[\frac{n-2-\delta(\xi,\xi')}{n-1}\right] \tag{32.20}$$

proof:

$$E[\underline{A}_\xi] = \frac{1}{n-1}\sum_\sigma \mathbb{1}(\xi\neq\sigma)E[\underline{x}^\sigma] = \mu \tag{32.21}$$

$$\begin{aligned}
\left\langle \underline{A}_\xi, \underline{A}_{\xi'} \right\rangle &= \frac{1}{n-1}\sum_\sigma\sum_{\sigma'}[1-\delta(\sigma,\xi)][1-\delta(\sigma',\xi')]\left\langle x^\sigma, x^{\sigma'}\right\rangle && (32.22)\\
&= \frac{V_1}{n-1}\sum_\sigma\sum_{\sigma'}[1-\delta^\sigma_\xi-\delta^{\sigma'}_{\xi'}+\delta^\sigma_\xi\delta^{\sigma'}_{\xi'}]\delta^\sigma_{\sigma'} && (32.23)\\
&= \frac{V_1}{n-1}[n-2+\delta^\xi_{\xi'}] && (32.24)
\end{aligned}$$

QED

Figure 32.2: For each point x^ξ, there is a corresponding point A_ξ which is $n-1$ times closer to the average A than x^ξ is.

Claim 46

$$A_\xi - A = \frac{A - x^\xi}{n-1} \tag{32.25}$$

Hence, the distance of a point x^ξ to the mean value A is $n-1$ times as large as the distance of A_ξ to A. (see Fig.32.2)

proof:

$$\begin{aligned} A_\xi - A &= \frac{1}{n-1}\left(\sum_\sigma x^\sigma - x^\xi\right) - \frac{1}{n}\sum_\sigma x^\sigma && (32.26)\\ &= x^\xi\left(\frac{-1}{n-1}\right) + \sum_\sigma x^\sigma \underbrace{\left(\frac{1}{n-1} - \frac{1}{n}\right)}_{\frac{1}{n(n-1)}} && (32.27)\\ &= \frac{A - x^\xi}{n-1} && (32.28) \end{aligned}$$

QED

Note that

$$(n-1)\sum_\xi (A_\xi - E_\xi[A_\xi])^2 = \hat{V}_1 \tag{32.29}$$

by Eq.(32.25). Hence, we can estimate V_1 from $\vec{A}$ instead of $\vec{x}$.

Note that

$$\begin{aligned} B_\xi &= nA - (n-1)A_\xi && (32.30)\\ &= \sum_\sigma x^\sigma - \sum_{\sigma\neq\xi} x^\sigma = x^\xi && (32.31) \end{aligned}$$

Since $B_\xi = x^\xi$, they have identical statistics. In particular, one can use for B_ξ the same μ and V_1 estimators that we defined in Eqs.(32.4) for x^σ.

Chapter 33

Junction Tree Algorithm

The Junction Tree (JT) algorithm is an algo for evaluating exact marginals of a bnet, including cases in which some nodes are fixed to a single state. (fixed nodes are called the a priori evidence.)

The JT algo starts by clustering the loops of a bnet into bigger nodes so as to transform the bnet into a polytree bnet. Then it applies Pearl Belief Propagation (see Chapter 41) to the ensuing polytree. The first breakthrough paper to achieve this agenda in full was Ref.[20] by Lauritzen, and Spiegelhalter in 1988. See the Wikipedia article Ref.[94] for more info and references on the JT algorithm.

I won't describe the JT algo any further here, because it would take too long for this brief book to give a complete treatment of it, including the mathematical proofs. If all you want to do is to code the JT algo, without delving into the mathematical theorems and proofs behind it, I strongly recommend Ref.[16]. Ref.[16] is an excellent cookbook for programmers of the JT algo. My open source program QuantumFog (see Ref.[57]) implements the JT algo in Python, following the recipe of Ref.[16].

Chapter 34

Kalman Filter

This chapter is based on Ref.[96], except we've replaced the variables F_t, w_t, v_t in that reference by A_t, ξ_t, ζ_t, respectively.

A Kalman Filter (KF) is a special case of a Hidden Markov Model. HMMs are discussed in Chapter 28.

Figure 34.1: Kalman Filter (KF) bnet.

Let

$t = 0, 1, 2, \ldots, T-1$ be the time.

$\underline{\xi}_t \in \mathbb{R}^{nx}, \underline{\zeta}_t \in \mathbb{R}^{nz}$ be random variables that represent hidden (unobserved) external Gaussian white noise.

$\underline{x}_t \in \mathbb{R}^{nx}$ be random variables that represent the hidden (unobserved) true state of the system.

$\underline{u}_t \in \mathbb{R}^{nu}, \underline{z}_t \in \mathbb{R}^{nz}$ be random variables that represent the measured (observed) state of the system.

The node TPMs, printed in blue, of the KF bnet Fig.34.1, are as follows:

$$P(\xi_t) = \mathcal{N}(\xi_t; 0, Q_t) \ , \tag{34.1}$$

where Q_t is given.

$$P(x_t|x_{t-1}, u_t, \xi_t) = \mathbb{1}(x_t = A_t x_{t-1} + B_t u_t + \xi_t) \ , \tag{34.2}$$

where $A_t, B_t u_t$ are given. $P(x_t|x_{t-1}, u_t, \xi_t)$ becomes $P(x_t)$ for $t = 0$.

$$P(\zeta_t) = \mathcal{N}(\zeta_t; 0, R_t) \ , \tag{34.3}$$

where R_t is given.

$$P(z_t|x_t, \zeta_t) = \mathbb{1}(z_t = H_t x_t + \zeta_t) \ , \tag{34.4}$$

where H_t is given.

34.1 Prediction Problem

Find $\hat{x}_t$ (the best possible estimate of x_t) and P_t (the state of the filter at time t) in terms of

1. $\hat{x}_{t-1}$ and P_{t-1}.
2. the 5 matrices $\mathcal{M}_t = (A_t, B_t, H_t, Q_t, R_t)$
3. the observed values of z_t and u_t.

See Fig.34.2. For that figure,

$$P(\hat{x}_t, P_t|\hat{x}_{t-1}, P_{t-1}, \mathcal{M}_t, z_t, u_t) = \delta(\hat{x}_t, ?)\delta(P_t, ?) \ . \tag{34.5}$$

$$\begin{tikzcd}
\mathcal{M}_0, \underline{z}_0, \underline{u}_0 \arrow[d] & \mathcal{M}_1, \underline{z}_1, \underline{u}_1 \arrow[d] & \mathcal{M}_2, \underline{z}_2, \underline{u}_2 \arrow[d] & \mathcal{M}_3, \underline{z}_3, \underline{u}_3 \arrow[d] \\
\underline{\hat{x}}_0, \underline{P}_0 \arrow[r] & \underline{\hat{x}}_1, \underline{P}_1 \arrow[r] & \underline{\hat{x}}_2, \underline{P}_2 \arrow[r] & \underline{\hat{x}}_3, \underline{P}_3
\end{tikzcd}$$

Figure 34.2: Evolution of $\hat{x}_t, P_t$ for a KF.

34.2 Solution

The algebraic solution of the prediction problem for a KF is as follows. See Fig.34.3 for a bnet representation of this algebraic solution.

Figure 34.3: Bnet representation of the algebraic solution of the prediction problem for a KF.

Define $\eta_{t|t} = \eta_t$ for $\eta = \hat{x}, P$.

- **Initial Conditions** $\hat{x}_0, P_0$
- **A priori estimates**

  a priori state estimate

$$\hat{x}_{t|t-1} = \underbrace{A_t\hat{x}_{t-1} + B_t u_t}_{x_t - \xi_t} \tag{34.6}$$

  a priori covariance estimate

$$P_{t|t-1} = A_t P_{t-1} A_t^T + Q_t \tag{34.7}$$

- **A posteriori estimates**

  Optimal Kalman gain K_t

$$S_t = H_t P_{t|t-1} H_t^T + R_t \tag{34.8}$$

$$
\begin{aligned}
K_t &= P_{t|t-1} H_t^T S_t^{-1} && (34.9)\\
&= P_{t|t-1} H_t^T \left[H_t P_{t|t-1} H_t^T + R_t \right]^{-1} && (34.10)
\end{aligned}
$$

a posteriori state estimate

$$
\tilde{y}_{t|t-1} = z_t - H_t \hat{x}_{t|t-1} \tag{34.11}
$$

$$
\begin{aligned}
\hat{x}_t &= \hat{x}_{t|t-1} + K_t \tilde{y}_{t|t-1} && (34.12)\\
&= (1 - K_t H_t)\hat{x}_{t|t-1} + K_t \underbrace{z_t}_{H_t x_t + \zeta_t} \quad \text{(interpolation formula)} && (34.13)
\end{aligned}
$$

$$
\tilde{y}_t = z_t - H_t \hat{x}_t \tag{34.14}
$$

a posteriori covariance estimate

$$
P_t = (I - K_t H_t)\, P_{t|t-1} \tag{34.15}
$$

34.3 Simple Example

r_t position, v_t velocity, a_t acceleration of point particle.

$$
x_t = A x_{t-1} + B u_t + \xi_t \tag{34.16}
$$

$$
x_t = \begin{bmatrix} r_t \\ v_t \end{bmatrix}, \quad u_t = a_t \tag{34.17}
$$

$$
A = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} \frac{1}{2}(\Delta t)^2 \\ \Delta t \end{bmatrix} \tag{34.18}
$$

$$
z_t = H x_t + \zeta_t \tag{34.19}
$$

$$
H = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{34.20}
$$

34.4 Invariants

Note that

$$x_t - \hat{x}_{t|t-1} = \xi_t \tag{34.21}$$

$$\begin{aligned} x_t - \hat{x}_t &= (1 - K_t H_t)(x_t - \hat{x}_{t|t-1}) - K_t \zeta_t & (34.22) \\ &= (1 - K_t H_t)\xi_t - K_t \zeta_t & (34.23) \end{aligned}$$

$$\begin{aligned} \tilde{y}_{t|t-1} &= \underbrace{z_t}_{H_t x_t + \zeta_t} - H_t \hat{x}_{t|t-1} & (34.24) \\ &= H_t \xi_t + \zeta_t & (34.25) \end{aligned}$$

$$\begin{aligned} \tilde{y}_t &= \underbrace{z_t}_{H_t x_t + \zeta_t} - H_t \hat{x}_t & (34.26) \\ &= H_t(x_t - \hat{x}_t) + \zeta_t & (34.27) \\ &= H_t(1 - K_t H_t)\xi_t + (1 - H_t K_t)\zeta_t & (34.28) \end{aligned}$$

$(x_t - \hat{x}_{t|t-1})$, $(x_t - \hat{x}_t)$, $\tilde{y}_{t|t-1}$ and $\tilde{y}_t$ are called **residuals**. Since ξ_t and ζ_t have zero mean value,

$$E[\underline{x}_t - \underline{\hat{x}}_t] = E[\underline{x}_t - \underline{\hat{x}}_{t|t-1}] = 0 \tag{34.29}$$

$$E[\underline{\tilde{y}}_t] = E[\underline{\tilde{y}}_{t|t-1}] = 0 \tag{34.30}$$

These zero mean value identities are called **invariants**.

34.5 Derivation of Solution

First, some notational conventions. Let

$$Cov(\underline{a})_{i,j} = \left\langle \underline{a}_i, \underline{a}_j \right\rangle \tag{34.31}$$

$$Cov(\underline{a}) = \left\langle \underline{a}, \underline{a}^T \right\rangle = \langle \underline{a}, tp. \rangle \tag{34.32}$$

$$[A, B]_+ = AB + B^T A^T \tag{34.33}$$

$$A + tp. = A + A^T \tag{34.34}$$

$tp.$ stands for transpose.

Now define

$$P_t = Cov(\underline{x}_t - \hat{\underline{x}}_t) \tag{34.35}$$

$$P_{t|t-1} = Cov(\underline{x}_t - \hat{\underline{x}}_{t|t-1}) \tag{34.36}$$

$$S_t = Cov(\tilde{\underline{y}}_{t|t-1}) \tag{34.37}$$

It follows that

$$\begin{aligned}
P_t &= \langle \underline{x}_t - \hat{\underline{x}}_t, tp. \rangle && (34.38)\\
&= \left\langle (1 - K_t H_t)(\underline{x}_t - \hat{\underline{x}}_{t|t-1}) - K_t \underline{\zeta}_t, tp. \right\rangle \quad \text{(by Eq.(34.23))} && (34.39)\\
&= \left\langle (1 - K_t H_t)(\underline{x}_t - \hat{\underline{x}}_{t|t-1}), tp. \right\rangle + \left\langle K_t \underline{\zeta}_t, tp. \right\rangle \ (\underline{\zeta}_t \text{ uncorrelated with } (\underline{x}_t - \hat{\underline{x}}_{t|t-1})) && (34.40)\\
&= (1 - K_t H_t) \underbrace{\langle \underline{x}_t - \hat{\underline{x}}_{t|t-1}, tp. \rangle}_{P_{t|t-1}} (1 - K_t H_t)^T + K_t \underbrace{\left\langle \underline{\zeta}_t, tp. \right\rangle}_{R_t} K_t^T && (34.41)\\
&= P_{t|t-1} - [K_t H_t, P_{t|t-1}]_+ + K_t \underbrace{\left(H_t P_{t|t-1} H_t^T + R_t \right)}_{S_t} K_t^T && (34.42)
\end{aligned}$$

Next we find the optimal Kalman gain K_t by minimizing with respect to K_t the following mean squared error.

$$\begin{aligned}
\mathcal{E} &= \sum_i E[(\underline{x}_t - \hat{\underline{x}}_t)_i^2] && (34.43)\\
&= \mathrm{tr} E\left[(\underline{x}_t - \hat{\underline{x}}_t)(\underline{x}_t - \hat{\underline{x}}_t)^T \right] && (34.44)\\
&= \mathrm{tr} P_t && (34.45)\\
&= \mathrm{tr}\left(P_{t|t-1} - [K_t H_t, P_{t|t-1}]_+ + K_t S_t K_t^T \right) && (34.46)
\end{aligned}$$

If we set to zero the variation of $\mathcal{E}$ when K_t varies, we get

$$0 = \delta \mathcal{E} = \mathrm{tr}\left[\left(-P_{t|t-1} H_t^T + K_t S_t \right) \delta K_t^T \right] + tp. \tag{34.47}$$

Hence

$$-P_{t|t-1} H_t^T + K_t S_t = 0 \tag{34.48}$$

$$K_t = P_{t|t-1} H_t^T S_t^{-1} \tag{34.49}$$

Chapter 35

Linear and Logistic Regression

Figure 35.1: Linear Regression

Figure 35.2: Bnet of Fig.35.1 with new $\vec{Y}$ node.

Estimators $\hat{y}$ for linear and logistic regression.

- **Linear Regression:** $y \in \mathbb{R}$. Note $\hat{y} \in \mathbb{R}$. $(x, \hat{y}(x))$ is the graph of a straight

line with y-intercept b and slope m.

$$\hat{y}(x; b, m) = b + mx \tag{35.1}$$

- **Logistic Regression:** $y \in \{0, 1\}$. Note $\hat{y} \in [0, 1]$. $(x, \hat{y}(x))$ is the graph of a sigmoid. Often in literature, b, m are replaced by β_0, β_1.

$$\hat{y}(x; b, m) = \text{smoid}(b + mx) \tag{35.2}$$

Define

$$V(b, m) = \sum_{x,y} P(x, y)|y - \hat{y}(x; b, m)|^2 \ . \tag{35.3}$$

We want to minimize $V(b, m)$ (called a cost or loss function) wrt b and m.

Node TPMs of Bnet of Fig.35.1 given next in blue.

$$P(b, m) = \text{given} \tag{35.4}$$

The first time it is used, (b, m) is arbitrary. After the first time, it is determined by previous stage.

Let

$$P_{\underline{x},\underline{y}}(x, y) = \frac{1}{nsam(\vec{x})} \sum_{\sigma} \mathbb{1}(x = x[\sigma], y = y[\sigma]) \ . \tag{35.5}$$

$$P(\vec{x}) = \prod_{\sigma} P(x[\sigma]) \tag{35.6}$$

$$P(\vec{y}|\vec{x}) = \prod_{\sigma} P(y[\sigma] \mid x[\sigma]) \tag{35.7}$$

$$P(\vec{\hat{y}}|\vec{x}, b, m) = \prod_{\sigma} \delta(\hat{y}[\sigma], \hat{y}(x[\sigma], b, m)) \tag{35.8}$$

$$P(V|\vec{\hat{y}}, \vec{y}) = \delta(V, \frac{1}{nsam(\vec{x})} \sum_{\sigma} |y[\sigma] - \hat{y}[\sigma]|^2) \tag{35.9}$$

Let $\eta_b, \eta_m > 0$. For $x = b, m$, if $x' - x = \Delta x = -\eta \frac{\partial V}{\partial x}$, then $\Delta V \approx \frac{-1}{\eta}(\Delta x)^2 \leq 0$ for $\eta > 0$. This is called "gradient descent".

$$P(b'|V, b) = \delta(b', b - \eta_b \partial_b V) \tag{35.10}$$

$$P(m'|V, m) = \delta(m', m - \eta_m \partial_m V) \tag{35.11}$$

35.1 Generalization to x with multiple components (features)

Suppose that for each sample σ, instead of $x[\sigma]$ being a scalar, it has n components called features:

$$x[\sigma] = (x_0[\sigma], x_1[\sigma], x_2[\sigma], \ldots x_{n-1}[\sigma]) \ . \tag{35.12}$$

Slope m is replaced by weights

$$w = (w_0, w_1, w_3, , \ldots, w_{n-1}) \ , \tag{35.13}$$

and the product of 2 scalars $mx[\sigma]$ is replaced by the inner vector product $w^T x[\sigma]$.

35.2 Alternative $V(b, m)$ for logistic regression

For logistic regression, since $y[\sigma] \in \{0, 1\}$ and $\hat{y}[\sigma] \in [0, 1]$ are both in the interval $[0, 1]$, they can be interpreted as probabilities. Define probability distributions $p[\sigma](x)$ and $\hat{p}[\sigma](x)$ for $x \in \{0, 1\}$ by

$$p[\sigma](1) = y[\sigma], \quad p[\sigma](0) = 1 - y[\sigma] \tag{35.14}$$

$$\hat{p}[\sigma](1) = \hat{y}[\sigma], \quad \hat{p}[\sigma](0) = 1 - \hat{y}[\sigma] \tag{35.15}$$

Then for logistic regression, the following 2 cost functions $V(b, m)$ can be used as alternatives to the cost function Eq.(35.3) previously given.

$$V(b, m) = \frac{1}{nsam(\vec{x})} \sum_{\sigma} D_{KL}(p[\sigma] \parallel \hat{p}[\sigma]) \tag{35.16}$$

and

$$\begin{aligned}
V(b, m) &= \frac{1}{nsam(\vec{x})} \sum_{\sigma} CE(p[\sigma] \parallel \hat{p}[\sigma]) && (35.17)\\
&= \frac{-1}{nsam(\vec{x})} \sum_{\sigma} \{y[\sigma] \ln \hat{y}[\sigma] + (1 - y[\sigma]) \ln(1 - \hat{y}[\sigma])\} && (35.18)\\
&= \frac{-1}{nsam(\vec{x})} \sum_{\sigma} \ln \left\{\hat{y}[\sigma]^{y[\sigma]} (1 - \hat{y}[\sigma])^{(1-y[\sigma])}\right\} && (35.19)\\
&= \frac{-1}{nsam(\vec{x})} \sum_{\sigma} \ln P(\underline{Y} = y[\sigma] \mid \hat{y} = \hat{y}[\sigma]) && (35.20)\\
&= -\sum_{x,y} P(x, y) \ln P(\underline{Y} = y | \hat{y} = \hat{y}(x, b, m)) && (35.21)
\end{aligned}$$

Above, we used

$$P(\underline{Y} = Y|\hat{y}) = \hat{y}^Y [1 - \hat{y}]^{1-Y} \tag{35.22}$$

for $Y \in S_{\underline{Y}} = \{0, 1\}$. (Bernoulli distribution).

There is no node corresponding to $\underline{Y}$ in the Bnet of Fig.35.1. Fig.35.2 shows a new Bnet that has a new node called $\vec{\underline{Y}}$ compared to the Bnet of Fig.35.1. One defines the TPMs for all nodes of Fig.35.2 except $\vec{\underline{Y}}$ and $\underline{V}$ the same as for Fig.35.1. For $\vec{\underline{Y}}$ and $\underline{V}$, one defines

$$P(Y[\sigma] \mid \vec{\hat{y}}) = P(\underline{Y} = Y[\sigma] \mid \hat{y}[\sigma]) \tag{35.23}$$

$$P(V|\vec{Y}, \vec{y}) = \delta(V, \frac{-1}{nsam(\vec{x})} \ln \mathcal{L}) \ , \tag{35.24}$$

where $\mathcal{L} = \prod_\sigma P(\underline{Y} = y[\sigma] \mid \hat{y}[\sigma])$=likelihood.

Chapter 36

Linear Deterministic Bnets with External Noise

In this chapter, we will consider bnets which were referred to, prior to the invention of bnets, as: Sewall Wright's **Path Analysis (PA)** and **linear Structural Equations Models (SEM)**. Judea Pearl in his books calls them **linear Structural Causal Models (SCM)**, because they are very convenient for doing causal analysis. We will refer to them as linear Deterministic with External Noise (LDEN) diagrams. This chapter is devoted to LDEN diagrams, except that we will say a few words about non-linear DEN diagrams at the end.

A **DEN diagram** is a special kind of bnet. To build a DEN diagram, start with a deterministic bnet G. The deterministic nodes of G are called the **endogenous (internal) variables**. Now make a bigger bnet $\overline{G}$ called a DEN diagram by adding to each node $\underline{a}$ of G a non-deterministic root node $\underline{u_a}$ pointing into $\underline{a}$ only. The nodes $\underline{u_a}$ are called the **exogenous (external) variables**. The exogenous variables make their children noisy. They are assumed to be unobserved and their TPMs are prior probability distributions. Since they are root nodes, they are mutually independent. When we draw a DEN diagram, we will never draw the exogenous nodes, leaving them implicit.

A **linear DEN diagram (LDEN)** is a DEN diagram whose deterministic nodes have a TPM that is a linear function of the states of the parent nodes.

36.1 Example of LDEN diagram

The TPMs, printed in blue, for the nodes of the LDEN diagram Fig.36.1, are as follows.

$$P(y|w, z, u_{\underline{y}}) = \mathbb{1}(y = \epsilon w + \delta z + u_{\underline{y}}) \tag{36.1}$$

Figure 36.1: Example of a LDEN diagram wherein $\underline{x}$ splits into two nodes $\underline{z}$ and $\underline{w}$, then merges into node $\underline{y}$. There is also an arrow $\underline{z} \to \underline{w}$. Exogenous nodes are not shown. The Greek letters represent real numbers.

$$P(w|x, z, u_{\underline{w}}) = \mathbb{1}(w = \beta x + \gamma z + u_{\underline{w}}) \tag{36.2}$$

$$P(z|x, u_{\underline{z}}) = \mathbb{1}(z = \alpha x + u_{\underline{z}}) \tag{36.3}$$

$$P(x|u_{\underline{x}}) = \mathbb{1}(x = u_{\underline{x}}) \tag{36.4}$$

Hence,

$$\begin{aligned} y &= \epsilon w + \delta z + u_{\underline{y}} && (36.5)\\ &= \epsilon(\beta x + \gamma z + u_{\underline{w}}) + \delta z + u_{\underline{y}} && (36.6)\\ &= (\epsilon\gamma + \delta)z + \epsilon\beta x + \epsilon u_{\underline{w}} + u_{\underline{y}} && (36.7)\\ &= (\epsilon\gamma + \delta)z + \epsilon\beta u_{\underline{x}} + \epsilon u_{\underline{w}} + u_{\underline{y}} \ . && (36.8) \end{aligned}$$

Therefore

$$\left(\frac{\partial y}{\partial z}\right)_{u_{\cdot} - u_{\underline{z}}} = \epsilon\gamma + \delta \ , \tag{36.9}$$

where the partial derivative holds fixed all exogenous variables except $u_{\underline{z}}$. Note that this partial derivative is a sum of terms, and that each of those terms represents a different directed path from $\underline{z}$ to $\underline{y}(\underline{z})$. This is a general property of LDEN diagrams.

36.2 Fully Connected LDEN diagrams

The bnets that will be considered in this section will all be fully connected. Fully connected bnets are defined in Chapter Definition of a Bayesian Network. This section

uses the notation $\langle \underline{x}, \underline{y} \rangle$ for the covariance of any two random variables $\underline{x}, \underline{y}$. This $\langle \underline{x}, \underline{y} \rangle$ notation is defined in Chapter Notational Conventions and Preliminaries.

Consider a LDEN diagram with deterministic nodes $\underline{x}. = (\underline{x}_k)_{k=0,1,\ldots nx-1}$ and corresponding exogenous nodes $\underline{u}. = (\underline{u}_k)_{k=0,1,\ldots nx-1}$. Assume $\langle \underline{u}_i, \underline{u}_j \rangle = 0$ if $i \neq j$. The strength of each connection $\underline{x}_i \to \underline{x}_j$ of the LDEN diagram is measured by a **structural coefficient** $\alpha_{j|i} \in \mathbb{R}$. Some of the $\alpha_{j|i}$ may be zero, in which case the corresponding arrow $i \to j$ would not be drawn.

36.2.1 Fully connected LDEN diagram with $nx = 2$

$$\begin{tikzcd} \underline{x}_0 \arrow[d, "\alpha 1|0"] \\ \underline{x}_1 \end{tikzcd}$$

Figure 36.2: Fully connected LDEN diagram with two $\underline{x}_j$ nodes (exogenous nodes $\underline{u}_j$ not shown).

Consider the LDEN diagram of Fig.36.2. This diagram represents the following **structural equations**:

$$\underline{x}_0 = \underline{u}_0 \tag{36.10a}$$

$$\underline{x}_1 = \alpha_{1|0}\underline{x}_0 + \underline{u}_1 \ . \tag{36.10b}$$

Eqs.36.10 constitute a system of 2 linear equations in 2 unknowns (the $\underline{x}$'s) so we can solve for the $\underline{x}$'s in terms of the α's and $\underline{u}$'s.

Note also that

$$\langle \underline{x}_1, \underline{x}_0 \rangle = \alpha_{1|0} \langle \underline{x}_0, \underline{x}_0 \rangle \ . \tag{36.11}$$

Thus, $\alpha_{1|0}$ can be estimated from the covariances $\langle \underline{x}_i, \underline{x}_j \rangle$.

36.2.2 Fully connected LDEN diagram with $nx = 3$

$$\begin{tikzcd} \underline{x}_0 \arrow[d, "\alpha_{1|0}"'] \arrow[dr, "\alpha_{2|0}"] & \\ \underline{x}_1 \arrow[r, "\alpha_{2|1}"] & \underline{x}_2 \end{tikzcd}$$

Figure 36.3: Fully connected LDEN diagram with three $\underline{x}_j$ nodes (exogenous nodes $\underline{u}_j$ not shown).

Consider the LDEN diagram of Fig.36.3. This diagram represents the following **structural equations**:

$$\underline{x}_0 = \underline{u}_0 \tag{36.12a}$$
$$\underline{x}_1 = \alpha_{1|0}\underline{x}_0 + \underline{u}_1 \tag{36.12b}$$
$$\underline{x}_2 = \alpha_{2|0}\underline{x}_0 + \alpha_{2|1}\underline{x}_1 + \underline{u}_2 \ . \tag{36.12c}$$

Eqs.36.12 constitute a system of 3 linear equations in 3 unknowns (the $\underline{x}$'s) so we can solve for the $\underline{x}$'s in terms of the α's and $\underline{u}$'s.

Note also that

$$\langle \underline{x}_1, \underline{x}_0 \rangle = \alpha_{1|0} \langle \underline{x}_0, \underline{x}_0 \rangle \tag{36.13a}$$
$$\langle \underline{x}_2, \underline{x}_0 \rangle = \alpha_{2|0} \langle \underline{x}_0, \underline{x}_0 \rangle + \alpha_{2|1} \langle \underline{x}_1, \underline{x}_0 \rangle \tag{36.13b}$$
$$\langle \underline{x}_2, \underline{x}_1 \rangle = \alpha_{2|0} \langle \underline{x}_0, \underline{x}_1 \rangle + \alpha_{2|1} \langle \underline{x}_1, \underline{x}_1 \rangle \tag{36.13c}$$

Eqs.36.13 constitute a system of 3 linear equations in 3 unknowns (the α's) so we can solve solve for the α's in terms of covariances $\langle \underline{x}_i, \underline{x}_j \rangle$. This gives an estimate for the α's.

36.2.3 Fully connected LDEN diagram with arbitrary nx

Let $\underline{x}. = (x_i)_{i=0,1,\ldots,nx-1}$ and $\underline{x}_{<i} = (x_k)_{k=0,1,\ldots,i-1}$. Consider a fully connected LDEN diagram with deterministic nodes labeled $\underline{x}_i$. The $\underline{x}_i$ labels are assumed to be in **topological order** (i.e., the parents of node $\underline{x}_i$ are $\underline{x}_{<i}$). Let the TPMs, printed in blue, for the nodes $\underline{x}.$ of the LDEN diagram, be

$$P(x_i|x_{<i}, u_i) = \mathbb{1}(x_i = \sum_{k<i} \alpha_{i|k} x_k + u_i) \ , \tag{36.14}$$

for some parameters $\alpha_{i|k} \in \mathbb{R}$. The exogenous nodes $\underline{u}.$ are assumed to be independent so

$$P(u.) = \prod_i P(u_i) \tag{36.15}$$

and

$$\langle \underline{u}_i, \underline{u}_j \rangle = 0 \text{ if } i \neq j \ . \tag{36.16}$$

Note that

$$P(x.) = \sum_{u.} P(u.) \prod_i P(x_i|x_{<i}, u_i) \tag{36.17}$$
$$= E_{\underline{u}.}[\prod_i P(x_i|x_{<i}, u_i)] \ . \tag{36.18}$$

In terms of random variables, this system is described by the following **structural equations**:

$$\underline{x}_i = \sum_{k<i} \alpha_{i|k} \underline{x}_k + \underline{u}_i \ . \tag{36.19}$$

The structural equations can be written in matrix form as follows. Define a strictly lower triangular matrix A with the connection strengths $\alpha_{i|k} \in \mathbb{R}$ as entries. For example, for $nx = 4$,

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \alpha_{1|0} & 0 & 0 & 0 \\ \alpha_{2|0} & \alpha_{2|1} & 0 & 0 \\ \alpha_{3|0} & \alpha_{3|1} & \alpha_{3|2} & 0 \end{bmatrix} \ . \tag{36.20}$$

If we now represent the multinodes $\underline{x}_\cdot$ and $\underline{u}_\cdot$ as column vectors $\underline{x}$ and $\underline{u}$, we get

$$\underline{x} = A\underline{x} + \underline{u} \ . \tag{36.21}$$

Note that

$$\underline{x} = (1 - A)^{-1} \underline{u} \ . \tag{36.22}$$

Therefore,

$$\underline{x}_i = f_i(\underline{u}_{\leq i}) \ . \tag{36.23}$$

Therefore, if $i > j$,

$$\left\langle \underline{u}_i, \underline{x}_j \right\rangle = \left\langle \underline{u}_i, f_j(\underline{u}_{\leq j}) \right\rangle = 0 \ . \tag{36.24}$$

Thus, if $i > j$,

$$\left\langle \underline{x}_i, \underline{x}_j \right\rangle = \sum_{k<i} \alpha_{i|k} \left\langle \underline{x}_k, \underline{x}_j \right\rangle + \left\langle \underline{u}_i, \underline{x}_j \right\rangle \tag{36.25}$$

$$= \sum_{k<i} \alpha_{i|k} \left\langle \underline{x}_k, \underline{x}_j \right\rangle \ . \tag{36.26}$$

In matrix notation, Eq.(36.26) becomes

$$\left\langle \underline{x}, \underline{x}^T \right\rangle_L = A[\left\langle \underline{x}, \underline{x}^T \right\rangle_L + \left\langle \underline{x}, \underline{x}^T \right\rangle_D] \tag{36.27}$$

where we are using $\left\langle \underline{x}, \underline{x}^T \right\rangle_{i,j} = \left\langle \underline{x}_i, \underline{x}_j \right\rangle$ and denoting the strictly lower triangular part and diagonal part of a matrix M by M_L and M_D. Thus,

$$A = \left\langle \underline{x}, \underline{x}^T \right\rangle_L [\left\langle \underline{x}, \underline{x}^T \right\rangle_L + \left\langle \underline{x}, \underline{x}^T \right\rangle_D]^{-1} \ . \tag{36.28}$$

This gives an estimate for the α's in terms of the covariances $\left\langle \underline{x}_i, \underline{x}_j \right\rangle$.

36.3 Non-linear DEN diagrams

This chapter is dedicated to linear DEN diagrams. This implicitly assumes that the deterministic nodes $\underline{x}$. of the DEN diagram have an interval of real values as their possible states. A trivial but very useful generalization of linear DEN diagrams is to replace Eq.(36.14) for the TPMs of the deterministic nodes of the diagram by

$$P(x_i|x_{<i}, u_i) = \mathbb{1}(x_i = f_i(x_{<i}, u_i)) \ , \tag{36.29}$$

with structural equations

$$\underline{x}_i = f_i(\underline{x}_{<i}, \underline{u}_i) \ , \tag{36.30}$$

for $i = 0, 1, \ldots, nx - 1$. Here the f_i are possibly non-linear functions that depend the states $x_{<i}$ and u_i of nodes $\underline{x}_{<i}$ and $\underline{u}_i$. If a node $\underline{x}_i$ has no arrows entering it (i.e., is a root node), then

$$P(x_i|x_{<i}, u_i) = P(x_i) = \delta(x_i, a) \tag{36.31}$$

and

$$\underline{x}_i = a \tag{36.32}$$

for some $a \in S_{\underline{x}_i}$.

With this generalization, we can make any $f_i()$ represent a continuous probability distribution such as a Gaussian, or a discrete-valued Boolean function such as an OR gate.

Eqs.(36.29) and (36.30) are the TPMs and structural equations for a fully connected, non-linear DEN diagram. For a non-fully connected diagram,

- replace the multinode $x_{<i}$ by a subset of itself, in Eqs.(36.29) and (36.30) , and
- delete the corresponding arrows from the graph.

Chapter 37

Markov Blankets

This chapter is based on the Wikipedia article, Ref.[103]. Markov blankets and Markov boundaries of bnets were apparently invented by Judea Pearl. His 1988 book Ref.[35], instead of a research paper, is usually given as the original reference.

Figure 37.1: In a bnet, the minimal Markov blanket, aka Markov boundary, of node $\underline{a}$.

We will treat vectors of random variables as if they were sets when using the $\in$, $\subset$ and $-$ operations. For example, if $\underline{x} = (\underline{x}_0, \underline{x}_1, \underline{x}_2, \underline{x}_3)$ and $\underline{b} = (\underline{x}_1, \underline{x}_2)$, then $\underline{x}_1 \in \underline{b} \subset \underline{x}$ and $\underline{x} - \underline{b} = (\underline{x}_0, \underline{x}_3)$.

Below, $H(\underline{a} : \underline{b}|\underline{c})$ denotes the conditional mutual information of random variables $\underline{a}$ and $\underline{b}$ conditioned on random variable $\underline{c}$. $H(\underline{a} : \underline{b}|\underline{c})$ is used in Shannon Information Theory, where it is defined by

$$H(\underline{a} : \underline{b}|\underline{c}) = \sum_{a,b,c} P(a,b,c) \ln \frac{P(a,b|c)}{P(a|c)P(b|c)} \ . \tag{37.1}$$

$H(\underline{a} : \underline{b}|\underline{c}) = 0$ iff $\underline{a}$ and $\underline{b}$ are independent (uncorrelated) when $\underline{c}$ is held fixed.

Suppose $\underline{a} \in \underline{X}$, $\underline{B} \subset \underline{X}$, but $\underline{a} \notin \underline{B}$. Then $\underline{B}$ is a **Markov blanket** of $\underline{a}$ if

$$H(\underline{a} : \underline{X} - \underline{a}|\underline{B}) = 0 \ . \tag{37.2}$$

In other words, one may assume that $\underline{a}$ depends on $\underline{B}$ only, and is independent of all random variables in $\underline{X} - (\underline{a} \cup \underline{B})$.

The **minimal Markov blanket** is called the **Markov boundary**.

In a bnet, the Markov boundary of a node $\underline{a}$, contains:

1. the parents of $\underline{a}$,
2. the children of $\underline{a}$,
3. the parents, other than $\underline{a}$, of the children of $\underline{a}$.

This is illustrated in Fig.37.1.

From the d-separation theorem (see Chapter 17), we get an intuitive motivation for the definition of Markov boundary. By conditioning on the parents and children of node $\underline{a}$, we block almost all, but not all, information from reaching node $\underline{a}$. The reason not all info is blocked is that conditioning on a child $\underline{c}$ of $\underline{a}$ creates paths from a parent of $\underline{c}$ to $\underline{c}$ to $\underline{a}$, wherein node $\underline{c}$ acts as a conditioned collider. To block such paths, we must condition on the parents of $\underline{c}$ too.

Chapter 38

Markov Chain Monte Carlo (MCMC)

Monte Carlo methods are methods for using random number generation to sample probability distributions. The subject of Monte Carlo methods has many branches, as you can see from its Wikipedia category list, Ref.[106]. MCMC (Markov Chain Monte Carlo) is just one of those branches, albeit a major one. Metropolis-Hastings (MH) sampling is a very important MCMC method. Gibbs sampling is a special case of MH sampling. This chapter covers both, MH and Gibbs sampling. It also covers a few other types of sampling.

Throughout this chapter, we use $P_{\underline{x}} : S_{\underline{x}} \to [0,1]$ to denote the target probability distribution that we wish to obtain samples from.

38.1 Inverse Cumulative Sampling

For more info about this topic and some original references, see Ref.[92].

This is one of the simplest methods for obtaining samples from a probability distribution $P_{\underline{x}}$, but it requires knowledge of the inverse cumulative distribution of $P_{\underline{x}}$, which is often not available.

The **cumulative distribution** function is defined by:

$$CUM_{\underline{x}}(x) = P(\underline{x} < x) = \int_{x'<x} dx' \; P_{\underline{x}}(x') \; . \tag{38.1}$$

Note that

$$P_{\underline{x}}(x) = \frac{d}{dx} CUM_{\underline{x}}(x) \; . \tag{38.2}$$

For $t = 0, 1, \ldots, T-1$, let

$\underline{u}^{(t)} \in [0,1]$= random variable, uniformly distributed over $[0,1]$.

$\vec{\underline{x}}^{(t)} = (\underline{x}^{(t)}[\sigma])_{\sigma=0,1,\ldots,nsam(t)-1}$ where $\underline{x}^{(t)}[\sigma] \in S_{\underline{x}}$ for all σ. Vector of samples collected up to time t.

$$\begin{tikzcd}
\underline{u}^{(0)} \arrow[d] & \underline{u}^{(1)} \arrow[d] & \underline{u}^{(2)} \arrow[d] \\
\underline{\vec{x}}^{(0)} \arrow[r] & \underline{\vec{x}}^{(1)} \arrow[r] & \underline{\vec{x}}^{(2)}
\end{tikzcd}$$

Figure 38.1: bnet for Inverse Cumulative Sampling

The TPMs, printed in blue, for the nodes of bnet Fig.38.1, are:

$$P(u^{(t)}) = 1 \tag{38.3}$$

$$P(\vec{x}^{(t)}|\vec{x}^{(t-1)}, u^{(t)}) = \delta(\quad \vec{x}^{(t)}, [\vec{x}^{(t-1)}, CUM_{\underline{x}}^{-1}(u^{(t)})] \quad) \tag{38.4}$$

Motivation

Figure 38.2: Motivation for Inverse Cumulative Sampling.

See Fig.38.2.

Note that if $\underline{u}$ is uniformly distributed over the interval $[0, 1]$ and $a \in [0, 1]$, then

$$P(\underline{u} < a) = a \ . \tag{38.5}$$

Thus

$$\begin{aligned} P(CUM_{\underline{x}}^{-1}(\underline{u}) < x) &= P(\underline{u} < CUM_{\underline{x}}(x)) && (38.6) \\ &= CUM_{\underline{x}}(x) \ . && (38.7) \end{aligned}$$

Therefore,

$$dP(CUM_{\underline{x}}^{-1}(\underline{u}) < x) = P_{\underline{x}}(x)dx \ . \tag{38.8}$$

38.2 Rejection Sampling

For more info about this topic and some original references, see Ref.[117].

This method samples from a "candidates" probability distribution $P_{\underline{c}} : S_{\underline{x}} \to [0,1]$, in cases where sampling directly from the target probability distribution $P_{\underline{x}} : S_{\underline{x}} \to [0,1]$ is not possible.

Figure 38.3: bnet for Rejection Sampling

For $t = 0, 1, \ldots, T-1$, let
$\underline{u}^{(t)} \in [0,1]$= random variable, uniformly distributed over $[0,1]$.
$\underline{a}^{(t)} \in \{0,1\}$= accept candidate? (no=0, yes=1)
$\underline{c}^{(t)} \in S_{\underline{x}}$= sample that is a candidate for being accepted
$\vec{\underline{x}}^{(t)} = (\underline{x}^{(t)}[\sigma])_{\sigma=0,1,\ldots,nsam(t)-1}$ where $\underline{x}^{(t)}[\sigma] \in S_{\underline{x}}$ for all σ. Vector of samples collected up to time t.

This algorithm requires a priori definition of a candidate probability distribution $P_{\underline{c}} : S_{\underline{x}} \to \mathbb{R}$ such that

$$P_{\underline{x}}(x) < \beta P_{\underline{c}}(x) \tag{38.9}$$

for all $x \in S_{\underline{x}}$, for some $\beta \in \mathbb{R}$.

The TPMs, printed in blue, for the nodes of bnet Fig.38.3, are:

$$P(u^{(t)} = u) = 1 \tag{38.10}$$

$$P(\underline{c}^{(t)} = c) = P_{\underline{c}}(c) \tag{38.11}$$

$$P(\underline{a}^{(t)} = a | \underline{c}^{(t)} = c, \underline{u}^{(t)} = u) = \begin{cases} \delta(a,0) & \text{if } u\beta P_{\underline{c}}(c) \geq P_{\underline{x}}(c) \\ \delta(a,1) & \text{if } u\beta P_{\underline{c}}(c) < P_{\underline{x}}(c) \end{cases} \tag{38.12}$$

$$P(\vec{x}^{(t)} | \vec{x}^{(t-1)}, \underline{a}^{(t)} = a, \underline{c}^{(t)} = c) = \begin{cases} \delta(\vec{x}^{(t)}, \vec{x}^{(t-1)}) & \text{if } a = 0 \\ \delta(\vec{x}^{(t)}, [\vec{x}^{(t-1)}, c]) & \text{if } a = 1 \end{cases} \tag{38.13}$$

This last equation is only defined for $t > 0$. For $t = 0$, the left hand side reduces to $P(\vec{x}^{(0)})$ which must be specified a priori.

Motivation

Figure 38.4: Motivation for Rejection Sampling.

See Fig.38.4.

38.3 Metropolis-Hastings Sampling

For more info about this topic and some original references, see Refs.[2] and [104].

An advantage of this method is that it can sample unnormalized probability distributions $(constant)P_{\underline{x}}$ because it only uses ratios of $P_{\underline{x}}$ at two different points. Another advantage of this method is that it scales much better than other sampling methods as the number of dimensions of the sampled variable $\underline{x}$ increases.

This method produces samples that take a finite amount of time to reach steady state. The samples are also theoretically correlated instead of being i.i.d. as one desires. To mitigate for the steady state problem, one discards an initial set of samples (the "burn-in" period). To mitigate for the correlation problem, one calculates the autocorrelation between the samples and keeps only samples separated by a time interval after which the samples cease to be autocorrelated to a good approximation.

For $t = 0, 1, \ldots, T-1$, let

$\underline{u}^{(t)} \in [0,1]$= random variable, uniformly distributed over $[0,1]$.

$\underline{a}^{(t)} \in \{0,1\}$= accept candidate? (no=0, yes=1)

$\underline{c}^{(t)} \in S_{\underline{x}}$= sample that is a candidate for being accepted

$\underline{m}^{(t)} \in S_{\underline{x}}$= memory of last accepted sample

$\underline{\vec{x}}^{(t)} = (\underline{x}^{(t)}[\sigma])_{\sigma=0,1,\ldots,nsam(t)-1}$ where $\underline{x}^{(t)}[\sigma] \in S_{\underline{x}}$ for all σ. Vector of samples collected up to time t.

A **proposal TPM** $P_{\underline{c}|\underline{x}} : S^2_{\underline{x}} \to [0,1]$ must be specified a priori for this algorithm.

The TPMs, printed in blue, for the nodes of bnet Fig.38.5, are:

$$P(u^{(t)} = u) = 1 \tag{38.14}$$

Figure 38.5: bnet for Metropolis-Hastings Sampling

$$P(\underline{c}^{(t)} = c|\underline{m}^{(t)} = m) = P_{\underline{c}|\underline{x}}(c|m) \tag{38.15}$$

$$P(\underline{a}^{(t)} = a|\underline{c}^{(t)} = c, \underline{u}^{(t)} = u, \underline{m}^{(t)} = m) = \begin{cases} \delta(a,0) & \text{if } u \geq \alpha(c|m) \\ \delta(a,1) & \text{if } u < \alpha(c|m) \end{cases} \tag{38.16}$$

where the **acceptance probability** α is defined as

$$\alpha(c|m) = \min\left(1, \frac{P_{\underline{c}|\underline{x}}(m|c)P_{\underline{x}}(c)}{P_{\underline{c}|\underline{x}}(c|m)P_{\underline{x}}(m)}\right) . \tag{38.17}$$

Note that if the proposal distribution is symmetric, then

$$\alpha(c|m) = \min\left(1, \frac{P_{\underline{x}}(c)}{P_{\underline{x}}(m)}\right) . \tag{38.18}$$

$$P(\vec{x}^{(t)}|\vec{x}^{(t-1)}, \underline{a}^{(t)} = a, \underline{c}^{(t)} = c) = \begin{cases} \delta(\vec{x}^{(t)}, \vec{x}^{(t-1)}) & \text{if } a = 0 \\ \delta(\vec{x}^{(t)}, [\vec{x}^{(t-1)}, c]) & \text{if } a = 1 \end{cases} \tag{38.19}$$

This last equation is only defined for $t > 0$. For $t = 0$, the left hand side reduces to $P(\vec{x}^{(0)})$ which must be specified a priori.

$$P(\underline{m}^{(t)} = m|\vec{x}^{(t)}) = \delta(m, \text{last component of } \vec{x}^{(t)}) . \tag{38.20}$$

This last equation is only defined for $t > 0$. For $t = 0$, the left hand side reduces to $P(\underline{m}^{(0)} = m)$ which must be specified a priori.

Motivation

See Fig.38.6.

Figure 38.6: Motivation for Metropolis-Hastings Sampling.

Consider a time homogeneous (its TPM is the same for all times) Markov chain with TPM $P(x'|x) = [T]_{x',x}$. Its **stationary distribution**, if it exists, is defined as

$$\pi = \lim_{n\to\infty} T^n \pi_0 \ . \tag{38.21}$$

Suppose the prob distribution $P_{\underline{x}}(x)$ that we wish to sample from satisfies

$$P_{\underline{x}}(x) = \pi(x) \ . \tag{38.22}$$

Reversibility (detailed balance): For all $x, x' \in S_{\underline{x}}$,

$$P(x'|x)\pi(x) = P(x|x')\pi(x') \ . \tag{38.23}$$

Detailed balance is a sufficient (although not necessary) condition for a unique stationary prob distribution π to exist.[1]

Let

$$P(x'|x) = P(\underline{a} = 1|x', x)P_{\underline{c}|\underline{x}}(x'|x) + \delta(x, x')P(\underline{a} = 0|x) \ , \tag{38.24}$$

where

$$P(\underline{a} = 0|x) = \sum_{x'} P(\underline{a} = 0|x', x)P_{\underline{c}|\underline{x}}(x'|x) \ . \tag{38.25}$$

Claim 47 *If*

$$P(\underline{a} = 1|x', x) = \alpha(x'|x) \ , \tag{38.26}$$

[1] As explained lucidly in Ref.[2], besides detailed balance, 2 other properties must also be satisfied by the Markov chain, irreducibility and aperiodicity. However, because of how it is constructed, the Metropolis-Hastings algorithm automatically produces a Markov chain that has those 2 properties.

then detailed balance is satisfied.

proof: Assume $x \neq x'$.

$$
\begin{aligned}
P(x'|x)P(x) &= P(\underline{a}=1|x',x)P_{\underline{c}|\underline{x}}(x'|x)P_{\underline{x}}(x) && (38.27)\\
&= \min\left(1, \frac{P_{\underline{c}|\underline{x}}(x|x')P_{\underline{x}}(x')}{P_{\underline{c}|\underline{x}}(x'|x)P_{\underline{x}}(x)}\right) P_{\underline{c}|\underline{x}}(x'|x)P_{\underline{x}}(x) && (38.28)\\
&= \min\left(P_{\underline{c}|\underline{x}}(x'|x)P_{\underline{x}}(x), P_{\underline{c}|\underline{x}}(x|x')P_{\underline{x}}(x')\right) && (38.29)\\
&= P(x|x')P(x') && (38.30)
\end{aligned}
$$

QED

38.4 Gibbs Sampling

For more info about this topic and some original references, see Ref.[86].

Gibbs sampling is a special case of Metropolis-Hastings sampling. Gibbs sampling is ideally suited for application to a bnet, because it is stated in terms of the conditional prob distributions of N random variables, and conditional prob distributions are part of the definition of a bnet.

Consider a bnet with nodes $\underline{x}_0, \underline{x}_1, \ldots, \underline{x}_{N-1}$

Identify the random variable $\underline{x} = (\underline{x}_0, \underline{x}_1, \ldots, \underline{x}_{N-1})$ with the random variable $\underline{x}$ used in Metropolis-Hastings sampling. For Gibbs sampling, we use the following proposal distribution:

$$
P_{\underline{c}|\underline{x}}(c|m) = \prod_{j=0}^{N-1} P(c_j \mid [m_i]_{i\neq j}) \ . \tag{38.31}
$$

Eq.(38.31) can be simplified using Markov Blankets (see Chapter 37) to the following:

$$
P_{\underline{c}|\underline{x}}(c|m) = \prod_{j=0}^{N-1} P(c_j \mid [m_i : \forall i \ni \underline{x}_i \in MB(\underline{x}_j)]) \ , \tag{38.32}
$$

where, for any node $\underline{a}$, we denote its Markov blanket by $MB(\underline{a})$.

An alternative proposal distribution that leads to much faster convergence is as follows. The idea is to make the components $c_j^{(t)}$ of candidate sample $c^{(t)}$ depend on the previous components $(c_i^{(t)})_{i<j}$. See the bnet Fig.38.7. The TPM for the nodes of that bnet are

$$
P(\underline{c}_j^{(t)} = c_j \mid (\underline{c}_i^{(t)})_{i<j} = (c_i)_{i<j}, \underline{m}^{(t-1)} = m) = P(c_j|(c_i)_{i<j}, (m_i)_{i>j}) \tag{38.33}
$$

Figure 38.7: In Gibbs sampling, the proposal distribution $P_{\underline{c}|\underline{x}}$ can be defined by making the components $c_j^{(t)}$ of candidate sample $c^{(t)}$ depend on the previous components $(c_i^{(t)})_{i<j}$.

for $j = 0, 1, \ldots, N-1$. This implies

$$P_{\underline{c}|\underline{x}}(\underline{c}^{(t)} = c|\underline{m}^{(t-1)} = m) = \prod_{j=0}^{N-1} P(c_j|(c_i)_{i<j}, (m_i)_{i>j}) \ . \tag{38.34}$$

As before, we can condition only on the Markov blanket of each node $\underline{x}_j$.

$$P_{\underline{c}|\underline{x}}(\underline{c}^{(t)} = c|\underline{m}^{(t-1)} = m) = \prod_{j=0}^{N-1} P(c_j|(c_i)_{i<j}, (m_i)_{i>j}, \text{ use only } c_i \text{ and } m_i \ni \underline{x}_i \in MB(\underline{x}_j)) \ . \tag{38.35}$$

38.5 Importance Sampling

For more info about this topic and some original references, see Ref.[90].

Suppose random variables $\underline{x}[\sigma] \in S_{\underline{x}}$ for $\sigma = 0, 1, \ldots, nsam-1$ are i.i.d. with probability distribution $P_{\underline{x}}$. Then

$$E_{\underline{x}}[f(x)] \approx \frac{1}{nsam} \sum_{\sigma=0}^{nsam-1} f(x[\sigma]) \tag{38.36}$$

for any $f : S_{\underline{x}} \to \mathbb{R}$. Sometimes, instead of using i.i.d. samples $\underline{x}[\sigma] \in S_{\underline{x}}$ where $\underline{x}[\sigma] \sim P_{\underline{x}}$, we wish to use i.i.d. samples $\underline{y}[\sigma] \in S_{\underline{x}}$ where $\underline{y}[\sigma] \sim P_{\underline{y}}$.

$$E_{\underline{x}}[f(\underline{x})] = \sum_x P_{\underline{x}}(x) f(x) \tag{38.37}$$

$$= \sum_x P_{\underline{y}}(x)) \frac{P_{\underline{x}}(x)}{P_{\underline{y}}(x)} f(x) \tag{38.38}$$

$$= E_{\underline{y}}[\frac{P_{\underline{x}}(y)}{P_{\underline{y}}(y)} f(y)] \tag{38.39}$$

Sampling from $P_{\underline{y}}(y)$ instead of $P_{\underline{x}}(x)$ might reduce (or increase) variance for a particular $f : S_{\underline{x}} \to \mathbb{R}$.

$$Var_{\underline{x}}[f(x)] = E_{\underline{x}}[(f(x))^2] - (E_{\underline{x}}[f(x)])^2 \tag{38.40}$$

$$Var_{\underline{y}}[\frac{P_{\underline{x}}(y)}{P_{\underline{y}}(y)} f(y)] = E_{\underline{y}}[(\frac{P_{\underline{x}}(y)}{P_{\underline{y}}(y)} f(y))^2] - (E_{\underline{y}}[\frac{P_{\underline{x}}(y)}{P_{\underline{y}}(y)} f(y)])^2 \tag{38.41}$$

$$= E_{\underline{x}}[\frac{P_{\underline{x}}(x)}{P_{\underline{y}}(x)} (f(x))^2] - (E_{\underline{x}}[f(x)])^2 \tag{38.42}$$

Chapter 39

Markov Chains

A Markov Chain is simply a bnet with the graph structure of a chain. For example, Fig.39.1 shows a chain with $n = 4$ nodes.

$$\underline{x}_0 \longrightarrow \underline{x}_1 \longrightarrow \underline{x}_2 \longrightarrow \underline{x}_3$$

Figure 39.1: Markov chain with $n = 4$ nodes.

Because of its graph structure, the TPM of each node only depends on the state of the previous node:

$$P(x_t|(x_a)_{a\neq t}) = P(x_t|x_{t-1}) \ , \tag{39.1}$$

where $(x_a)_{a\neq t}$ are all the nodes except x_t itself and $t = 1, 2, \ldots, n-1$.

If there exists a single TPM $P_{\underline{x}_1|\underline{x}_0}$ such that

$$P(x_t|x_{t-1}) = P_{\underline{x}_1|\underline{x}_0}(x_t|x_{t-1}) \tag{39.2}$$

for $t = 1, 2, \ldots, n-1$, then we say that the Markov chain is **time homogeneous**.

Claim 48 *(Data Processing Inequality (DPI))*

Consider a Markov chain $\underline{x}_0 \to \underline{x}_1 \cdots \to \underline{x}_{n-1}$. *Suppose* $0 \leq a < m < b \leq n-1$. *Then*

$$H(\underline{x}_a : \underline{x}_b) \leq \min[H(\underline{x}_a : \underline{x}_m), H(\underline{x}_m : \underline{x}_b)] \tag{39.3}$$

See Ref.[78] for references where the DPI is proven. This inequality confirms our intuitive expectations that the information transmitted (i.e., the mutual information(MI)) from $\underline{a}$ to $\underline{b}$ (or vice versa since MI is symmetric) is smaller or equal to the one transmitted from $\underline{a}$ to $\underline{m}$ or from $\underline{m}$ to $\underline{b}$ because $\underline{a}$ and $\underline{b}$ are "farther apart" and "some info can get lost during transmission through the mediator node $\underline{m}$".

Chapter 40

Mediation Analysis

This chapter is mostly based on Refs.[38, 36] by Pearl.

To fully understand this chapter, the reader should first read Chapter 10 where do and imagine operators are defined.

Figure 40.1: DEN bnets G_0 and G are used to discuss mediation analysis. In graph G_0, the external variables are independent, whereas in graph G they are not.

The term "mediation analysis" (MA) refers to the analysis by Pearl of the DEN bnet G in Fig.40.1. We will discuss MA in terms of DEN bnets, just as Pearl does. However, note that much of what we will say applies also to the general (i.e., not necessarily a DEN) bnet G_{gen} show in Fig. 40.2.

In the DEN bnet G, node $\underline{d}$ influences node $\underline{y}$ both directly and via the mediator node $\underline{m}$. The structural equations for G are of the form:

$$\underline{d} = \underline{u}_{\underline{d}} \tag{40.1a}$$

$$\underline{m} = f_{\underline{m}}(\underline{d}, \underline{u}_{\underline{m}}) \tag{40.1b}$$

$$\underline{y} = f_{\underline{y}}(\underline{d}, \underline{m}, \underline{u}_{\underline{y}}) \ . \tag{40.1c}$$

G_{gen}

Figure 40.2: General bnet used to discuss mediation analysis. Node $\underline{u}$ is a hidden confounder.

Thus,

$$\underline{y} = f_{\underline{y}}(u_{\underline{d}}, f_{\underline{m}}(\underline{u}_{\underline{d}}, \underline{u}_{\underline{m}}), \underline{u}_{\underline{y}}) \ . \tag{40.2}$$

$\mathcal{D}_{\underline{d}=5}G$ $\qquad$ $\mathcal{I}_{\underline{d}\to\underline{m}}(5)G$

Figure 40.3: Graph G of Fig.40.1 after applying do operator $\mathcal{D}_{\underline{d}=5}$ and imagine operator $\mathcal{I}_{\underline{d}\to\underline{m}}(5)$.

If we apply $\mathcal{D}_{\underline{d}=5}G$ to Eqs.(40.1), we get

$$\begin{aligned} \underline{d} &= 5 && (40.3a) \\ \underline{m} &= f_{\underline{m}}(\underline{d}, \underline{u}_{\underline{m}}) && (40.3b) \\ \underline{y} &= f_{\underline{y}}(\underline{d}, \underline{m}, \underline{u}_{\underline{y}}) \ . && (40.3c) \end{aligned}$$

Eqs.40.3 are represented graphically in Fig.40.3. We will often denote the random variable $\underline{y}$ in Eqs.(40.3) by the more explicit symbol $\underline{y}_{\mathcal{D}_{\underline{d}=5}G}$. Pearl often refers to this $\underline{y}$ by the less explicit symbol Y_5 or $Y_5(u)$ where $u = (u_{\underline{m}}, u_{\underline{y}}) = u_{!\underline{d}}$.

If we apply $\mathcal{I}_{\underline{d}\to\underline{m}}(5)G$ to Eqs.(40.1), we get

$$\underline{d} = \underline{u}_{\underline{d}} \tag{40.4a}$$
$$\underline{m} = f_{\underline{m}}(5, \underline{u}_{\underline{m}}) \tag{40.4b}$$
$$\underline{y} = f_{\underline{y}}(\underline{d}, \underline{m}, \underline{u}_{\underline{y}}) \ . \tag{40.4c}$$

Eqs.40.4 are represented graphically in Fig.40.3. We will often denote the random variable $\underline{y}$ in Eqs.(40.4) by the more explicit symbol $\underline{y}_{\mathcal{I}_{\underline{d}\to\underline{m}}(5)G}$. Pearl often refers to this $\underline{y}$ by the less explicit symbol Y_5 or $Y_5(u)$ where $u = (u_{\underline{d}}, u_{\underline{m}}, u_{\underline{y}})$.

Figure 40.4: Graph G of Fig.40.1 after applying the do operators $\mathcal{D}_{\underline{d}=d}$ and $\mathcal{D}_{\underline{d}=d}\mathcal{D}_{\underline{m}=m}$.

Define

$$\mathcal{Y}_d = E[\underline{y}_{\mathcal{D}_{\underline{d}=d}G}] = \sum_y yP(y|\mathcal{D}\underline{d} = d) \tag{40.5}$$

and

$$\mathcal{Y}_d^m = E[\underline{y}_{\mathcal{D}_{\underline{d}=d}\mathcal{D}_{\underline{m}=m}G}] = \sum_y yP(y|\mathcal{D}\underline{d} = d, \mathcal{D}\underline{m} = m) \tag{40.6}$$

The two DEN diagrams $\mathcal{D}_{\underline{d}=d}G$ and $\mathcal{D}_{\underline{d}=d}\mathcal{D}_{\underline{m}=m}G$ used in the definitions of $\mathcal{Y}_d$ and $\mathcal{Y}_d^m$ are given in Fig.40.4.

Now define the Total Effect (TE), and the Controlled Direct Effect (CDE) by

$$TE = \mathcal{Y}_1 - \mathcal{Y}_0 \tag{40.7}$$
$$CDE(m) = \mathcal{Y}_1^m - \mathcal{Y}_0^m \tag{40.8}$$

Define

$$\overline{\mathcal{Y}}_d^{d'} = E[\underline{y}_{\mathcal{D}_{\underline{d}=d}\mathcal{I}_{\underline{d}\to\underline{m}}(d')G}] = \sum_y yP(y|\mathcal{D}\underline{d} = d, \mathcal{I}_{\underline{m}}\underline{d} = d') \tag{40.9}$$

$$\mathcal{D}_{\underline{d}=d}\mathcal{I}_{\underline{d}\to\underline{m}}(d')G$$

Figure 40.5: Graph G of Fig.40.1 after applying the imagine operator $\mathcal{I}$ to arrow $\underline{d} \to \underline{m}$ and the do operator to node $\underline{d}$.

Fig.40.5 shows the diagram $\mathcal{D}_{\underline{d}=d}\mathcal{I}_{\underline{d}\to\underline{y}}(d')G$ used in the definition of $\overline{\mathcal{Y}}_d^{d'}$.

Note that

$$\overline{\mathcal{Y}}_d^{d'} = \sum_m \mathcal{Y}_d^m P(\underline{m} = m|d') \tag{40.10}$$

if there is no arrow $\underline{u}_{\underline{m}} \to \underline{m}$. This expresses $P(y|\mathcal{D}\underline{d} = d, \mathcal{I}\underline{d} = d')$ in terms of $P(y|\mathcal{D}\underline{d} = d, \mathcal{D}\underline{m} = m)$.

Define

$$\overline{\mathcal{Y}}_d^{d'-} = \overline{\mathcal{Y}}_d^{d'} - \overline{\mathcal{Y}}_{d'}^{d'} \tag{40.11}$$

and

$$\overline{\mathcal{Y}}_{d-}^{d'} = \overline{\mathcal{Y}}_d^{d'} - \overline{\mathcal{Y}}_d^{d} \tag{40.12}$$

Now define the Natural Direct Effect (NDE), and the Natural Indirect Effect (NIE) by

$$NDE_d^{d'} = \overline{\mathcal{Y}}_d^{d'-} \tag{40.13}$$

$$NIE_d^{d'} = \overline{\mathcal{Y}}_{d-}^{d'} . \tag{40.14}$$

Note that

$$NDE_1^0 - NIE_1^0 = -\overline{\mathcal{Y}}_0^0 + \overline{\mathcal{Y}}_1^1 \tag{40.15}$$

$$= TE . \tag{40.16}$$

Linear Case

Consider the LDEN of Fig.40.6. One has

$$\underline{d} = \underline{u}_{\underline{d}} \tag{40.17a}$$

$$\underline{m} = \alpha\underline{d} + \underline{u}_{\underline{m}} \tag{40.17b}$$

$$\underline{y} = \gamma\underline{d} + \beta\underline{m} + \underline{u}_y . \tag{40.17c}$$

Figure 40.6: LDEN that is used to discuss mediation analysis.

$$\mathcal{Y}_d = (\gamma + \alpha\beta)d \tag{40.18}$$

$$\mathcal{Y}_d^m = \gamma d + \beta m \tag{40.19}$$

$$\overline{\mathcal{Y}}_d^{d'} = \gamma d + \alpha\beta d' \tag{40.20}$$

$$TE = \mathcal{Y}_1 - \mathcal{Y}_0 = \gamma + \alpha\beta \tag{40.21}$$

$$CDE(m) = \mathcal{Y}_1^m - \mathcal{Y}_0^m = \gamma \tag{40.22}$$

$$NDE_1^0 = \overline{\mathcal{Y}}_1^{0-} = \gamma \tag{40.23}$$

$$NIE_1^0 = \overline{\mathcal{Y}}_{1-}^0 = -\alpha\beta \tag{40.24}$$

As expected, $NDE_1^0 - NIE_1^0 = TE$.

TE and the "controlled effect" $CDE(m)$ contain do operators but no imagine operators so one can do do-intervention experiments to calculate them. On the other hand, the "natural effects" NDE_1^0 and NIE_1^0 contain imagine operators (and therefore counterfactual distributions) so it's not obvious how to calculate them, or even whether it is possible to do so. The next claim shows how to calculate $\overline{\mathcal{Y}}_d^{d'}$ in certain cases. In technical jargon, the claim gives sufficient conditions for $\mathcal{DI}$-identifiability of $\overline{\mathcal{Y}}_d^{d'}$. NDE_1^0 and NIE_1^0 can be calculated once $\overline{\mathcal{Y}}_d^{d'}$ is known.

Claim 49 *(Unconfounded Mediation, from Ref.[38])*
If *then*

$$P(y|\mathcal{D}\underline{d} = d, \mathcal{I}_{\underline{m}}\underline{d} = d') = \sum_m P(y|d,m)P(m|d') \tag{40.25}$$

$$\begin{tikzcd}[column sep=small] \mathcal{I}\underline{d} = d' \arrow[dr] & & \\ \mathcal{D}\underline{d} = d \arrow[r] & y \end{tikzcd} \quad = \quad \begin{tikzcd} \mathcal{I}\underline{d} = d' \arrow[r] & \sum m \arrow[dr] & \\ \mathcal{D}\underline{d} = d \arrow[rr] & & y \end{tikzcd} \tag{40.26}$$

proof: See Claim 31.
QED

Claim 50 *(Mediation with universal prior $\underline{\xi}$ and universal confounder $\underline{u}$, from Ref.[38])*

If *then*

$$\begin{tikzcd} & \overset{\frown}{\underline{u}} \arrow[dl] \arrow[d] \arrow[ddr] & \\ \underline{\xi} \arrow[r] \arrow[d] \arrow[drr] & \underline{m} \arrow[dr] & \\ \underline{d} \arrow[r] \arrow[urr] & & \underline{y} \end{tikzcd}$$

$$P(y|\mathcal{D}\underline{d} = d, \mathcal{I}_{\underline{m}}\underline{d} = d') = \sum_{\xi} \sum_{m} P(y|d, m, \xi) P(m|d', \xi) P(\xi) \tag{40.27}$$

$$\begin{tikzcd}[column sep=small] \mathcal{I}\underline{d} = d' \arrow[dr] & & \\ \mathcal{D}\underline{d} = d \arrow[r] & y \end{tikzcd} \quad = \quad \begin{tikzcd} \mathcal{I}\underline{d} = d' \arrow[rr, bend right] & \sum \xi \arrow[r] \arrow[drr] & \sum m \arrow[dr] & \\ \mathcal{D}\underline{d} = d \arrow[rrr] & & & y \end{tikzcd} \tag{40.28}$$

proof: See Claim 32.
QED

Actually,

$$\sum_{m} P(y|d, m) P(m|d') = \sum_{\xi} \sum_{m} P(y|d, m, \xi) P(m|d', \xi) P(\xi) \tag{40.29}$$

so the adjustment formulae in Claims 49 and 50 are equivalent, but if the available dataset contains info about ξ, then the adjustment formula that uses that ξ info should be used, as it will be more sensitive to deviations from the DAG model being hypothesized.

Chapter 41

Message Passing (Belief Propagation)

Belief Propagation was first proposed in 1982 Ref.[34] by Judea Pearl to simplify the exact evaluation of the probability of one node conditioned on other nodes of a bnet (exact inference). It gives exact results for trees and polytrees (i.e. bnets with a single connected component and no loops). For bnets with loops, it gives approximate results (loopy belief propagation), and it has been generalized to the junction tree algorithm (see Chapter 33) which can do exact inference for general bnets with loops. The basic idea behind the junction tree algorithm is to eliminate loops by clustering them into single nodes.

In his book Ref.[35], Pearl explains two types of Message Passing (i.e., distributed computing in a bnet). In Chapter 4, he discusses one type of MP which he calls Belief Propagation (BP) or Belief Updating. In Chapter 5, he introduces a second type of MP which is he calls Belief Revision, but which I prefer to call Explanation Optimization (EO). This chapter will be devoted to BP only.

This chapter is mostly based on chapter 4 of Ref.[35] by Pearl. Refs.[64], and [28] were also helpful in writing this chapter.

41.1 Distributed Soldier Counting

Consider a group of soldiers marching single file. Fig.41.1 shows several methods by which a member of the group can obtain a count of the soldiers without breaking the line to do global operations. This can be done in a distributed fashion, with every soldier doing only local operations (i.e., each soldier can only send messages to either the soldier in front or the one in back). Such distributed soldier counting is a rudimentary type of BP. In the next section, we will generalize this BP for soldiers to BP for a Markov chain.

Figure 41.1: Distributed soldier counting (This example comes from Chapter 4 of Ref.[35]). Green dots indicate the beginning and red dots the end of a count. Only first soldier can calculate total count in (a). Only third soldier can calculate total count in (b,c). All soldiers can calculate the total count in (d,e). One starting point in (a,b,e). Two ends as starting points in (c,d).

41.2 Spring Systems

Figure 41.2: Spring system. Point masses connected by springs.

See Ref.[121] for an introduction to spring systems. Ideal springs between the point mass nodes would not be sufficient. One would have to add damping to the springs so as to reach an equilibrium. Time dependent forces (loads) pointing into or out of the page, applied to the point masses, would generate signals that would propagate like BP messages.

41.3 BP for Markov Chains

Figure 41.3: 3 node Markov chain $\underline{\epsilon}^+ \leftarrow \underline{x} \leftarrow \underline{\epsilon}^-$. The π messages (probability functions) travel downstream (i.e., they carry info in the direction of the graph arrows, towards the future) and are indicated by a dashed arrow or by a left double arrow $\Leftarrow$. The λ messages (likelihood functions) travel upstream (i.e., they carry info opposite to direction of the graph arrows, towards the past) and are indicated by a dotted arrow or by a right double arrow $\Rightarrow$. $\underline{\epsilon}^+$ stands for future evidence and $\underline{\epsilon}^-$ for past evidence.

Consider the 3 node Markov chain $\underline{\epsilon}^+ \leftarrow \underline{x} \leftarrow \underline{\epsilon}^-$ shown in Fig.41.3. Define[1]

$$\pi_{\underline{\epsilon}^+ \Leftarrow \underline{x}}(x) = P(x|\epsilon^-) \text{ (past of } \underline{x}) \tag{41.1}$$

$$\lambda_{\underline{\epsilon}^+ \Rightarrow \underline{x}}(x) = P(\epsilon^+|x) \text{ (future of } \underline{x}) \tag{41.2}$$

$$\pi_{\underline{x} \Leftarrow \underline{\epsilon}^-}(\epsilon^-) = P(\epsilon^-) = \delta(\epsilon^-, \epsilon_0^-) \text{ (past of } \underline{\epsilon}^-) \tag{41.3}$$

$$\lambda_{\underline{x} \Rightarrow \underline{\epsilon}^-}(\epsilon^-) = P(\epsilon^+|\epsilon^-) \text{ (future of } \underline{\epsilon}^-) \ . \tag{41.4}$$

Furthermore, define the Belief BEL in x to be

$$BEL_{\underline{x}}(x) = P(x|\epsilon) \ , \tag{41.5}$$

where

$$\underline{\epsilon} = \underline{\epsilon}^+ \cup \underline{\epsilon}^- \ . \tag{41.6}$$

It follows that

$$\begin{aligned} BEL_{\underline{x}}(x) &= P(x|\epsilon^+, \epsilon^-) = && (41.7)\\ &= \mathcal{N}(!x) P(\epsilon^+, x, \epsilon^-) && (41.8)\\ &= \mathcal{N}(!x) P(\epsilon^+|x) P(x|\epsilon^-) && (41.9)\\ &= \mathcal{N}(!x) \lambda_{\underline{\epsilon}^+ \Rightarrow \underline{x}}(x) \pi_{\underline{\epsilon}^+ \Leftarrow \underline{x}}(x) \ . && (41.10) \end{aligned}$$

Note that Bayes rule would affirm that[2]

$$P(x|\epsilon^+) = \mathcal{N}(!x) \underbrace{P(\epsilon^+|x)}_{\lambda_{\underline{\epsilon}^+ \Rightarrow \underline{x}}(x)} P(x) \ . \tag{41.11}$$

Thus, Eq.(41.10) is like a 2-sided Janus Bayes rule.

Note that the π messages and λ messages propagate independently of each other, via the TPM $P(x|\epsilon^-)$:

$$\underbrace{\pi_{\underline{\epsilon}^+ \Leftarrow \underline{x}}(x)}_{P(x|\epsilon_0^-)} = \sum_{\epsilon^-} P(x|\epsilon^-) \underbrace{\pi_{\underline{x} \Leftarrow \underline{\epsilon}^-}(\epsilon^-)}_{\delta(\epsilon^-, \epsilon_0^-)} \tag{41.12a}$$

[1] The pattern behind these definitions, in case it eludes you, is as follows: the π's always carry information about the past and the λ's about the future. But the past or future of what? Of the argument of the function. Out of the two random variables in the subscript of the function, the one on the right hand side of the subscript, the one which is adjacent but beneath the argument, is always the argument.

[2] As usual in this book, $\mathcal{N}(!x)$ means a constant that is independent of x.

$$\underbrace{\lambda_{\underline{x}\Rightarrow\underline{\epsilon}^-}(\epsilon^-)}_{P(\epsilon^+|\epsilon^-)} = \sum_x P(x|\epsilon^-)\,\underbrace{\lambda_{\underline{\epsilon}^+\Rightarrow\underline{x}}(x)}_{P(\epsilon^+|x)} \tag{41.12b}$$

Eqs.(41.12) suggest that we define an edge bnet for the π and λ messages (these messages live in the edges between the nodes $\underline{\epsilon}^+, \underline{x}, \underline{\epsilon}^-$). Such an edge bnet, shown in Fig.41.4, is complementary to bnet for the nodes themselves. We will call it the **BP 2-track bnet** for the bnet Fig.41.3, because it has two "tracks", one for π messages and another for λ ones. The TPMs, shown in blue, for the nodes of bnet Fig.41.4, are as follows:

$$\begin{tikzcd}
\underline{\pi}_{\underline{\epsilon}^+\Leftarrow\underline{x}} \arrow[dr] & & \underline{\pi}_{\underline{x}\Leftarrow\underline{\epsilon}^-} \arrow[ll] \\
& \underline{B}_{\underline{x}} & \\
\underline{\lambda}_{\underline{\epsilon}^+\Rightarrow\underline{x}} \arrow[ur] \arrow[rr] & & \underline{\lambda}_{\underline{x}\Rightarrow\underline{\epsilon}^-}
\end{tikzcd}$$

Figure 41.4: BP 2-track bnet for the bnet Fig.41.3.

$$P(\pi_{\underline{x}\Leftarrow\underline{\epsilon}^-}) = \prod_{\epsilon^-} \mathbb{1}(\pi_{\underline{x}\Leftarrow\underline{\epsilon}^-}(\epsilon^-) = P(\epsilon^-)) \tag{41.13}$$

$$P(\pi_{\underline{\epsilon}^+\Leftarrow\underline{x}}|\pi_{\underline{x}\Leftarrow\underline{\epsilon}^-}) = \prod_x \mathbb{1}\left(\pi_{\underline{\epsilon}^+\Leftarrow\underline{x}}(x) = \sum_{\epsilon^-} P(x|\epsilon^-)\pi_{\underline{x}\Leftarrow\underline{\epsilon}^-}(\epsilon^-)\right) \tag{41.14}$$

$$P(B_{\underline{x}}|\pi_{\underline{\epsilon}^+\Leftarrow\underline{x}}, \lambda_{\underline{\epsilon}^+\Rightarrow\underline{x}}) = \prod_x \mathbb{1}\left(B_{\underline{x}}(x) = BEL_{\underline{x}}(x)\right) \tag{41.15}$$

$$P(\lambda_{\underline{\epsilon}^+\Rightarrow\underline{x}}) = \prod_x \mathbb{1}\left(\lambda_{\underline{\epsilon}^+\Rightarrow\underline{x}}(x) = P(\epsilon^+|x)\right) \tag{41.16}$$

$$P(\lambda_{\underline{x}\Rightarrow\underline{\epsilon}^-}|\lambda_{\underline{\epsilon}^+\Rightarrow\underline{x}}) = \prod_{\epsilon^-} \mathbb{1}\left(\lambda_{\underline{x}\Rightarrow\underline{\epsilon}^-}(\epsilon^-) = \sum_x P(x|\epsilon^-)\lambda_{\underline{\epsilon}^+\Rightarrow\underline{x}}(x)\right) \tag{41.17}$$

Figure 41.5: 5 node Markov chain

So far in this section, we have considered Markov chains with 3 nodes. Before concluding our discussion of BP for Markov chains, let us consider BP for a slightly longer chain. Let us consider the 5 node Markov chain $\underline{\epsilon}^+ \leftarrow \underline{b} \leftarrow \underline{x} \leftarrow \underline{a} \leftarrow \underline{\epsilon}^-$ shown in Fig.41.5. We have already dealt with the end nodes of a Markov chain in the 3 node Markov chain example above, so in the 5 node case, let us focus on the internal (i.e., not at an end) node $\underline{x}$ and its neighbors $\underline{a}$ and $\underline{b}$. Define

$$\pi_{\underline{b} \Leftarrow \underline{x}}(x) = P(x|\epsilon^-) \text{ (past of } \underline{x}) \ , \tag{41.18}$$

$$\lambda_{\underline{b} \Rightarrow \underline{x}}(x) = P(\epsilon^+|x) \text{ (future of } \underline{x}) \ , \tag{41.19}$$

$$\pi_{\underline{x} \Leftarrow \underline{a}}(a) = P(a|\epsilon^-) \text{ (past of } \underline{a}) \tag{41.20}$$

and

$$\lambda_{\underline{x} \Rightarrow \underline{a}}(a) = P(\epsilon^+|a) \text{ (future of } \underline{a}) \ . \tag{41.21}$$

Define the Belief BEL in x to be

$$BEL_{\underline{x}}(x) = P(x|\epsilon) \ , \tag{41.22}$$

where

$$\underline{\epsilon} = \underline{\epsilon}^+ \cup \underline{\epsilon}^- \ . \tag{41.23}$$

Then

$$\begin{aligned} BEL_{\underline{x}}(x) &= \mathcal{N}(!x) P(\epsilon^+|x) P(x|\epsilon^-) & (41.24) \\ &= \mathcal{N}(!x) \lambda_{\underline{b} \Rightarrow \underline{x}}(x) \pi_{\underline{b} \Leftarrow \underline{x}}(x) \ . & (41.25) \end{aligned}$$

In analogy with the case of BP for a 3 node Markov chain, we can define the bnet Fig.41.6, which we refer to as the **BP 2-track bnet** for Fig.41.5. The TPMs, printed in blue, for the nodes of bnet Fig.41.6, are as follows:

$$P(\pi_{\underline{b} \Leftarrow \underline{x}} | \pi_{\underline{x} \Leftarrow \underline{a}}) = \prod_x \mathbb{1} \left(\pi_{\underline{b} \Leftarrow \underline{x}}(x) = \sum_a P(x|a) \pi_{\underline{x} \Leftarrow \underline{a}}(a) \right) \tag{41.26}$$

Figure 41.6: BP 2-track bnet for the bnet Fig.41.5.

$$P(B_{\underline{x}}|\pi_{\underline{b}\Leftarrow\underline{x}},\lambda_{\underline{b}\Rightarrow\underline{x}}) = \prod_x \mathbb{1}\left(B_{\underline{x}}(x) = BEL_{\underline{x}}(x)\right) \tag{41.27}$$

$$P(\lambda_{\underline{x}\Rightarrow\underline{a}}|\lambda_{\underline{b}\Rightarrow\underline{x}}) = \prod_a \mathbb{1}\left(\lambda_{\underline{x}\Rightarrow\underline{a}}(a) = \sum_x P(x|a)\lambda_{\underline{b}\Rightarrow\underline{x}}(x)\right) \tag{41.28}$$

Let us represent the Markov chain of Fig.41.5 by $\underline{x}_{nx-1} \leftarrow \ldots, \underline{x}_2 \leftarrow \underline{x}_1 \leftarrow \underline{x}_0$ where $nx = 5$. For any node $\underline{x}_i$ with parent $\underline{px}_i = \underline{x}_{i-1}$ and child $\underline{cx}_i = \underline{x}_{i+1}$, define the **memory matrix** $\mathcal{M}_{\underline{x}_i}$ for node $\underline{x}_i$ as

$$\mathcal{M}_{\underline{x}_i} = [\mathcal{M}^+_{\underline{x}_i}, \mathcal{M}^-_{\underline{x}_i}] \,, \tag{41.29}$$

where $+$ =future, $-$ =past, and

$$\mathcal{M}^+_{\underline{x}_i} = \begin{bmatrix} \pi_{\underline{cx}_i\Leftarrow\underline{x}_i}(\cdot) \\ \lambda_{\underline{cx}_i\Rightarrow\underline{x}_i}(\cdot) \end{bmatrix} \,, \quad \mathcal{M}^-_{\underline{x}_i} = \begin{bmatrix} \pi_{\underline{x}_i\Leftarrow\underline{px}_i}(\cdot) \\ \lambda_{\underline{x}_i\Rightarrow\underline{px}_i}(\cdot) \end{bmatrix} \,. \tag{41.30}$$

Note that

$$\mathcal{M}^-_{\underline{x}_i} = \mathcal{M}^+_{\underline{px}_i} \tag{41.31}$$

for all nodes $\underline{x}_i$. We will refer to Eqs.(41.31) as the **memory overlap conditions**.

We will also use a permuted version of the memory matrix

$$\mathcal{M}'_{\underline{x}_i} = [\mathcal{M}^{OUT}_{\underline{x}_i}, \mathcal{M}^{IN}_{\underline{x}_i}] \,, \tag{41.32}$$

where

$$\mathcal{M}^{OUT}_{\underline{x}_i} = \begin{bmatrix} \pi_{\underline{cx}_i\Leftarrow\underline{x}_i}(\cdot) \\ \lambda_{\underline{x}_i\Rightarrow\underline{px}_i}(\cdot) \end{bmatrix} \,, \quad \mathcal{M}^{IN}_{\underline{x}_i} = \begin{bmatrix} \pi_{\underline{x}_i\Leftarrow\underline{px}_i}(\cdot) \\ \lambda_{\underline{cx}_i\Rightarrow\underline{x}_i}(\cdot) \end{bmatrix} \,. \tag{41.33}$$

Unfortunately, 2-track bnets cannot be generalized in any obvious way from Markov chains to more complicated DAGs. An alternative to 2-track bnets that still

$$\underline{\mathcal{M}}_{\underline{\epsilon}^+} \longleftarrow \underline{\mathcal{M}}_{\underline{b}} \longleftarrow \underline{\mathcal{M}}_{\underline{x}} \longleftarrow \underline{\mathcal{M}}_{\underline{a}} \longleftarrow \mathcal{M}_{\underline{\epsilon}^-}$$

Figure 41.7: BP Memory Bnet for the bnet Fig.41.5.

carries message info in its nodes, are memory bnets. An BP **memory bnet** is a bnet which takes each node of an original bnet and adds a local memory to it. More specifically, it keeps that DAG but replaces each node $\underline{x}_i$ by a memory $\underline{\mathcal{M}}_{\underline{x}_i}$. Fig.41.7 shows the memory bnet for the bnet Fig.41.5. The TPM, printed in blue, for the node $\underline{\mathcal{M}}_{\underline{x}}$ of the memory bnet Fig.41.7, is as follows

$$P(\mathcal{M}_{\underline{x}_i}|\mathcal{M}_{\underline{n}\in nb(\underline{x}_i)}) = AB \ , \tag{41.34}$$

where

$$A = \mathbb{1}(\mathcal{M}^-_{\underline{x}_i} = \mathcal{M}^+_{\underline{px}_i}) \ , \tag{41.35}$$

and

$$B = \mathbb{1}(\mathcal{M}^{OUT}_{\underline{x}_i} = \mathcal{C}(\mathcal{M}^{IN}_{\underline{x}_i})) \ . \tag{41.36}$$

The function $\mathcal{C}$, which we will call the **BP local computation**, maps $\mathcal{M}^{IN}_{\underline{x}_i}$ into $\mathcal{M}^{OUT}_{\underline{x}_i}$. More explicitly, $\mathcal{C}$ is defined so that

$$B = \underbrace{P(\pi_{\underline{b}\Leftarrow\underline{x}}|\pi_{\underline{x}\Leftarrow\underline{a}})}_{B_\pi}\underbrace{P(\lambda_{\underline{x}\Rightarrow\underline{a}}|\lambda_{\underline{b}\Rightarrow\underline{x}})}_{B_\pi} \ , \tag{41.37}$$

where B_π and B_λ are given by Eqs.(41.26) and (41.28), respectively.

The BP memory bnet Fig. 41.7 is a deterministic bnet. A deterministic bnet is basically just a coupled system of equations (CSE) for some unknowns x_i. A CSE per se does not include with it a method for solving for the x_i. Such methods are not unique. For example, for the distributed soldier counting problem, the various methods that we described for counting soldiers are just different methods for solving the same CSE. One can describe a method for solving a CSE using a dynamic bnet.[3] To solve the CSE represented by the memory bnet Fig.41.7, we will use the dynamic bnet Fig.41.8. Henceforth, we will refer to Fig.41.8 as an **BP dynamic bnet** for Fig.41.7.

Next, we will explain the meaning of Fig.41.8. Fig.41.8 is a step by step recipe (i.e., algorithm) for solving a CSE, where the unknowns are memory matrices. Each step encoded in Fig.41.8 corresponds to a specific message sending event, where the messages are sent along the edges of the Markov chain Fig.41.5. These message sending events are portrayed in chronological order in Fig.41.9. In that figure, π

[3] The term dynamic bnet was used in Chapter 20 to mean a time inhomogeneous Markov chain, but here we are stretching its meaning to include Markov chains that aren't time inhomogeneous.

Figure 41.8: BP dynamic bnet for the bnet Fig.41.7.

Figure 41.9: Steps encoded in the bnet Fig.41.8. Note the similarity of this figure to Fig.41.1 (d) for soldier counting.

messages are indicated by dashed red arrows, and λ messages by dotted red arrows. These steps, or message sending events, lead to an updating of the memory matrices that we are solving for. Each step propagates information between the memory nodes. In the usual Pearl BP algo, the evidence nodes initiate the BP chain of message passing events. These events continue until the memory matrices reach an equilibrium and the CSE is solved.

To use bnet Fig.41.8, we need to specify the **initial conditions** (i.e., the value of $\mathcal{M}^{(0)}_{\underline{x}_i}$ for all i). For that, one can use

$$\pi^{(0)}_{\underline{px}_0 \Leftarrow \underline{x}_0} = P(x_0) \ , \tag{41.38}$$

$$\lambda^{(0)}_{\underline{cx}_i \Rightarrow \underline{x}_{nx-1}}(x_{nx-1}) = \delta(x_{nx-1}, x'_{nx-1}) \ . \tag{41.39}$$

All other $\mathcal{M}^{(0)}_{\underline{x}_i}$ entries for all i can be set to 1.

The TPMs, printed in blue, for the nodes of Fig.41.8, can all be summarized by

$$P(\mathcal{M}^{(t)}_{\underline{x}_i} | \mathcal{M}^{(t-1)}_{\underline{n} \in nb(\underline{x}_i)}, \mathcal{M}^{(t-1)}_{\underline{x}_i}) = AB \ , \tag{41.40}$$

where

$$A = \begin{cases} \mathbb{1}(\mathcal{M}^{(t)-}_{\underline{x}_i} = \mathcal{M}^{(t-1)+}_{\underline{px}_i}) & \text{if input from } \underline{px}_i \\ \\ \mathbb{1}(\mathcal{M}^{(t)+}_{\underline{x}_i} = \mathcal{M}^{(t-1)-}_{\underline{cx}_i}) & \text{if input from } \underline{cx}_i \end{cases} \ , \tag{41.41}$$

and

$$B = \mathbb{1}(\mathcal{M}^{(t)OUT}_{\underline{x}_i} = \mathcal{C}(\mathcal{M}^{(t)IN}_{\underline{x}_i})) \ . \tag{41.42}$$

The function $\mathcal{C}$, which we will call the **BP local computation**, maps $\mathcal{M}^{(t)IN}_{\underline{x}_i}$ into $\mathcal{M}^{(t)OUT}_{\underline{x}_i}$. More explicitly, $\mathcal{C}$ is defined so that

$$B = B_\pi B_\lambda \tag{41.43}$$

where

$$B_\pi = \prod_x \mathbb{1} \left(\underbrace{\pi^{(t)}_{\underline{b} \Leftarrow \underline{x}}(x)}_{OUT} = \sum_a P(x|a) \underbrace{\pi^{(t)}_{\underline{x} \Leftarrow \underline{a}}(a)}_{IN} \right) \tag{41.44}$$

and

$$B_\lambda = \prod_a \mathbb{1} \left(\underbrace{\lambda^{(t)}_{\underline{x} \Rightarrow \underline{a}}(a)}_{OUT} = \sum_x P(x|a) \underbrace{\lambda^{(t)}_{\underline{b} \Rightarrow \underline{x}}(x)}_{IN} \right) \ . \tag{41.45}$$

The basic idea behind Eq.(41.42), which we will call the **memory updating equation**, is simple: the memory overlap conditions translate the information from time $t-1$ to t, and then the local computation translates IN to OUT at fixed time t.

41.4 BP Algorithm for Polytrees

Consider Fig.41.10, which illustrates a bnet node $\underline{x}$ receiving and sending messages to its neighbors. The π messages (probability functions) travel downstream (i.e.,

they carry info in the direction of the graph arrows, towards the future) and are indicated by a dashed arrow or by a left double arrow $\Leftarrow$. The λ messages (likelihood functions) travel upstream (i.e., they carry info opposite to direction of the graph arrows, towards the past) and are indicated by a dotted arrow or by a right double arrow $\Rightarrow$.

Note that argument arg of the $\pi(arg)$ and $\lambda(arg)$ functions is always the same as the letter in the subscript that is closest to the argument.

Note that in Fig.41.10, we indicate messages that travel "downstream" (resp., "upstream"), by arrows with dashed (resp., dotted) lines as shafts. Mnemonic: think of the shaft as a velocity vector field for the message. You travel faster when you swim downstream as opposed to upstream.

$pa(\underline{x})$ = parents of node $\underline{x}$
$ch(\underline{x})$ = children of node $\underline{x}$
$nb(\underline{x}) = pa(\underline{x}) \cup ch(\underline{x})$ = neighbors of node $\underline{x}$

Figure 41.10: Node $\underline{x}$ receiving and sending messages to its neighbors. (neighbors= parents and children).

We define a **memory matrix** $\mathcal{M}_{\underline{x}}$ for node $\underline{x}$ as

$$\mathcal{M}_{\underline{x}} = [\mathcal{M}^+_{\underline{x}}, \mathcal{M}^-_{\underline{x}}] \ , \tag{41.46}$$

where $+$ =future, $-$ =past, and

$$\mathcal{M}^+_{\underline{x}} = \left[\ \pi_{\underline{b}\Leftarrow\underline{x}}(\cdot) \quad \lambda_{\underline{b}\Rightarrow\underline{x}}(\cdot) \ \right]_{\underline{b}\in ch(\underline{x})} = [\mathcal{M}^+_{\underline{b},\underline{x}}]_{\underline{b}\in ch(\underline{x})} \ , \tag{41.47}$$

$$\mathcal{M}^-_{\underline{x}} = \left[\begin{array}{c} \pi_{\underline{x}\Leftarrow\underline{a}}(\cdot) \\ \lambda_{\underline{x}\Rightarrow\underline{a}}(\cdot) \end{array} \right]_{\underline{a}\in pa(\underline{x})} = [\mathcal{M}^-_{\underline{x},\underline{a}}]_{\underline{a}\in pa(\underline{x})} \ . \tag{41.48}$$

Note that

$$\mathcal{M}^-_{\underline{x},\underline{a}} = \mathcal{M}^+_{\underline{a},\underline{x}} \tag{41.49}$$

for every arrow $\underline{x} \leftarrow \underline{a}$. We will refer to Eqs.(41.49) as the **memory overlap conditions**.

We will also use a permuted version of the memory matrix

$$\mathcal{M}'_{\underline{x}} = [\mathcal{M}^{OUT}_{\underline{x}}, \mathcal{M}^{IN}_{\underline{x}}] \ , \tag{41.50}$$

where

$$\mathcal{M}^{OUT}_{\underline{x}} = \begin{pmatrix} [\pi_{\underline{b} \Leftarrow \underline{x}}(\cdot)]_{\underline{b} \in ch(\underline{x})} \\ [\lambda_{\underline{x} \Rightarrow \underline{a}}(\cdot)]_{\underline{a} \in pa(\underline{x})} \ , \end{pmatrix} = [\mathcal{M}^{OUT}_{\underline{x},\underline{n}}]_{\underline{n} \in nb(\underline{x})} \ , \tag{41.51}$$

$$\mathcal{M}^{IN}_{\underline{x}} = \begin{pmatrix} [\pi_{\underline{x} \Leftarrow \underline{a}}(\cdot)]_{\underline{a} \in pa(\underline{x})} \\ [\lambda_{\underline{b} \Rightarrow \underline{x}}(\cdot)]_{\underline{b} \in ch(\underline{x})} \end{pmatrix} = [\mathcal{M}^{IN}_{\underline{x},\underline{n}}]_{\underline{n} \in nb(\underline{x})} \ . \tag{41.52}$$

For times $t = 0, 1, \ldots, T-1$, we calculate $\mathcal{M}^{(t)}_{\underline{x}}$ in two steps: first we calculate $\mathcal{M}^{(t)IN}_{\underline{x}}$ from earlier memories at time $t-1$, then we calculate $\mathcal{M}^{(t)OUT}_{\underline{x}}$:

Figure 41.11: Subgraph of a bnet showing two cases (RULE 1 and RULE 2) of message info flow. The yellow node is a gossip monger. It receives messages from all the green nodes, and then it relays a joint message to the red node. Union of green nodes and the red node = full neighborhood of yellow node. There are two possible cases: the red node is either a parent or a child of the yellow one. As usual, we use arrows with dashed (resp., dotted) shafts for downstream (resp., upstream) messages. Blue boxes indicate Markov chain case.

An **evidence node** is a node whose TPM is a delta function set to a particular state of the node. We will assume, without loss of generality, that all evidence nodes are leaf nodes. If that is not the case, any evidence node $\underline{e}$ that is not a leaf node, can be given a new companion leaf node $\underline{l}$ connected to $\underline{e}$ by an arrow $\underline{l} \leftarrow \underline{e}$, and such that $\underline{l}$ has a delta function TPM.

1. **Calculating $\mathcal{M}_{\underline{x}}^{(t)IN}$ from signals received from $\underline{n} \in nb(\underline{x})$, sent at earlier time $t-1$:**

   Set

$$\mathcal{M}_{\underline{x},\underline{a}}^{(t)-}|_{\pi} = \mathcal{M}_{\underline{a},\underline{x}}^{(t-1)+}|_{\pi} \ , \tag{41.53}$$

   for all $\underline{a} \in pa(\underline{x})$, and

$$\mathcal{M}_{\underline{b},\underline{x}}^{(t)+}|_{\lambda} = \mathcal{M}_{\underline{x},\underline{b}}^{(t-1)-}|_{\lambda} \ , \tag{41.54}$$

   for all $\underline{b} \in ch(\underline{x})$. By $X|_{\lambda}$ (resp., $X|_{\pi}$) we mean the λ (resp., π) component of X.

2. **Calculating $\mathcal{M}_{\underline{x}}^{(t)OUT}$ from already calculated $\mathcal{M}_{\underline{x}}^{(t)IN}$:**

   Let $\underline{a}^{na} = (\underline{a}_i)_{i=0,1,\ldots,na-1}$ denote the parents of $\underline{x}$ and $\underline{b}^{nb} = (\underline{b}_i)_{i=0,1,\ldots,nb-1}$ its children.

   Define

$$\pi_{\underline{x}}(x) = \sum_{a^{na}} P(x|a^{na}) \prod_i \pi_{\underline{x} \Leftarrow \underline{a}_i}(a_i) \tag{41.55}$$

$$= E_{\underline{a}^{na}}[P(x|a^{na})] \tag{41.56}$$

   (boundary case: if $\underline{x}$ is a root node, use $\pi_{\underline{x}}(x) = P(x)$.) and

$$\lambda_{\underline{x}}(x) = \prod_i \lambda_{\underline{b}_i \Rightarrow \underline{x}}(x) \ . \tag{41.57}$$

   (boundary case: if $\underline{x}$ is a leaf node, use $\lambda_{\underline{x}}(x) = 1$.)

   - **RULE 1: (red parent)**
     From the $\lambda_{\underline{x} \Rightarrow \underline{a}}$ panel of Fig.41.11, we get

$$\underbrace{\lambda_{\underline{x} \Rightarrow \underline{a}_i}(a_i)}_{OUT} = \mathcal{N}(!a_i) \sum_x \left[\underbrace{\lambda_{\underline{x}}(x)}_{IN} \sum_{(a_k)_{k \neq i}} \left(P(x|a^{na}) \prod_{k \neq i} \underbrace{\pi_{\underline{x} \Leftarrow \underline{a}_k}(a_k)}_{IN} \right) \right] \tag{41.58}$$

$$= \mathcal{N}(!a_i) \sum_x \left[\lambda_{\underline{x}}(x) E_{(\underline{a}_k)_{k \neq i}}[P(x|a^{na})] \right] \tag{41.59}$$

$$= \mathcal{N}(!a_i) E_{(\underline{a}_k)_{k \neq i}} E_{\underline{x}|a^{na}} \lambda_{\underline{x}}(x) \tag{41.60}$$

     (boundary case: if $\underline{x}$ is a root node, use $\lambda_{\underline{x} \Rightarrow \underline{a}_i}(a_i) = \mathcal{N}(!a_i)$.)

- **RULE 2: (red child)**
  From the $\pi_{\underline{b}\Leftarrow\underline{x}}$ panel of Fig.41.11, we get

$$\underbrace{\pi_{\underline{b}_i\Leftarrow\underline{x}}(x)}_{OUT} = \mathcal{N}(!x)\underbrace{\pi_{\underline{x}}(x)}_{IN}\prod_{k\neq i}\underbrace{\lambda_{\underline{b}_k\Rightarrow\underline{x}}(x)}_{IN} \tag{41.61}$$

  (boundary case: if $\underline{x}$ is a leaf node, use $\pi_{\underline{b}_i\Leftarrow\underline{x}}(x) = \mathcal{N}(!x)\pi_{\underline{x}}(x)$.)

In the above equations, if the range set of a product is empty, then define the product as 1; i.e., $\prod_{k\in\emptyset} F(k) = 1$.

Claim: Define

$$BEL^{(t)}(x) = \mathcal{N}(!x)\lambda^{(t)}_{\underline{x}}(x)\pi^{(t)}_{\underline{x}}(x) \ . \tag{41.62}$$

Then

$$\lim_{t\to\infty} BEL^{(t)}(x) = P(x|\epsilon) \ . \tag{41.63}$$

This says that the belief in $\underline{x} = x$ converges to $P(x|\epsilon)$ and it equals the product of messages received from all parents and children of $\underline{x} = x$.

41.4.1 How BP algo for polytrees reduces to the BP algo for Markov chains

It is instructive to see how the BP algo for polytrees reduces to BP algo for Markov chains.

For a Markov chain, node $\underline{x}$ has a single parent (i.e., ancestor) $\underline{a}$ and a single child $\underline{b}$.

Therefore, Eqs.(41.55) and (41.57) reduce to

$$\pi_{\underline{x}}(x) = \sum_a P(x|a)\pi_{\underline{x}\Leftarrow\underline{a}}(a) \tag{41.64}$$

and

$$\lambda_{\underline{x}}(x) = \lambda_{\underline{b}\Rightarrow\underline{x}}(x) \ . \tag{41.65}$$

RULE 1 given by Eq.(41.58) reduces to

$$\lambda_{\underline{x}\Rightarrow\underline{a}}(a) = \mathcal{N}(!a)\sum_x \lambda_{\underline{x}}(x)P(x|a) \tag{41.66}$$

$$= \mathcal{N}(!a)\sum_x \lambda_{\underline{b}\Rightarrow\underline{x}}(x)P(x|a) \tag{41.67}$$

RULE 2 given by Eq.(41.61) reduces to

$$\pi_{\underline{b}\Leftarrow\underline{x}}(x) = \mathcal{N}(!x)\pi_{\underline{x}}(x) \tag{41.68}$$

$$= \sum_a P(x|a)\pi_{\underline{x}\Leftarrow\underline{a}}(a) \ . \tag{41.69}$$

41.5 Derivation of BP Algorithm for Polytrees

This derivation is taken from the 1988 book Ref.[35] by Judea Pearl, where it is presented very lucidly. We only made some minor changes in notation.

Notation

The BP algorithm yields an expansion for $P(x|\epsilon)$.

$\underline{x}$= the focus node, arbitrary node of bnet that we are focusing on to calculate its $P(x|\epsilon)$.

$(\underline{a}_i)_{i=0,1,\ldots,na-1}$. = parent nodes (mnemonic: a=ancestor) of $\underline{x}$

$(\underline{b}_i)_{i=0,1,\ldots,nb-1}$. = children nodes of $\underline{x}$.

$\underline{\epsilon}$= set of nodes for which there is evidence; that is, $\underline{\epsilon} = \epsilon$, so the state of these nodes is fixed.

$\underline{\epsilon}^-_{\underline{x}} = \underline{\epsilon} \cap an(\underline{x})$ (evidence in past of $\underline{x}$)[4]

$\epsilon^-_{\underline{x}\underline{a}_i} = \epsilon^-_{\underline{x}} \cap an(\underline{a}_i)$.

Note that $\epsilon^-_{\underline{x}} = \cup_i \epsilon^-_{\underline{x}\underline{a}_i}$

$\underline{\epsilon}^+_{\underline{x}} = \underline{\epsilon} \cap [de(\underline{x}) \cup \underline{x}]$ (evidence in future of $\underline{x}$)

$\epsilon^+_{\underline{x}\underline{b}_i} = \epsilon^+_{\underline{x}} \cap [de(\underline{b}_i) \cup \underline{b}_i]$.

Note that $\epsilon^+_{\underline{x}} = \cup_i \epsilon^+_{\underline{x}\underline{b}_i}$

Note that $\underline{\epsilon} = \underline{\epsilon}^+_{\underline{x}} \cup \epsilon^-_{\underline{x}}$

$$\pi_{\underline{x}}(x) = P(x|\epsilon^-_{\underline{x}}) \tag{41.70}$$

$$\pi_{\underline{x}\Leftarrow\underline{a}_i}(a_i) = P(a_i|\epsilon^-_{\underline{x}\underline{a}_i}) \tag{41.71}$$

$$\pi_{\underline{b}_i\Leftarrow\underline{x}}(x) = P(x|\epsilon^-_{\underline{x}\underline{b}_i}) \tag{41.72}$$

$$\lambda_{\underline{x}}(x) = P(\epsilon^+_{\underline{x}}|x) \tag{41.73}$$

$$\lambda_{\underline{b}_i\Rightarrow\underline{x}}(x) = P(\epsilon^+_{\underline{x}\underline{b}_i}|x) \tag{41.74}$$

$$\lambda_{\underline{x}\Rightarrow\underline{a}_i}(a_i) = P(\epsilon^+_{\underline{x}\underline{a}_i}|a_i) \tag{41.75}$$

Expansions of $\lambda_{\underline{x}}(x)$ and $\pi_{\underline{x}}(x)$ into products of single node messages.

[4] Careful: Chapter 4 of Ref.[35] uses $-$ indicate the future and $+$ to indicate the past. This is the opposite of our notation.

$$\underbrace{P(x|\epsilon^-_{\underline{x}})}_{\pi_{\underline{x}}(x)} = P(x|\cup_i \epsilon^-_{\underline{xa}_i}) \tag{41.76}$$

$$= \sum_{a^{na}} P(x|a^{na})P(a^{na}|\cup_i \epsilon^-_{\underline{xa}_i}) \tag{41.77}$$

$$= \sum_{a^{na}} P(x|a^{na}) \prod_i \underbrace{P(a_i|\epsilon^-_{\underline{xa}_i})}_{\pi_{\underline{x}\Leftarrow\underline{a}_i}(a_i)} \tag{41.78}$$

$$\underbrace{P(\epsilon^+_{\underline{x}}|x)}_{\lambda_{\underline{x}}(x)} = \prod_i \underbrace{P(\epsilon^+_{\underline{xb}_i}|x)}_{\lambda_{\underline{b}_i\Rightarrow\underline{x}}(x)} \tag{41.79}$$

Note that past and future evidences $\epsilon^-_{\underline{x}}$ and $\epsilon^+_{\underline{x}}$ that are causally connected to $\underline{x}$ are conditionally independent at fixed $\underline{x}$:

$$P(\epsilon^+_{\underline{x}}, \epsilon^-_{\underline{x}}|x) = P(\epsilon^+_{\underline{x}}|x)P(\epsilon^-_{\underline{x}}|x) \ . \tag{41.80}$$

This observation is key to the proof of the following claim:

Claim 51

$$P(x|\epsilon^+_{\underline{x}}, \epsilon^-_{\underline{x}}) = P(\epsilon^+_{\underline{x}}|x)P(x|\epsilon^-_{\underline{x}})\frac{1}{P(\epsilon^+_{\underline{x}}|\epsilon^-_{\underline{x}})} \tag{41.81}$$

$$= \mathcal{N}(!x)P(\epsilon^+_{\underline{x}}|x)P(x|\epsilon^-_{\underline{x}}) \tag{41.82}$$

$$= \mathcal{N}(!x) \quad (\epsilon^+_{\underline{x}} \leftarrow x \leftarrow \epsilon^-_{\underline{x}}) \tag{41.83}$$

$$= \mathcal{N}(!x)\lambda_{\underline{x}}(x)\pi_{\underline{x}}(x) \tag{41.84}$$

proof:

$$P(x|\epsilon^+_{\underline{x}}, \epsilon^-_{\underline{x}}) = P(\epsilon^+_{\underline{x}}, \epsilon^-_{\underline{x}}|x)\frac{P(x)}{P(\epsilon^+_{\underline{x}}, \epsilon^-_{\underline{x}})} \tag{41.85}$$

$$= P(\epsilon^+_{\underline{x}}|x)P(\epsilon^-_{\underline{x}}|x)\frac{P(x)}{P(\epsilon^+_{\underline{x}}, \epsilon^-_{\underline{x}})} \tag{41.86}$$

$$= P(\epsilon^+_{\underline{x}}|x)P(x|\epsilon^-_{\underline{x}})\frac{P(\epsilon^-_{\underline{x}})}{P(\epsilon^+_{\underline{x}}, \epsilon^-_{\underline{x}})} \tag{41.87}$$

$$= P(\epsilon^+_{\underline{x}}|x)P(x|\epsilon^-_{\underline{x}})\frac{1}{P(\epsilon^+_{\underline{x}}|\epsilon^-_{\underline{x}})} \tag{41.88}$$

QED

Next we prove BP rules 1 and 2.

Figure 41.12: This figure is used in the derivation of the BP RULE 1.

- **RULE 1 (red parent)**

  Note that

$$\epsilon^+_{\underline{x}} \cup \cup_{k\neq i}\epsilon^-_{\underline{xa}_k} = (\epsilon^+_{\underline{x}} \cup \epsilon^-_{\underline{x}}) - \epsilon^-_{\underline{xa}_i} \tag{41.89}$$

$$= \epsilon^+_{\underline{xa}_i} \tag{41.90}$$

Let $y = (a_k)_{k\neq i}$ and $\epsilon^-_{\underline{y}} = (\epsilon^-_{\underline{xa}_k})_{k\neq i}$.

$$\underbrace{P(\epsilon^+_{\underline{xa}_i}|a_i)}_{\lambda_{\underline{x}\Rightarrow\underline{a}_i}(a_i)} = P(\epsilon^+_{\underline{x}}, \epsilon^-_{\underline{y}}|a_i) \tag{41.91}$$

$$= \sum_x \sum_y P(\epsilon^+_{\underline{x}}, \epsilon^-_{\underline{y}}|x, y)P(x, y|a_i) \tag{41.92}$$

$$= \sum_x \sum_y P(\epsilon^+_{\underline{x}}|x)P(\epsilon^-_{\underline{y}}|y)P(x|y, a_i)P(y|a_i) \tag{41.93}$$

$$= P(\epsilon^-_{\underline{y}}) \sum_x \sum_y P(\epsilon^+_{\underline{x}}|x)\frac{P(y|\epsilon^-_{\underline{y}})}{P(y)}P(x|y, a_i)\underbrace{P(y|a_i)}_{=P(y)} \tag{41.94}$$

$$= \mathcal{N}(!a_i) \sum_x \sum_y P(\epsilon^+_{\underline{x}}|x)P(x|\underbrace{y, a_i}_{a^{na}})P(y|\epsilon^-_{\underline{y}}) \tag{41.95}$$

$$= \mathcal{N}(!a_i) \sum_x \underbrace{P(\epsilon^+_{\underline{x}}|x)}_{\lambda_{\underline{x}}(x)} \sum_{(a_k)_{k\neq i}} P(x|a^{na}) \prod_{k\neq i} \underbrace{P(a_k|\epsilon^-_{\underline{xa}_k})}_{\pi_{\underline{x}\Leftarrow\underline{a}_k}(a_k)} \tag{41.96}$$

Figure 41.13: This figure is used in the derivation of the BP RULE 2.

- **RULE 2 (red child)**

  Note that

$$(\cup_{k\neq i}\epsilon^+_{\underline{x}\underline{b}_k}) \cup \epsilon^-_{\underline{x}} = (\epsilon^+_{\underline{x}} \cup \epsilon^-_{\underline{x}}) - \epsilon^+_{\underline{x}\underline{b}_i} \tag{41.97}$$

$$= \epsilon^-_{\underline{x}\underline{b}_i} \tag{41.98}$$

$$\underbrace{P(x|\epsilon^-_{\underline{x}\underline{b}_i})}_{\pi_{\underline{b}_i \Leftarrow \underline{x}}(x)} = P(x|(\epsilon^+_{\underline{x}\underline{b}_k})_{k\neq i}, \epsilon^-_{\underline{x}}) \tag{41.99}$$

$$= \mathcal{N}(!x)P((\epsilon^+_{\underline{x}\underline{b}_k})_{k\neq i}|x)P(x|\epsilon^-_{\underline{x}}) \tag{41.100}$$

$$= \mathcal{N}(!x)\left(\prod_{k\neq i}\underbrace{P(\epsilon^+_{\underline{x}\underline{b}_k}|x)}_{\lambda_{\underline{b}_k \Rightarrow \underline{x}}(x)}\right)\underbrace{P(x|\epsilon^-_{\underline{x}})}_{\pi_{\underline{x}}(x)} \tag{41.101}$$

41.6 Example of BP algo for a Tree

In this section, we describe how to apply the BP algo to the tree bnet Fig.41.14. In Fig.41.14, if we replace each integer i by the random variable $\underline{A}_i$, we get an **original bnet**, and if we replace each i by $\underline{\mathcal{M}}_{\underline{A}_i}$, we get the **BP memory bnet** of the original bnet. In Fig.41.14, the magenta nodes are evidence nodes and the green ones aren't.

We want to solve for the memory matrices of the memory bnet. To do so, we use the **BP dynamic bnet** Fig.41.15. The steps encoded in the dynamic bnet are shown in Fig.41.16. Fig.41.16 has frames in chronological order, showing the direction of travel of the $\pi\&\lambda$ information. This sequence of frames also indicates the order in which we solve for the entries of the memory matrices. The information first emanates from the evidence nodes. It propagates generally upstream, although some nodes can generate downstream flow. Some of the info reaches the root node and is reflected there. The root node is the only one that is capable of reflection (i.e., instant output along an arrow, in response to input along that arrow). Eventually, all info reaches the leaf nodes via downstream propagation and is absorbed there.

Figure 41.14: Example tree bnet used to illustrate BP.

Figure 41.15: BP dynamic bnet for the bnet Fig.41.14.

Figure 41.16: Steps encoded in the bnet Fig.41.15.

41.7 Bipartite bnets

By a **bipartite bnet** we will mean a bnet in which all nodes are either root nodes (parentless) or leaf nodes (childless). BP simplifies when dealing with bipartite bnets. In this section, we will explain how it simplifies. But before doing so, let us explain how the following two types of diagrams can be replaced by equivalent bipartite bnets:

- Factor Graphs
- Tree bnets

Consider a product $g = \prod_\alpha f_\alpha$ of scalar functions $f_\alpha : S_{\underline{x}_0} \times S_{\underline{x}_1} \times \ldots S_{\underline{x}_{nx-1}} \to \mathbb{R}$ for $\alpha = 0, 1, \ldots, nf - 1$. For instance, consider $g : S_{\underline{x}_0} \times S_{\underline{x}_1} \times S_{\underline{x}_2} \to \mathbb{R}$ defined by:

$$g(x_0, x_1, x_2) = f_0(x_0) f_1(x_0, x_1) f_2(x_0, x_1) f_3(x_1, x_2) \ . \tag{41.102}$$

The **factor graph** for this function g is given by Fig.41.17.

Figure 41.17: Factor graph for function g defined by Eq.(41.102).

Figure 41.18: Bipartite bnet corresponding to factor graph Fig.41.17.

One can map any factor graph (the "source") to a special bipartite bnet (the "image"), as follows. Replace each x_i by $\underline{x}_i \in S_{\underline{x}_i}$ for $i = 0, 1, \ldots, nx - 1$ and each f_α by $\underline{f}_\alpha$ for $\alpha = 0, 1, \ldots, nf - 1$. Then replace the connections (edges) of the factor graph by arrows from $\underline{x}_i$ to $\underline{f}_\alpha$. For example, Fig.41.18 is the image bipartite bnet of the source factor graph Fig.41.17.

Let $\underline{x}^{nx} = (\underline{x}_0, \underline{x}_1, \ldots, \underline{x}_{nx-1})$ and $\underline{f}^{nf} = (\underline{f}_0, \underline{f}_1, \ldots, \underline{f}_{nf-1})$. Let[5] $f_\alpha \in \{0,1\}$ for all α, and $y_\alpha = f_\alpha(x_{nb(\underline{f}_\alpha)})$. Here we are using $nb(\underline{f}_\alpha)$ to denote the neighborhood of node $\underline{f}_\alpha$ in the image bipartite bnet, and we are using x_S to denote $(x_i)_{i \in S}$. Without loss of generality, we will assume that $y_\alpha \in [0,1]$ for all α. Then we define the node TPMs, printed in blue, for the image bipartite bnet, as follows.

$$P(f_\alpha | x_{nx(\underline{f}_\alpha)}) = y_\alpha \delta(f_\alpha, 1) + [1 - y_\alpha]\delta(f_\alpha, 0) \tag{41.103}$$

for $\alpha = 0, 1, \ldots, nf - 1$ and

$$P_{\underline{x}_i}(x_i) = \text{arbitrary prior} \tag{41.104}$$

for $i = 0, 1, \ldots, nx - 1$.

Note that

$$P(f^{nf} = 1^{nf} | x^{nx}) = \prod_\alpha f_\alpha(x_{nb(\underline{f}_\alpha)}) \ . \tag{41.105}$$

A **tree bnet** is a bnet for which all nodes have exactly one parent except for the apex root node which has none. A tree bnet is very much like the filing system in a computer.

One can map a tree bnet (the "source") into an equivalent bipartite bnet (the "image") as follows. Replace each arrow

$$\underline{x} \longrightarrow \underline{y} \tag{41.106}$$

of the tree bnet by

$$\underline{x} \longrightarrow \underline{P_{\underline{y}|\underline{x}}} \longleftarrow \underline{y} \ . \tag{41.107}$$

For example, the tree bnet Fig.41.19 has the image bipartite bnet given by Fig.41.20. The bnet Fig.41.21 is just a different way of drawing the bnet Fig.41.20.

The node TPMs, printed in blue, for the image bipartite bnet Fig.41.20, are as follows. We express the TPMs of the image bnet in terms of the TPMs of the source bnet Fig.41.19. Let

$$P(P_{\underline{y}|\underline{x}} | x, y) = P_{\underline{y}|\underline{x}}(y|x)\delta(P_{\underline{y}|\underline{x}}, 1) + (1 - P_{\underline{y}|\underline{x}}(y|x))\delta(P_{\underline{y}|\underline{x}}, 0) \tag{41.108}$$

for all the leaf nodes $\underline{P_{\underline{y}|\underline{x}}} \in \{0,1\}$ of the image bipartite bnet. Also, let

$$P_{\underline{y}}(y) = \text{arbitrary prior} \tag{41.109}$$

[5] Note that we are using f_α to denote both a function $f_\alpha(\cdot)$ and a Boolean value. Which one we mean will be clear from context. f_α could also be used to denote, besides a function and a Boolean value, the real number $y_\alpha = f_\alpha(x_{nb(\underline{f}_\alpha)})$. However, we won't be using it that third way in this chapter.

Figure 41.19: Example of a tree bnet.

Figure 41.20: Bipartite bnet corresponding to tree bnet Fig.41.19.

Figure 41.21: Different way of drawing the bnet Fig.41.20.

for all the root nodes $\underline{y}$ of the image bipartite bnet except when $\underline{y}$ corresponds to the root node $\underline{A}$ of the source tree bnet. In that exceptional case,

$$P_{\underline{y}}(y) = P_{\underline{A}}(y) \ . \tag{41.110}$$

41.8 BP for bipartite bnets (BP-BB)

For a bipartite bnet as defined above, with root nodes $\underline{x}_i$ and leaf nodes $\underline{f}_\alpha$, let

$$nb(i) = \{\alpha : \underline{f}_\alpha \in nb(\underline{x}_i)\} \ , \tag{41.111}$$

$$nb(\alpha) = \{i : \underline{x}_i \in nb(\underline{f}_\alpha)\} \ , \tag{41.112}$$

$$m_{\alpha \Leftarrow i}(x_i) = \pi_{\underline{f}_\alpha \Leftarrow \underline{x}_i}(x_i) \ , \tag{41.113}$$

$$m_{\alpha \Rightarrow i}(x_i) = \lambda_{\underline{f}_\alpha \Rightarrow \underline{x}_i}(x_i) \ , \tag{41.114}$$

Figure 41.22: Fig.41.11 becomes this figure for the special case of a bipartite bnet. Union of green nodes and the red node = full neighborhood of yellow node. There are two possible cases: the red node is either a parent or a child of the yellow node.

Next we will show how to find $m^{(t)}_{\alpha \Leftarrow i}$ and $m^{(t)}_{\alpha \Rightarrow i}$ from $m^{(t-1)}_{\alpha \Leftarrow i}$ and $m^{(t-1)}_{\alpha \Rightarrow i}$.

1. **Traversing an x (i.e., root) node.**

   See the $m_{\underline{f}_2 \Leftarrow \underline{x}_2}$ panel of Fig.41.22.

   For $i = 0, 1, \ldots, nx - 1$, if $\alpha \in nb(i)$, then,

$$m^{(t)}_{\alpha \Leftarrow i}(x_i) = \prod_{\beta \in nb(i) - \alpha} m^{(t-1)}_{\beta \Rightarrow i}(x_i) \ , \tag{41.115}$$

   whereas if $\alpha \notin nb(i)$

$$m^{(t)}_{\alpha \Leftarrow i}(x_i) = m^{(t-1)}_{\alpha \Leftarrow i}(x_i) \ . \tag{41.116}$$

2. **Traversing an f (i.e., leaf) node.**

   See the $m_{\underline{f}_2 \Rightarrow \underline{x}_2}$ panel of Fig.41.22.

For $\alpha = 0, 1, \ldots, nf-1$, if $i \in nb(\alpha)$, then

$$m^{(t)}_{\alpha \Rightarrow i}(x_i) = \sum_{(x_k)_{k \in nb(\alpha)-i}} f_\alpha(x_{nb(\alpha)}) \prod_{k \in nb(\alpha)-i} m^{(t-1)}_{\alpha \Leftarrow k}(x_k) \tag{41.117}$$

$$= E^{(t-1)}_{(x_k)_{k \in nb(\alpha)-i}}[f_\alpha(x_{nb(\alpha)})] \ , \tag{41.118}$$

whereas if $i \notin nb(\alpha)$

$$m^{(t)}_{\alpha \Rightarrow i}(x_i) = m^{(t-1)}_{\alpha \Rightarrow i}(x_i) \ . \tag{41.119}$$

In the above equations, if the range set of a product is empty, then define the product as 1; i.e., $\prod_{k \in \emptyset} F(k) = 1$.

Claim:

$$P(x_i|\epsilon) = \lim_{t \to \infty} \mathcal{N}(!x_i) \prod_{\alpha \in nb(i)} m^{(t)}_{\alpha \Rightarrow i}(x_i) \tag{41.120}$$

and

$$P(x_{nb(\alpha)}|\epsilon) = \lim_{t \to \infty} \mathcal{N}(!x_{nb(\alpha)}) f_\alpha(x_{nb(\alpha)}) \prod_{k \in nb(\alpha)} m^{(t)}_{\alpha \Leftarrow k}(x_k) \ . \tag{41.121}$$

41.8.1 BP-BB and general BP agree on Markov chains

It is instructive to compare the belief values (i.e., $P(x_i|\epsilon)$) obtained by applying the general (i.e., polytree) BP and BP-BB algorithms to a Markov chain. Next we show that both algorithms yield the same belief values.

Figure 41.23: Traversing a root node of a Markov chain (a)Propagation towards left (i.e., towards future). (b)Propagation towards right (i.e., towards past).

Consider the BP-BB rule for traversing a root node. When traveling towards the left as in Fig.41.23 (a), it implies that

$$m_{\alpha \Rightarrow 2}(x_2) = m_{\beta \Leftarrow 2}(x_2) \ , \tag{41.122}$$

and when traveling towards the right as in Fig.41.23 (b), it implies that

$$m_{\beta\Rightarrow 2}(x_2) = m_{\alpha\Leftarrow 2}(x_2) \ . \tag{41.123}$$

Figure 41.24: Traversing a leaf node of a Markov chain (a)Propagation towards left (i.e., towards future). (b)Propagation towards right (i.e., towards past).

Now consider the BP-BB rule for traversing a leaf node. When traveling to the left as in Fig.41.24 (a), it implies that

$$\underbrace{m_{\alpha\Rightarrow 2}(x_2)}_{\lambda} = \sum_{x_1} P(x_2|x_1) \underbrace{m_{\alpha\Leftarrow 1}(x_1)}_{\pi} \ . \tag{41.124}$$

One can rewrite the left and right hand sides (LHS, RHS) of Eq.(41.124) as follows

$$RHS = \sum_{x_1} P(x_2|x_1)\pi_{\alpha\Leftarrow 1}(x_1) \ , \tag{41.125}$$

and

$$LHS = m_{\alpha\Rightarrow 2}(x_2) = m_{\beta\Leftarrow 2}(x_2) = \pi_{\beta\Leftarrow 2}(x_2) \ , \tag{41.126}$$

Therefore

$$\pi_{\beta\Leftarrow 2}(x_2) \sum_{x_1} P(x_2|x_1)\pi_{\alpha\Leftarrow 1}(x_1) \ . \tag{41.127}$$

Once again, consider the BP-BB rule for traversing a leaf node. When traveling to the right as in Fig.41.24 (b), it implies that

$$\underbrace{m_{\alpha\Rightarrow 1}(x_1)}_{\lambda} = \sum_{x_2} P(x_2|x_1) \underbrace{m_{\alpha\Leftarrow 2}(x_2)}_{\pi} \ . \tag{41.128}$$

One can rewrite the left and right hand sides (LHS, RHS) of Eq.(41.128) as follows

$$RHS = \ = \sum_{x_2} P(x_2|x_1)\pi_{\alpha\Leftarrow 2}(x_2) \tag{41.129}$$

$$= \sum_{x_2} P(x_2|x_1)\lambda_{\beta\Rightarrow 2}(x_2) \ , \tag{41.130}$$

and

$$LHS = \lambda_{\alpha\Rightarrow 1}(x_1) \ . \tag{41.131}$$

Therefore,

$$\lambda_{\alpha\Rightarrow 1}(x_1) = \sum_{x_2} P(x_2|x_1)\lambda_{\beta\Rightarrow 2}(x_2) \ . \tag{41.132}$$

Finally, note that Eq.(41.120) becomes

$$\begin{aligned} P(x_2|\epsilon) &= \mathcal{N}(!x_2)m_{\beta\Rightarrow 2}(x_2)m_{\alpha\Rightarrow 2}(x_2) && (41.133)\\ &= \mathcal{N}(!x_2)m_{\alpha\Leftarrow 2}(x_2)m_{\alpha\Rightarrow 2}(x_2) && (41.134)\\ &= \mathcal{N}(!x_2)\pi_{\alpha\Leftarrow 2}(x_2)\lambda_{\alpha\Rightarrow 2}(x_2) && (41.135)\\ &= \mathcal{N}(!x_2)P(x_2|\epsilon^-)P(x_2|\epsilon^+) && (41.136) \end{aligned}$$

and Eq.(41.121) becomes

$$\begin{aligned} P(x_2, x_1) &= \mathcal{N}(!x_2, !x_1)P(x_2|x_1)m_{\alpha\Leftarrow 1}(x_1)m_{\alpha\Leftarrow 2}(x_2) && (41.137)\\ &= \mathcal{N}(!x_2, !x_1)P(x_2|x_1)\pi_{\alpha\Leftarrow 1}(x_1)\pi_{\alpha\Leftarrow 2}(x_2) \ . && (41.138) \end{aligned}$$

41.8.2 BP-BB and general BP agree on tree bnets.

It is instructive to compare the belief values (i.e., $P(x_i|\epsilon)$) obtained by applying the general (i.e., polytree) BP and BP-BB algorithms to a tree bnet. Next we show that both algorithms yield the same belief values.

Applying to the left panel of Fig.41.25 the BP-BB rule for traversing a root node, we get

$$m_{\alpha\Leftarrow\underline{x}}(x) = \prod_i m_{\beta_i\Rightarrow\underline{x}}(x) \ . \tag{41.139}$$

Applying to the left panel of Fig.41.25 the BP-BB rule for traversing a leaf node, we get

$$m_{\alpha\Rightarrow\underline{a}}(a) = \mathcal{N}(!a)\sum_x m_{\alpha\Leftarrow\underline{x}}(x)P(x|a) \ . \tag{41.140}$$

Combining Eqs.(41.139) and (41.140), we get

$$m_{\alpha\Rightarrow\underline{a}}(a) = \mathcal{N}(!a)\sum_x P(x|a)\prod_i m_{\beta_i\Rightarrow\underline{x}}(x) \ , \tag{41.141}$$

Figure 41.25: Subgraph of a tree bnet. This is the same as Fig.41.11, except that here the yellow node has a single parent because this is a subgraph of a tree bnet, not of an arbitrary bnet like Fig.41.11. The subgraph has been converted to a subgraph of a bipartite bnet by inserting a collider leaf node, labeled by a Greek letter, at the center of each edge of the tree bnet. Red arrows indicate the direction of message info flow.

which can be rewritten as

$$\lambda_{\underline{x}\Rightarrow\underline{a}}(a) = \mathcal{N}(!a)\sum_x P(x|a)\underbrace{\prod_i \lambda_{\underline{b}_i\Rightarrow\underline{x}}(x)}_{\lambda_{\underline{x}}(x)} \ . \tag{41.142}$$

Eq.(41.142) is just RULE 1 for general BP.

Applying to the right panel of Fig.41.25 the BP-BB rule for traversing a root node, we get

$$m_{\beta_i\Leftarrow\underline{x}}(x) = \mathcal{N}(!x)m_{\alpha\Rightarrow\underline{x}}(x)\prod_{k\neq i} m_{\beta_k\Rightarrow\underline{x}}(x) \tag{41.143}$$

Applying to the right panel of Fig.41.25 the BP-BB rule for traversing a leaf node, we get

$$\begin{aligned} m_{\alpha\Rightarrow\underline{x}}(x) &= \sum_a P(x|a)m_{\alpha\Leftarrow a}(a) && (41.144)\\ &= \sum_a P(x|a)\pi_{\underline{x}\Leftarrow a}(a) && (41.145)\\ &= \pi_{\underline{x}}(x) \ . && (41.146) \end{aligned}$$

Combining Eqs.(41.143) and (41.146), we get

$$\pi_{\underline{b}_i \Leftarrow \underline{x}}(x) = \mathcal{N}(!x)\pi_{\underline{x}}(x) \prod_{k \neq i} \lambda_{\underline{b}_k \Rightarrow \underline{x}}(x) \ . \tag{41.147}$$

Eq.(41.147) is just RULE 2 of general BP.

41.9 BP-BB and sum-product decomposition

BP-BB yields what is often referred to as a **sum-product decomposition**. I don't like that name because it is unnecessarily confusing, and it fails to convey the recursive nature[6] of the decomposition. I prefer to call it a **recursive sum of products (RSOP) decomposition**, and will call it so henceforth in this chapter.

Expressing the marginals of a bnet as RSOPs, which is what BP does, reduces the complexity of the calculation. (i.e., the total number of additions and multiplications that need to be performed) That makes using the BP algo very advantageous. For instance, consider a Markov chain $\underline{x}_{n-1} \leftarrow \cdots \leftarrow \underline{x}_1 \leftarrow \underline{x}_0$, where $x_i \in \{0, 1, 2\}$ for all i. Note that if we calculate $P(x_{n-1})$ as follows

$$P(x_{n-1}) = \left[\sum_{x_{n-2}} P(x_{n-1}|x_{n-2}) \ldots \left[\sum_{x_1} P(x_2|x_1) \left[\sum_{x_0} P(x_1|x_0)P(x_0) \right] \right] \ldots \right] \ , \tag{41.148}$$

we need to perform $2(n-1)$ additions and $3(n-1)$ multiplications. On the other hand, if we calculate $P(x_{n-1})$ as follows

$$P(x_{n-1}) = \sum_{x_{n-2}} \ldots \sum_{x_1} \sum_{x_0} P(x_{n-1}|x_{n-2}) \ldots P(x_2|x_1)P(x_1|x_0)P(x_0) \ , \tag{41.149}$$

we need to perform $3^n - 1$ additions and $3^n(n-1)$ multiplications.

[6] By "recursive nature", we mean bootstrapped definitions that lead to nested sums. The recursive nature of BP is evident from RULES 1 and 2 that define λ's and π's in terms of other λ's and π's.

Chapter 42

Message Passing (Belief Propagation) in Quantum Mechanics

See Ref.[56].

Chapter 43

Meta-learners for estimating ATE

This section is based on the final 2 chapters of Ref.[8].

The Average Treatment Effect (ATE) is defined in Chapter 52.

Economists are huge fans of Linear Regression (LR), and traditionally calculate ATE using LR. But in recent times, they have begun to calculate ATE using Machine Learning (ML) instead. This chapter describes various methods that economists and others have devised for calculating ATE with ML. These methods are called **meta-learners** because they involve multiple ML or LR steps.

Using ML to calculate ATE captures non-linear trends whose exclusion might sometimes lead to a poor result. On the other hand, ML is more expensive computationally than LR, and it introduces the danger of overfitting, a danger which is nonexistent with LR.

Below, we represent each Linear Regression (LR) step as follows. We list a dataset; i.e., a set of tuples indexed by the individuals σ of a population Σ such that $|\Sigma| = nsam$. The independent variables of the LR (i.e., x^σ) are unboxed and the dependent variable (aka target feature) (i.e., y^σ) is shown inside a box. Then we show an arrow with the superscript "LR-fit", followed by the fit function obtained by performing the LR.[1]

$$\{(\sigma, x^\sigma = [x_i^\sigma], \boxed{y^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{LR}-\text{fit}} \hat{y}(x) = \alpha + x_i\beta_i \tag{43.1}$$

Analogously, below, we represent each Supervised Machine Learning (ML) step as follows.

$$\{(\sigma, x^\sigma, \boxed{y^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{ML}-\text{fit}} \hat{y}(x) \tag{43.2}$$

Henceforth, we will use $\delta(x)(\approx \text{ATE})$ to denote the treatment effect.

- **S (Single)-learner**

$$\{(\sigma, x^\sigma, d^\sigma, \boxed{y^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{ML}-\text{fit}} \hat{y}(d, x) \tag{43.3}$$

[1] In this chapter, we will occasionally use the Einstein summation convention; i.e., implicit sum over repeated indices.

$$\delta(x) = \hat{y}(1, x) - \hat{y}(0, x) \tag{43.4}$$

- **T (Twin)-learner**

$$\{(\sigma, x^\sigma, d^\sigma = 0, \boxed{y^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \hat{y}_0(x) \tag{43.5}$$

$$\{(\sigma, x^\sigma, d^\sigma = 1, \boxed{y^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \hat{y}_1(x) \tag{43.6}$$

$$\delta(x) = \hat{y}_1(x) - \hat{y}_0(x) \tag{43.7}$$

- **X (Cross)-learner**

  do T-learner, get $\hat{y}_0(x)$, $\hat{y}_1(x)$

$$\{(\sigma, x^\sigma, d^\sigma = 0, \boxed{y^\sigma - \hat{y}_1(x^\sigma)}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \mathcal{Y}_0(x) \tag{43.8}$$

$$\{(\sigma, x^\sigma, d^\sigma = 1, \boxed{y^\sigma - \hat{y}_0(x^\sigma)}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \mathcal{Y}_1(x) \tag{43.9}$$

$$\delta(x) = \frac{1}{2}[\mathcal{Y}_1(x) - \mathcal{Y}_0(x)] \tag{43.10}$$

- **De-biased (aka Orthogonal) ML**

  Standard supervised ML is performed with two features, the independent feature and the dependent or target feature. Supervised ML has a target feature. Unsupervised ML doesn't.

  In **De-biased LR**, we do LR with two residuals. Let's call them the **independent residual** and the **dependent or target residual**. These two residuals are calculated with the help of two previously performed LR steps.

  In **De-biased ML**, we do ML or LR (either one) with two residuals. These two residuals are calculated with the help of two previously performed ML steps (instead of two LR steps).

  The FWL theorem discussed in Chapter 22 shows how to do De-biased LR. Next, we will describe how to do De-biased ML.

  We start by doing two ML steps:

$$\{(\sigma, x^\sigma, \boxed{d^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \hat{d}(x) \tag{43.11}$$

$$\{(\sigma, x^\sigma, \boxed{y^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \hat{y}(x) \tag{43.12}$$

After these two ML steps, we do either LR or ML to get $\delta(x)$.
Let

$$\Delta d^\sigma = d^\sigma - \hat{d}(x^\sigma) \tag{43.13}$$

$$\Delta y^\sigma = y^\sigma - \hat{y}(x^\sigma) \tag{43.14}$$

Options for the final learning step that calculates $\delta(x)$:

1. Do a standard LR to get a $\delta(x)$ that is a constant (i.e., x independent):

$$\{(\sigma, \Delta d^\sigma, \boxed{\Delta y^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{LR-fit}} \mathcal{Y}(\Delta d) = \alpha + \Delta d \delta \tag{43.15}$$

$$\delta = \text{coefficient of } \Delta d \text{ in } \mathcal{Y}(\Delta d). \tag{43.16}$$

2. Do a LR with a $x * d$ cross term to get a $\delta(x)$ that is linear in x:

$$\{(\sigma, x^\sigma, \Delta d^\sigma, \boxed{\Delta y^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{LR-fit}} \mathcal{Y}(\Delta d, x) = \alpha_0 + x\beta_0 + \Delta d(\alpha_1 + x\beta_1) \tag{43.17}$$

$$\delta(x) = \mathcal{Y}(\Delta d|_{d=1}, x) - \mathcal{Y}(\Delta d|_{d=0}, x) = \alpha_1 + x\beta_1 \tag{43.18}$$

3. Do ML with a weighted cost function to get a general fit $\hat{\delta}(x)$.
   The cost function used in standard ML (see Eq.(43.2)) is:

$$\mathcal{C} = \frac{1}{nsam} \sum_\sigma [y^\sigma - \hat{y}(x^\sigma)]^2 \tag{43.19}$$

   Define the cost function in this case as

$$\mathcal{C} = \frac{1}{nsam} \sum_\sigma \left[\Delta y^\sigma - \hat{\delta}(x^\sigma)\Delta d^\sigma\right]^2 \tag{43.20}$$

$$= \frac{1}{nsam} \sum_\sigma [\Delta d^\sigma]^2 \left[\frac{\Delta y^\sigma}{\Delta d^\sigma} - \hat{\delta}(x^\sigma)\right]^2 \tag{43.21}$$

   This is a weighted cost function with weights $[\Delta d^\sigma]^2$.

$$\{(\sigma, x^\sigma, \boxed{\frac{\Delta y^\sigma}{\Delta d^\sigma}}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \hat{\delta}(x) \tag{43.22}$$

4. Do ML with Δd^σ as an independent feature to get a general fit $\hat{\delta}(\Delta d, x)$:

$$\{(\sigma, x^\sigma, \Delta d^\sigma, \boxed{\frac{\Delta y^\sigma}{\Delta d^\sigma}}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \hat{\delta}(\Delta d, x) \tag{43.23}$$

$$\delta(x) = \hat{\delta}(\Delta d|_{d=1}, x) \tag{43.24}$$

Chapter 44

Missing Data, Imputation

This chapter assumes that the reader has read some parts of Chapter 21 on the Expectation Maximization (EM) algo and Chapter 38 on Markov Chain Monte Carlo (MCMC).

	h_0	x_0	x_1	x_2
1	NA	0	1	1
2	NA	0	0	0
3	NA	1	1	0
4	NA	NA	1	NA
5	NA	0	NA	1
6	NA	0	0	1

	h_0	x_0	x_1	x_2	m
1	NA	0	1	1	(0,0,0)
2	NA	0	0	0	(0,0,0)
3	NA	1	1	0	(0,0,0)
4	NA	0 0 1 1	1	0 1 0 1	(1,0,1)
5	NA	0	0 1	1	(0,1,0)
6	NA	0	0	1	(0,0,0)

Table 44.1: **Left Table:** Dataset with $nsam = 6$ and some missing entries, for 4 binary variables h_0, x_0, x_1, x_2. NA=not available. The h_0 column is completely missing because h_0 is an unobserved latent variable. **Right Table:** All possibilities for $x_i = NA$ cells of left table have been enumerated. A new column labeled m has been added. $m_i = \mathbb{1}(x_i \text{ is missing})$ for $i = 0, 1, 2$.

Suppose that you have compiled a **dataset** $\vec{x} = (x[\sigma])_{\sigma=0,1,\ldots,nsam-1}$ where $x = (x_0, x_1, \ldots, x_{nx-1})$ from a study or survey. It consists of $nsam$ number of samples (sample= row), and nx columns (each column is a different feature, or observation). Suppose that some of the cells in this matrix are empty. Throwing away all the incomplete rows is okay if the number of incomplete rows is much smaller than $nsam$. If not, throwing them away would throw away a substantial amount of information contained in all the filled cells in those incomplete rows, plus it might bias your dataset. This chapter deals with how to fill those empty cells with plausible fake data. A fancy name for this process is **imputation**. There is no unique way of

fabricating fake data, but some fakes are better than others by some metrics. This chapter will consider two popular ways (EM and MCMC) of filling those empty cells with their "most likely" values based on the cells of the dataset that aren't missing, and also based on some bnet model that is expected to describe well the dataset.

Notation: $\underline{\vec{a}} = (\underline{a}[\sigma])_{\sigma=0,1,\ldots,nsam-1}$, where $nsam$ is the number of samples. Will sometimes denote $a[\sigma]$ by $a^{[\sigma]}$.

For concreteness, we will apply the concepts of this chapter to the dataset with missing data given by Table 44.1.

44.1 Imputation via EM

We begin by augmenting Fig.21.1 (the first figure in Chapter 21) by adding to it a new node $\underline{\vec{m}}$ called the **missingness variable**. Recall that node $\underline{\theta}$ represents the **unknown parameters**, node $\underline{\vec{x}}$ represents the **observed variables**, and node $\underline{\vec{h}}$ represents the **latent variables**. Both $\underline{\theta}$ and $\underline{\vec{h}}$ are hidden (i.e., unobserved). Fig.44.1 shows 3 popular ways of connecting node $\underline{\vec{m}}$ to the other nodes in the graph Fig.21.1.

Figure 44.1: The left bnet is seldom assumed. The middle bnet is referred to as the MAR (missing at random) assumption. The right bnet is referred to as the not-MAR (NMAR) assumption.

From Fig.44.1, we have

$$P(\vec{m}|\vec{x},\vec{h},\theta) = \begin{cases} P(\vec{m}|\theta) & \text{Seldom assumed. Called missing-CAR (MCAR)} \\ P(\vec{m}|\vec{x},\theta) & \text{MAR} \\ P(\vec{m}|\vec{x},\vec{h},\theta) & \text{not-MAR (NMAR)} \end{cases} . \tag{44.1}$$

For doing imputation via EM, we connect node $\underline{\vec{m}}$ as shown in the middle bnet (called MAR) of Fig.44.1.

For the example of Table 44.1, we have variables $\underline{\vec{m}}, \underline{\vec{x}}$ and $\underline{\vec{h}}$ whose values range over the following sets:

$\underline{\vec{x}} = (\underline{\vec{x}}_0, \underline{\vec{x}}_1, \underline{\vec{x}}_2)$
$\underline{\vec{h}} = (\underline{\vec{h}}_0)$
$\underline{h}_0[\sigma] \in \{0,1\}$,
$\underline{x}_i[\sigma] \in \{0,1\}$ for $i = 0,1,2$,

Figure 44.2: MAR bnet with $nsam = 3$.

Figure 44.3: Our example for imputation via EM assumes this bnet between nodes $\underline{m}[\sigma], \underline{x}[\sigma], \underline{h}[\sigma]$.

$\underline{m}_i[\sigma] \in \{0, 1\}$ for $i = 0, 1, 2$.

For concreteness, we will assume that the Markov chain $\underline{m}[\sigma] \leftarrow \underline{x}[\sigma] \leftarrow \underline{h}[\sigma]$ has a finer grained DAG structure given by Fig.44.3. where we will omit the dashed arrows. If one doesn't want to assume that the data can be fitted well by the bnet of Fig.44.3 without the dashed arrows, one can include those arrows too, at the expense of more unknown parameters (i.e., degrees of freedom) to be lumped into θ. We will parameterize the TPMs corresponding to Fig.44.3 using a Categorical Distribution for each column of the TPMs. We will thus assume that the bnet of Fig.44.3 has the following TPMs, printed in blue.

$$P(h_0^{[\sigma]}|\theta) = \begin{array}{c|c} & \\ \hline & 1-\theta_0 \\ 1 & \theta_0 \end{array} \tag{44.2}$$

$$P(x_0^{[\sigma]}|\theta) = \begin{array}{c|c} & \\ \hline 0 & 1-\theta_1 \\ 1 & \theta_1 \end{array} \tag{44.3}$$

$$P(x_1^{[\sigma]} \mid x_0^{[\sigma]}, h^{[\sigma]}, \theta) = \begin{array}{c|cccc} & 00 & 01 & 10 & 11 \\ \hline 0 & 1-\theta_2 & 1-\theta_3 & 1-\theta_4 & 1-\theta_5 \\ 1 & \theta_2 & \theta_3 & \theta_4 & \theta_5 \end{array} \tag{44.4}$$

$$P(x_2^{[\sigma]} \mid x_1^{[\sigma]}, x_0^{[\sigma]}, \theta) = \begin{array}{c|cccc} & 00 & 01 & 10 & 11 \\ \hline 0 & 1-\theta_6 & 1-\theta_7 & 1-\theta_8 & 1-\theta_9 \\ 1 & \theta_6 & \theta_7 & \theta_8 & \theta_9 \end{array} \tag{44.5}$$

$$P(m^{[\sigma]}|x^{[\sigma]}, \theta) = \frac{1}{nsam} P((x_i)_{\forall i \ni m_i=1} \mid (x_i)_{\forall i \ni m_i=0}, \theta) \tag{44.6}$$

Eq.(44.6) can be illustrated as follows. In Table 44.2, we added a $P(m)$ column to Table 44.1.

	h_0	x_0	x_1	x_2	m	$P(m)$
1	NA	0	1	1	(0,0,0)	$\frac{1}{nsam}$
2	NA	0	0	0	(0,0,0)	$\frac{1}{nsam}$
3	NA	1	1	0	(0,0,0)	$\frac{1}{nsam}$
4	NA	0 0 1 1	1	0 1 0 1	(1,0,1)	$\frac{1}{nsam}P(x_0=0, x_2=0 \mid x_1=1, \theta)$ $\frac{1}{nsam}P(x_0=0, x_2=1 \mid x_1=1, \theta)$ $\frac{1}{nsam}P(x_0=1, x_2=0 \mid x_1=1, \theta)$ $\frac{1}{nsam}P(x_0=1, x_2=1 \mid x_1=1, \theta)$
5	NA	0	0 1	1	(0,1,0)	$\frac{1}{nsam}P(x_1=0 \mid x_0=0, x_2=1, \theta)$ $\frac{1}{nsam}P(x_1=1 \mid x_0=0, x_2=1, \theta)$
6	NA	0	0	1	(0,0,0)	$\frac{1}{nsam}$

Table 44.2: $P(m)$ column added to Table 44.1. Note that $\sum_m P(m) = 1$.

$$\theta = (\theta_i)_{i=0,1,\ldots,9} \tag{44.7}$$

$$P(m^{[\sigma]}, x^{[\sigma]}, h^{[\sigma]}|\theta) = P(m^{[\sigma]}|x^{[\sigma]}, \theta) P(x^{[\sigma]}|h^{[\sigma]}, \theta) P(h^{[\sigma]}|\theta) \tag{44.8}$$

$$P(x^{[\sigma]}|h^{[\sigma]}, \theta) = P(x_2^{[\sigma]}|x_1^{[\sigma]}, x_0^{[\sigma]}, \theta) P(x_1^{[\sigma]}|x_0^{[\sigma]}, h^{[\sigma]}, \theta) P(x_0^{[\sigma]}|\theta) \tag{44.9}$$

$$P(x_1^{[\sigma]}|x_0^{[\sigma]}, \theta) = \sum_h P(x_1^{[\sigma]}|x_0^{[\sigma]}, h^{[\sigma]}, \theta) P(h^{[\sigma]}|\theta) \tag{44.10}$$

$$P(x^{[\sigma]}|\theta) = P(x_2^{[\sigma]}|x_1^{[\sigma]}, x_0^{[\sigma]}, \theta) P(x_1^{[\sigma]}|x_0^{[\sigma]}, \theta) P(x_0^{[\sigma]}|\theta) \tag{44.11}$$

$$Q(\theta|\theta^{(t)}) = \sum_{\vec{m},\vec{h}} P(\vec{m},\vec{h} \mid \vec{x},\theta^{(t)}) \ln P(\vec{m},\vec{x},\vec{h}|\theta) \tag{44.12}$$

$$= \sum_{\vec{m},\vec{h}} \left[\prod_{\sigma} P(m^{[\sigma]},h^{[\sigma]} \mid x^{[\sigma]},\theta^{(t)})\right] \ln \left[\prod_{\sigma} P(m^{[\sigma]},x^{[\sigma]},h^{[\sigma]}|\theta)\right] \tag{44.13}$$

$$= \sum_{\sigma} \sum_{m^{[\sigma]},h^{[\sigma]}} P(m^{[\sigma]},h^{[\sigma]} \mid x^{[\sigma]},\theta^{(t)}) \ln P(m^{[\sigma]},x^{[\sigma]},h^{[\sigma]}|\theta) \tag{44.14}$$

$$= \sum_{\sigma} \sum_{m^{[\sigma]},h^{[\sigma]}} \frac{P(m^{[\sigma]},h^{[\sigma]},x^{[\sigma]} \mid \theta^{(t)})}{P(x^{[\sigma]} \mid \theta^{(t)})} \ln P(m^{[\sigma]},x^{[\sigma]},h^{[\sigma]}|\theta) \tag{44.15}$$

Once you find optimal parameters θ^* by recursing this $Q(\theta|\theta^{(t)})$, you can evaluate numerically the $P(m)$ column of Table 44.2. In Table 44.2, out of the 4 sub-rows for row 4, choose the one with the highest probability. Similarly, out of the 2 sub-rows for row 5, choose the one with the highest probability.

44.2 Imputation via MCMC

A simple and popular way to do imputation via MCMC is described in Ref.[51]. It goes as follows.

Let

$$\underline{H}[\sigma] = (\underline{h}[\sigma],\underline{m}[\sigma]) \tag{44.16}$$

for $\sigma = 0,1,\ldots,nsam-1$. Initialize $\theta^{(0)}$ to a random value within the allowed ranges. Do the following 2 steps, for $t = 0,1,\ldots,T-1$, where T is large enough that $\theta^{(t)}$ has reached a steady value that is independent of $\theta^{(0)}$. To do the sampling, use a standard sampling technique such as Gibbs sampling.

- **STEP 1:** For $\sigma = 0,1,\ldots,nsam-1$, find a sample

$$(H^{[\sigma]})^{(t+1)} \sim P(H^{[\sigma]}|x^{[\sigma]},\theta^{(t)}) \ . \tag{44.17a}$$

- **STEP 2:** Find a sample

$$\theta^{(t+1)} \sim P^{(t+1)}(\theta) \tag{44.17b}$$

  where

$$P^{(t+1)}(\theta) = \mathcal{N}(!\theta)P(\vec{x},\vec{H}^{(t+1)}|\theta) \tag{44.17c}$$

$$= \mathcal{N}(!\theta)\prod_{\sigma} P(x^{[\sigma]},(H^{[\sigma]})^{(t+1)}|\theta) \ . \tag{44.17d}$$

Fig.44.4 illustrates this two step recursive process using a bnet.

Figure 44.4: bnet illustrating Eqs.(44.17) for doing imputation via MCMC. The **same** node $\underline{\vec{x}}$ appears twice to make the graph clearer.

44.3 Multiple Imputations

Multiple imputations means calculating θ^* (i.e., the optimum θ) and the concomitant dataset $\vec{x}^*, \vec{H}^*$, via any method (such as EM or MCMC), a large number of times, starting from different, randomly chosen $\theta^{(0)}$ initial parameters. Then calculating the average and the variance of $\theta^*, \vec{x}^*, \vec{H}^*$ and functions thereof.

Chapter 45

Monty Hall Problem

Figure 45.1: Monty Hall Problem.

Mr. Monty Hall, host of the game show "Lets Make a Deal", hides a car behind one of three doors and a goat behind each of the other two. The contestant picks Door No. 1, but before opening it, Mr. Hall opens Door No. 2 to reveal a goat. Should the contestant stick with No. 1 or switch to No. 3?

The Monty Hall problem can be modeled by the bnet Fig.45.1, where

- $\underline{c}$= the door behind which the car actually is.
- $\underline{y}$= the door opened by you (the contestant), on your first selection.
- $\underline{m}$= the door opened by Monty (game host)

We label the doors 1,2,3 so $S_{\underline{c}} = S_{\underline{y}} = S_{\underline{m}} = \{1, 2, 3\}$.
Node matrices printed in blue:

$$P(c) = \frac{1}{3} \text{ for all } c \tag{45.1}$$

$$P(y) = \frac{1}{3} \text{ for all } y \tag{45.2}$$

$$P(m|c, y) = \mathbb{1}(m \neq c) \left[\frac{1}{2} \mathbb{1}(y = c) + \mathbb{1}(y \neq c) \mathbb{1}(m \neq y) \right] \tag{45.3}$$

It's easy to show that the above node probabilities imply that

$$P(c = 1|m = 2, y = 1) = \frac{1}{3} \tag{45.4}$$

$$P(c = 3|m = 2, y = 1) = \frac{2}{3} \tag{45.5}$$

So you are twice as likely to win if you switch your final selection to be the door which is neither your first choice nor Monty's choice.

The way I justify this to myself is: Monty gives you a piece of information. If you don't switch your choice, you are wasting that info, whereas if you switch, you are using the info.

Chapter 46

Multi-armed Bandits

Figure 46.1: Multi-armed bandit (MAB).

Figure 46.2: Bernoulli MAB (Bern-MAB) with 4 arms. For a Bern-MAB, the conditional probability distribution $P(r|a)$ for reward r, given arm a, is initially totally unknown to the gambler (agent), but it is known by the environment to be a Bernoulli distribution $P(r|a) = \mu_a^r(1-\mu_a)^{1-r}$ for $r \in \{0, 1\}$, where $0 < \mu_a < 1$ for each a. The percentages shown are μ_a for each arm a.

This chapter is mostly based on Refs. [42] and [59].

Multi-armed Bandits (MABs) are a simple version of Reinforcement Learning (RL). RL is discussed in Chapter 58.

The term "one-armed-bandit" is a humorous term for what is also called a slot machine. A slot machine is a gambling device which has a slot into which you put coins or tokens for the privilege of being allowed to pull down a lever (arm) on one side of the device. This action generates a random combination of three shapes, which may or may not, depending on their combination, entitle the player to a money award.

Multi-armed bandit (MAB) is the name given to the optimization problem that considers an agent (gambler) that is playing multiple one-armed-bandits, each with a possibly different odds of winning. The optimization problem is to determine an efficient schedule whereby the gambler can converge on the device with the highest odds of winning.

MABs are often used in marketing as an alternative to A/B testing. These 2 methods yield different information but overlap in that they both can discover consumer preferences.

The MAB problem is an optimization problem (i.e., finding the maximum of a reward function or the minimum of a cost function). As with any minimization problem, an algorithm to solve it runs the danger of converging to a local minimum that isn't the global (i.e., the overall) minimum. This danger can be diminished by doing both **exploration and exploitation**. Algorithms that do no exploration, only exploitation, are said to be **greedy**, and they are at the highest risk of converging to a non-global minimum.

46.1 Bnet for MAB

Figure 46.3: Bnet for a multi-armed bandit (MAB).

Let
$t \in \{0, 1, 2, \ldots\}$ be the **time** slice (step),

$\underline{a}_t \in \{0, 1, \ldots, N_{\underline{a}} - 1\} = S_{\underline{a}}$ be the **arm** of bandit that is pulled at time t, out of $N_{\underline{a}}$ arms. In RL language, it's also the **action** taken.

$\underline{r}_t \in \mathbb{R}$ be the **reward** at time t,

$s_{<t} = [\underbrace{(a_\tau, r_\tau)}_{s_\tau} : \tau < t]$ be the **state** at time t.

Fig.46.3 shows a bnet for a MAB. The TPMs, printed in blue, for this bnet, are as follows.

$$P(s_{<t}|s_{<t-1}, a_{t-1}, r_{t-1})) = \mathbb{1}(s_{<t} = s_{<t-1} \cup (a_{t-1}, r_{t-1})) \tag{46.1}$$

For $t = 0$, $s_{<0} = \emptyset$, so choose a random a_0.

$$P(a_t|s_{<t}) = \mathbb{1}(a_t = a_t^*) \ \text{ (agent's response)} \ , \tag{46.2}$$

where a_t^* depends on the **strategy (aka policy)** being used by the **gambler (agent)**. We consider various strategies below. a_t^* is defined solely in terms of empirical data (i.e., a_t, r_t values) collected from previous time slices. That is the only type of information that the gambler is privy to. a_0^* is chosen at random. The formulae for a_t^* that we give below for various strategies should only be used for $t > 0$.

$$P(r_t|a_t) = P_{\underline{r}|\underline{a}}(r_t|a_t) \ \text{ (environment's response)} \ . \tag{46.3}$$

$P_{\underline{r}|\underline{a}}$ is a probabilistic model that models the **environment** in which the agent lives. This assumes that the a_t (and the r_t) are i.i.d. $P_{\underline{r}|\underline{a}}$ depends on parameters whose values are known by the environment, but are not known a priori by the gambler. In fact, the goal of this exercise is for the gambler to find ever more accurate estimates of those parameters, using only the empirical data he/she can collect from the past, starting from total ignorance about those parameters.

For a **Bernoulli MAB (Bern-MAB)**, the conditional probability distribution $P_{\underline{r}|\underline{a}}$ is a Bernoulli distribution

$$P(r|a) = \mu_a^r (1 - \mu_a)^{1-r} \tag{46.4}$$

for $r \in \{0, 1\}$, where $0 < \mu_a < 1$ for each a. The parameters μ_a are initially unknown to the gambler. Note that $E[\underline{r}|a] = \sum_r r P(r|a) = P(\underline{r} = 1|a) = \mu_a$.

For a **Gaussian MAB**, the conditional probability distribution $P_{\underline{r}|\underline{a}}$ is a Normal (Gaussian) distribution

$$P(r|a) = \mathcal{N}(r; \mu_a, \sigma_a^2) \ . \tag{46.5}$$

for $r \in \mathbb{R}$. The parameters μ_a, σ_a^2 are initially unknown to the gambler.

46.2 Reward functions

For $a \in S_{\underline{a}}$, define the **Long term Average Reward** for action a by

$$\mu_a = Q(a) = E_{|a}[\underline{r}] = \sum_r r P(r|a) \ . \tag{46.6}$$

Let

$$a^* = \operatorname*{argmax}_a Q(a) \tag{46.7}$$

$$\mu^* = \max_a Q(a) = Q(a^*) \tag{46.8}$$

$$\Delta_a = \mu^* - Q(a) \tag{46.9}$$

μ_a, a^*, μ^* and Δ_a are not known to the gambler.

Define the **Instantaneous (at time t) Average Reward** for action a by

$$Q_t(a) = \frac{1}{N_t(a)} \sum_{\tau=0}^{t} r_\tau \mathbb{1}(a_\tau = a) \tag{46.10a}$$

where

$$N_t(a) = N_{in} + \sum_{\tau=0}^{t} \mathbb{1}(a_\tau = a) \tag{46.10b}$$

$N_{in} > 0$ insures that we never divide by zero. Note that Eqs.(46.10) can be stated recursively as

$$N_t(a)Q_t(a) = N_{t-1}(a)Q_{t-1}(a) + r_t \mathbb{1}(a_t = a) \tag{46.11a}$$

$$N_t(a) = N_{t-1}(a) + \mathbb{1}(a_t = a) \tag{46.11b}$$

with $Q_{-1}(a) = 0$ and $N_{-1}(a) = N_{in}$ for all a. We assume that at large t, $N_{in} << N_t(a)$ for all a. For instance, one can use $N_{in} = 1$.

$Q(a)$ is not known to the gambler but $N_t(a)$ and $Q_t(a)$ are because they are empirical.

We will write a hat over random variables that are defined by an empirical probability distribution.[1] Whereas we assume that $\underline{a}_t$ and $\underline{r}_t$ are i.i.d., we will not assume that $\hat{a}_t$ and $\hat{r}_t$ are i.i.d. for finite times t. What we will assume is that as $t \to \infty$,

$$\underline{\hat{a}}_t \to \underline{a} \ , \ \underline{\hat{r}}_t \to \underline{r} \tag{46.12}$$

[1] An alternative convention is to not distinguish between $\underline{\hat{a}}_t$ and $\underline{a}_t$, or between $\underline{\hat{r}}_t$ and $\underline{r}_t$, but to distinguish between $P(r_t|a_t)$ and $\hat{P}(r_t|a_t)$, where $\hat{P}()$ is an empirical estimate of $P()$.

Note that

$$\sum_a \frac{N_t(a)}{t+1} = 1 \tag{46.13}$$

so define

$$P(\underline{\hat{a}}_t = a) = \frac{N_t(a)}{t+1} \ . \tag{46.14}$$

Thus

$$E[Q(\underline{\hat{a}}_t)] = \sum_a P(\underline{\hat{a}}_t = a) Q(a) \ . \tag{46.15}$$

Claim 52

$$Q_t(a) = E_{|\hat{a}_t = a}[\underline{\hat{r}}_t] \tag{46.16}$$

proof:

Note that

$$\sum_r \sum_{\tau=0}^{t} \frac{\mathbb{1}(a_\tau = a, r_\tau = r)}{N_t(a)} = 1 \ . \tag{46.17}$$

so define

$$P(\underline{\hat{r}}_t = r | \underline{\hat{a}}_t = a) = \sum_{\tau=0}^{t} \frac{\mathbb{1}(a_\tau = a, r_\tau = r)}{N_t(a)} \ . \tag{46.18}$$

Thus

$$\begin{aligned} Q_t(a) &= \sum_r r \sum_{\tau=0}^{t} \frac{\mathbb{1}(a_\tau = a, r_\tau = r)}{N_t(a)} && (46.19) \\ &= \sum_r r P(\underline{\hat{r}}_t = r | \underline{\hat{a}}_t = a) && (46.20) \\ &= E_{|\hat{a}_t = a}[\underline{\hat{r}}_t] && (46.21) \end{aligned}$$

QED

Claim 53 *As $t \to \infty$,*

$$Q_t(a) \to E_{\underline{a}=a}[\underline{r}] = Q(a) \tag{46.22}$$

$$E[Q(\underline{\hat{a}}_t)] \to E[Q(\underline{a})] \tag{46.23}$$

proof: This is clear from $\hat{a}_t \to a$ and $\hat{r}_t \to r$ and Claim 52.
QED

46.3 Regret functions

Define the **Instantaneous (at time t) Average Regret (I-Regret)** by

$$IReg_t = \mu^* - E[Q(\underline{\hat{a}}_t)] = E[\Delta_{\underline{\hat{a}}_t}] \tag{46.24}$$

and the **Cumulative (for times $\leq t$) Average Regret (C-Regret)** by

$$CReg_t = \sum_{\tau=0}^{t} IReg_\tau \ . \tag{46.25}$$

Note that

$$\begin{aligned} CReg_t &= \sum_{\tau=0}^{t} E[\Delta_{\underline{\hat{a}}_\tau}] && (46.26) \\ &= \sum_a \Delta_a \sum_{\tau=0}^{t} \frac{N_\tau(a)}{\tau+1} && (46.27) \end{aligned}$$

Let

$$\rho_t = \frac{1}{t+1} \sum_{t=0}^{t} r_t \ . \tag{46.28}$$

Define the **Cumulative Average Reward (C-Reward)** by

$$CRew_t = E_{|\underline{\hat{a}}_t = a}[(t+1)\underline{\hat{\rho}}_t] \ . \tag{46.29}$$

It can be shown that minimizing the C-Regret is equivalent to maximizing the C-Reward.

Claim 54 *As $t \to \infty$,*

$$IReg_t \to E[\Delta_{\underline{a}}] \tag{46.30}$$

proof: This is clear from $\hat{a}_t \to a$ and $\hat{r}_t \to r$.
QED

46.4 Strategies with random exploration

In this section, we consider MAB algorithms that explore all values of the action a at random. In the section following this one, we consider MAB algorithm that do a more deliberate search of the action space.

46.4.1 ϵ-greedy algorithm

Recall that $\underline{a}_t \in \{0, 1, \ldots, N_{\underline{a}} - 1\} = S_{\underline{a}}$. The user of the algorithm specifies an $\epsilon \in [0, 1]$ which measures the amount of exploration to be conducted.

Figure 46.4: Extra structure added to MAB Bnet Fig.46.3 for ϵ-greedy algorithm.

For each t, add extra the structure shown in Fig.46.4 to the MAB bnet Fig.46.3. $\underline{g}_t \in \{0, 1\}$ and $\underline{a}_t, \underline{A}_t \in S_{\underline{a}}$. ("g" stands for greedy). The TPMs, printed in blue, for the new nodes, are as follows

$$P(g_t) = \begin{cases} \epsilon \ \text{(exploration)} & \text{if } g_t = 0 \\ 1 - \epsilon \ \text{(exploitation)} & \text{if } g_t = 1 \end{cases} . \tag{46.31}$$

$$P(A_t|g_t) = \begin{cases} \frac{1}{N_{\underline{a}}} \ \text{(exploration)} & \text{if } g_t = 0 \\ \mathbb{1}(A_t = 0) \ \text{(exploitation)} & \text{if } g_t = 1 \end{cases} \tag{46.32}$$

$$P(a_t|A_t, g_t) = \begin{cases} \mathbb{1}(a_t = A_t) \ \text{(exploration)} & \text{if } g_t = 0 \\ \mathbb{1}(a_t = a_t^*) \ \text{(exploitation)} & \text{if } g_t = 1 \end{cases} \tag{46.33}$$

where a_t^* is defined as follows:

$$\boxed{a_t^* = \operatorname*{argmax}_a Q_{t-1}(a)} . \tag{46.34}$$

As $t \to \infty$,

$$\begin{aligned} \frac{N_t(a)}{t+1} &= P(\hat{\underline{a}}_t = a) && (46.35) \\ &\to P(a, g_t = 0)\epsilon + P(a, g_t = 1)(1 - \epsilon) && (46.36) \\ &\geq P(a, g_t = 0)\epsilon && (46.37) \\ &= \frac{\epsilon}{N_{\underline{a}}} && (46.38) \end{aligned}$$

Hence

$$IReg_t = \sum_a \Delta_a \sum_{\tau=0}^{t} \frac{N_\tau(a)}{\tau+1} \tag{46.39}$$

$$\geq \frac{\epsilon}{N_{\underline{a}}} \sum_a \Delta_a \tag{46.40}$$

and $CReg_t \geq \mathcal{N}(!t)(t+1)$.

46.4.2 ϵ_t-greedy algorithm

Replace time-independent constant ϵ in the ϵ-greedy algorithm by a time dependent function ϵ_t.

46.5 Strategies with nonrandom exploration

46.5.1 Upper Confidence Bounds (UCB) algorithms

A MAB algorithm that maximizes merely $Q_t(a)$ to get a_t^* is totally greedy (i.e., does no exploration, only exploitation). This doesn't work too well because once the algo finds a particular a_t^*, it sticks with it. For times t after that, the $Q_t(a)$ for all a stay more or less the same. The $N_t(a)$ for $a \neq a_t^*$ also stay the same. Only $N_t(a_t^*)$ increases. To avoid this problem, **Upper Confidence Bounds (UCB) algorithms** maximize an effective $Q_t(a)_{\text{eff}} = Q_t(a) + U_t(a)$, where $U_t(a) > 0$, instead of maximizing merely $Q_t(a)$ to get a_t^*. $U_t(a)$ is proportional to $1/\sqrt{N_t(a)}$ (that's a property of UCBs). Adding $U_t(a)$ to $Q_t(a)$ gives those a with low $N_a(a)$ a $Q_t(a)_{\text{eff}} >> Q_t(a)$. This encourages exploration of a different from $a_t^* = \operatorname{argmax} Q_t(a)$. In conclusion, for all UCB algorithms, we have

$$a_t^* = \operatorname*{argmax}_a [Q_{t-1}(a) + U_{t-1}(a)] \tag{46.41}$$

Different UCB algos differ only in the definition of $U_t(a)$.

Next, we consider two UCB algorithms, frequentist UCB (UCB1), and Bayesian UCB.

Frequentist UCB (UCB1) algorithm

Claim 55 *(Hoeffdings Inequality- HI) Let $\underline{x}_0, \underline{x}_1, \ldots, \underline{x}_{T-1}$ be i.i.d. random variables such that $0 \leq \underline{x}_t \leq 1$ for all t. Let $\underline{m}_{T-1} = \frac{1}{T}\sum_{t=0}^{T-1} \underline{x}_t$ denote the sample mean. Then for $u > 0$, we have:*

$$P\left(E[\underline{x}] > \underline{m}_{T-1} + u\right) \leq e^{-2Tu^2} . \tag{46.42}$$

proof: See Ref.[89].
QED

Let[2]

$$\underline{x}^T = \left\{ \underline{x}_t : t \in \mathbb{Z}_{[0,T-1]} \right\} \tag{46.43}$$

and

$$\underline{r}_{\leq t-1}(a) = \left\{ \underline{r}_\tau : \underline{a}_\tau = a, \tau \in \mathbb{Z}_{[0,t-1]} \right\} . \tag{46.44}$$

Note that $\underline{x}^T$ has T components and $\underline{r}_{\leq t-1}(a)$ has $N_{t-1}(a)$ components. If we apply the HI with $\underline{x}^T$ replaced by $\underline{r}_{\leq t-1}(a)$, we get

$$P\left(Q(a) > Q_{t-1}(a) + U_{t-1}(a)\right) \quad \leq \quad e^{-2N_{t-1}(a)[U_{t-1}(a)]^2} \tag{46.45}$$

If we define a threshold probability p by

$$p = e^{-2N_{t-1}(a)[U_{t-1}(a)]^2} \ , \tag{46.46}$$

then

$$U_{t-1}(a) = \sqrt{\frac{-\ln p}{2N_{t-1}(a)}} \tag{46.47}$$

If we choose $p = (t-1)^{-\alpha}$,

$$\boxed{a_t^* = \underset{a}{\operatorname{argmax}} \left[Q_{t-1}(a) + \sqrt{\frac{\alpha(t-1)}{2N_{t-1}(a)}} \ \right]} \tag{46.48}$$

Bayesian UCB algorithm

Claim 56 [3] [4] *(Bayesian updating of mean and deviation. See Fig.46.5.)*
Suppose

$$x^n = [x_i] \ , \quad x_i \ \textit{are i.i.d. with} \ \underline{x_i}|\mu, \tau \sim \mathcal{N}(\mu, \tau) \tag{46.49}$$

$$\underline{\mu}|\tau \sim \mathcal{N}(\mu_0, n_0\tau) \tag{46.50}$$

$$\underline{\tau} \sim \text{Gamma}(\alpha, \beta) \ . \tag{46.51}$$

[2] As usual, we define $\mathbb{Z}_{[a,b]} = \{a, a+1, a+2, \ldots, b\}$ for $a < b$.
[3] For a **standard deviation** σ, the **precision** τ is defined as $\tau = \frac{1}{\sigma^2}$.
[4] $\underline{x}|\lambda \sim \mathcal{D}(\lambda)$ means that $P(x|\lambda) = \mathcal{D}(x;\lambda)$. $\mathcal{N}()$ stands for the Normal distribution and Gamma() for the Gamma distribution. See Ref.[83] for a discussion of the Gamma distribution.

Then the posterior is

$$\underline{\mu}|\tau, x^n \sim \mathcal{N}\left(\frac{n\overline{x} + n_0\mu_0}{n + n_0} \;\;,\;\; (n + n_0)\tau\right) \tag{46.52}$$

$$\underline{\tau}|x^n \sim \text{Gamma}\left(\alpha + \frac{n}{2} \;\;,\;\; \beta + \frac{1}{2}\sum_i (x_i - \overline{x})^2 + \frac{nn_0}{2(n + n_0)}(\overline{x} - \mu_0)^2\right) \tag{46.53}$$

proof: See Ref.[17].
QED

Figure 46.5: Prior and Posterior Bnets in Claim 56.

In the Bayesian UCB algorithm, we use

$$\boxed{a_t^* = \underset{a}{\text{argmax}}\, E_{\mu_a,\sigma_a|s_{<t}}\left[\mu_a + c\frac{\sigma_a}{\sqrt{N_{t-1}(a)}}\right]} \tag{46.54}$$

for some $c > 0$.

Eq.46.54 requires that we know $P(\mu_a, \sigma_a|s_{<t})$; i.e., the posterior distribution of μ_a, σ_a assuming the prior history $s_{<t}$. This follows from Bayes theorem if we assume a Normal distribution for the likelihood $P(s_{<t}|\mu_a, \sigma_a)$ and we assume conjugate priors for $P(\mu_a, \sigma_a)$. More precisely, if we replace $\underline{x}^n$ by $\underline{s}_{<t}$ in Claim 56, then

$$P(\mu_a, \tau_a|s_{<t}) = P(\mu_a|\tau_a, s_{<t})P(\tau_a|s_{<t}) \tag{46.55}$$

where $P(\mu_a|\tau_a, s_{<t})$ and $P(\tau_a|s_{<t})$ are given by Claim 56.

46.5.2 Thompson Sampling MAB (TS-MAB) algorithm

Bnet for general TS-MAB algorithm

The Thompson Sampling MAB (TS-MAB) algorithm is described by the bnet Fig.46.6. This bnet differs from the bnet Fig.46.3 in that it includes new nodes $\underline{\lambda}_t$. The TPMs,

Figure 46.6: Bnet for TS-MAB algorithm.

printed in blue, for the nodes in the bnet Fig.46.6, are as follows.

$$P(s_{<t}|s_{<t-1}, a_{t-1}, r_{t-1}) = \text{ same as for Fig.46.3} \tag{46.56}$$

$$P(\lambda_t|s_{<t}, \lambda_{t-1}) = \mathbb{1}(\lambda_t = \lambda_t^*(s_{<t}, \lambda_{t-1})) \tag{46.57}$$

where λ_t^* is a function to be defined below.

$$P(a_t|\lambda_t) = \mathbb{1}[a_t = a_t(\lambda_t)] \text{ (Agent's response)} \tag{46.58}$$

where $a_t(\lambda_t)$ is a function to be described below.

$$P(r_t|a_t) = \text{ same as for Fig.46.3. (Environment's response)} \tag{46.59}$$

Let

$$Q_t(a, \lambda_t) = \sum_r r P(\hat{\underline{r}}_t = r|\hat{\underline{a}}_t = a, \lambda_t) \tag{46.60}$$

$$= E_{|\hat{\underline{a}}_t = a, \lambda_t}[\hat{\underline{r}}_t] \ . \tag{46.61}$$

Define

$$\boxed{a_t(\lambda_t) = \underset{a}{\operatorname{argmax}}\, Q_t(a, \lambda_t)} \ . \tag{46.62}$$

Note that

$$
\begin{aligned}
P(a_t|s_{<t}) &= \sum_{\lambda_t} P(\lambda_t|s_{<t})\mathbb{1}[Q_t(a_t,\lambda_t) = \max_a Q_t(a,\lambda_t)] && (46.63a)\\
&= \sum_{\lambda_t} P(\lambda_t|s_{<t})\mathbb{1}[a_t = a_t(\lambda_t)] && (46.63b)\\
&= E_{\lambda_t|s_{<t}}\{\mathbb{1}[a_t = a_t(\lambda_t)]\} \ . && (46.63c)
\end{aligned}
$$

If we further assume that $P(\lambda_t|s_{<t})$ is a delta function, then Eqs.(46.63) reduce to

$$
P(a_t|s_{<t}) = \mathbb{1}[a_t = a_t(\lambda_t^*)] \tag{46.64}
$$

TS-MAB algorithm with Beta agent and Bernoulli environment

The Beta distribution $\mathrm{Beta}(x;\alpha,\beta)$ (see Ref.[67]) is defined for $\alpha > 0, \beta > 0$ and $x \in [0,1]$. Since $x \in [0,1]$, x can be interpreted as a probability. The mean and variance of the Beta distribution are

$$
E[\underline{x}] = \frac{\alpha}{\alpha+\beta} \ , \tag{46.65}
$$

$$
\langle \underline{x}, \underline{x} \rangle = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} \ . \tag{46.66}
$$

From this mean and variance, we see that if we increase α by one and leave β the same, the mean moves towards 1 and the variance decreases. Likewise, if we increase β by 1 and leave α the same, the mean moves towards 0 and the variance decreases.

Let

$$
\alpha_t^a = \sum_{\tau=0}^{t} \mathbb{1}(r_\tau = 1, a_\tau = a) \tag{46.67}
$$

$$
\beta_t^a = \sum_{\tau=0}^{t} \mathbb{1}(r_\tau = 0, a_\tau = a) \tag{46.68}
$$

$$
\lambda_t^a = (\alpha_{t-1}^a, \beta_{t-1}^a) \tag{46.69}
$$

$$
\lambda_t = [(\alpha_{t-1}^a, \beta_{t-1}^a) : a \in S_{\underline{a}}] \tag{46.70}
$$

$$
\begin{aligned}
Q_t(a,\lambda_t) &= \sum_r r\mathrm{Beta}(r;\alpha_{t-1}^a,\beta_{t-1}^a) && (46.71a)\\
&= \frac{\alpha_{t-1}^a}{\alpha_{t-1}^a + \beta_{t-1}^a} && (46.71b)
\end{aligned}
$$

The TS-MAB algorithm for a Beta agent and Bernoulli environment can be described by the bnet of Fig.46.6. For this special case of Fig.46.6, the TPMs, printed in blue, are as follows.

$$P(s_{<t}|s_{<t-1}, a_{t-1}, r_{t-1}) = \text{ same as for Fig.46.6} \tag{46.72}$$

$$P(\lambda_t^a|s_{<t}, \lambda_{t-1}) = \mathbb{1}(\lambda_t^a = \lambda_t^{*a}) = \begin{cases} \mathbb{1}(\lambda_t^a = \lambda_{t-1}^a) & \text{if } a \neq a_{t-1} \\ \mathbb{1}(\lambda_t^a = (\alpha_{t-2}^a + 1, \beta_{t-2}^a)) & \text{if } a_{t-1} = a \text{ and } r_{t-1} = 1 \\ \mathbb{1}(\lambda_t^a = (\alpha_{t-2}^a, \beta_{t-2}^a + 1)) & \text{if } a_{t-1} = a \text{ and } r_{t-1} = 0 \end{cases} \tag{46.73}$$

$$P(\underline{a}_t|\lambda_t) = \mathbb{1}(a_t = a_t(\lambda_t)) \tag{46.74}$$

$$= \mathbb{1}(a_t = \operatorname*{argmax}_a \underbrace{Q_t(a, \lambda_t)}_{\text{see Eq.(46.71).}}) \text{ (Beta agent response)} \tag{46.75}$$

$$P(r_t|a_t) = \mu_{a_t}^{r_t}(1 - \mu_{a_t})^{r_t} \text{ (Bernoulli environment response) .} \tag{46.76}$$

μ_a known to environment but unknown to agent.

TS-MAB algorithm, skeletal reprise

The TS-MAB algorithm is not very complicated but explaining it with precision requires nightmarishly many indices. Here is a pedagogical reprise of what we have said so far, where we have stripped out some of the inessential indices.

$$P(\lambda) = \mathbb{1}(\lambda, \lambda^*) \tag{46.77}$$

$$P(a|\lambda) = \mathbb{1}(a = a(\lambda)) \tag{46.78}$$

$$= \mathbb{1}(a = \operatorname*{argmax}_a \sum_r r P(\hat{r} = r|\hat{a} = a, \lambda)) \tag{46.79}$$

$$= \mathbb{1}(a = \operatorname*{argmax}_a \sum_r r \text{Beta}(r; \lambda^a)) \text{ (Beta agent response)} \tag{46.80}$$

Define $q()$ by

$$q(r; \lambda^a) = P(\hat{r} = r|\hat{a} = a, \lambda) \tag{46.81}$$

- PRIOR:

$$
\begin{aligned}
P(a) &= \sum_{\lambda} P(\lambda)P(a|\lambda) && (46.82)\\
&= P(a|\lambda^*) && (46.83)\\
&= \mathbb{1}(a = \operatorname*{argmax}_a \sum_r rq(r;\lambda^{*a}) && (46.84)
\end{aligned}
$$

  1. Use fact that q =Beta

$$\sum_r r\text{Beta}(r;\lambda^{*a}) = \frac{\alpha^{*a}}{\alpha^{*a}+\beta^{*a}} \tag{46.85}$$

  2. Don't use fact that q =Beta.
     For each a, get samples $r^\sigma \sim q(r;\lambda^{*a})$ for $\sigma = 0, 1, \ldots, nsam-1$ and estimate

$$\sum_r rq(r;\lambda^{*a}) \approx \frac{1}{nsam}\sum_\sigma r^\sigma q(r^\sigma;\lambda^{*a}) \ . \tag{46.86}$$

     This sampling is why TS is called a sampling.

- LIKELIHOOD:

$$P(r|a) = \mu_a^r(1-\mu_a)^r \ \ \text{(Bernoulli environment response)} \tag{46.87}$$

46.5.3 Grad-MAB algorithm

Let

$$\lambda_{t+1}(a) = \lambda_t(a) + \eta r_t[\mathbb{1}(a_t = a) - \pi_t(a)] \tag{46.88}$$

for some $\eta > 0$, where $\pi_t(a)$ is defined by

$$\pi_t(a) = P(\underline{a}_t = a|\lambda_t) = \underbrace{\frac{e^{\lambda_t(a)}}{\sum_a e^{\lambda_t(a)}}}_{\text{softmax}(\lambda_t)_a} \ . \tag{46.89}$$

The $\lambda_t(a)$ are called **scores** at time t. Let

$$\boxed{a_t^* = \operatorname*{argmax}_a \pi_t(a)} \tag{46.90}$$

The Gradient MAB (Grad-MAB) algorithm is described by the bnet Fig.46.7. This bnet differs from the bnet Fig.46.3 in that it includes new nodes $\underline{\lambda}_t$. The TPMs, printed in blue, for the nodes in the bnet Fig.46.7, are as follows.

$$P(\underline{\lambda}_{t+1} = \lambda|a_t, r_t, \lambda_t) = \mathbb{1}(\lambda = \lambda_{t+1} \text{ given by Eq.(46.88).} \tag{46.91}$$

Figure 46.7: Bnet for Grad-MAB algorithm.

$$P(a_t|\lambda_t) = \mathbb{1}(a_t = a_t^*) \quad \text{or, alternatively,} = P(\underline{a}_t = a_t|\lambda_t) = \pi_t(a_t) \quad \text{(agent's response)} \tag{46.92}$$

$$P(r_t|a_t) = P_{\underline{r}|\underline{a}}(r_t|a_t) \quad \text{(environment's response)} \; . \tag{46.93}$$

Motivation:
Define the **Instantaneous Average Reward** by

$$\mathcal{R}_t(\lambda_t) = \sum_a P(\underline{a}_t = a|\lambda_t) E_{\underline{r}_t|\underline{a}_t=a}[\underline{r}_t] = E_{\underline{r}_t,\underline{a}_t|\lambda_t}[\underline{r}_t] \; . \tag{46.94}$$

We will assume that as $t \to \infty$, $\underline{a}_t \to \underline{a}$, $\underline{r}_t \to \underline{r}$ and $\lambda_t \to \lambda$. Therefore, $\mathcal{R}_t(\lambda_t) \to E_{\underline{r},\underline{a}|\lambda}[\underline{r}] = E_{\underline{r}|\lambda}[\underline{r}]$.

Note that if $E_{\underline{r}_t|\underline{a}_t=a}[\underline{r}_t] = B$, where B is independent of a, then $\mathcal{R}_t(\lambda_t) = B$ and $\frac{\partial \mathcal{R}_t}{\partial \lambda_t(a)} = 0$.

Claim 57 *The gradient of $\mathcal{R}_t(\lambda_t)$ is*

$$\frac{\partial \mathcal{R}_t}{\partial \lambda_t(a)} = E_{\underline{r}_t,\underline{a}_t|\lambda_t}[g(\underline{r}_t, \underline{a}_t|a, \lambda_t)] \tag{46.95}$$

where

$$g(\underline{r}_t, \underline{a}_t|a, \lambda_t) = \underline{r}_t[\mathbb{1}(\underline{a}_t = a) - \pi_t(a)] \tag{46.96}$$

proof:

$$
\begin{aligned}
\frac{\partial \mathcal{R}_t}{\partial \lambda_t(a)} &= \sum_{a'} \frac{\partial \pi_t(a')}{\partial \lambda_t(a)} E_{\underline{r}_t|\underline{a}_t=a'}[\underline{r}_t] && (46.97)\\
&= \sum_{a'} \pi_t(a') \frac{\partial \ln \pi_t(a')}{\partial \lambda_t(a)} E_{\underline{r}_t|\underline{a}_t=a'}[\underline{r}_t] && (46.98)\\
&= \sum_{a'} \pi_t(a') \frac{\partial}{\partial \lambda_t(a)} \left[\ln \frac{\exp[\lambda_t(a')]}{\sum_a \exp[\lambda_t(a)]} \right] E_{\underline{r}_t|\underline{a}_t=a'}[\underline{r}_t] && (46.99)\\
&= \sum_{a'} \pi_t(a') [\mathbb{1}(a'=a) - \pi_t(a)] E_{\underline{r}_t|\underline{a}_t=a'}[\underline{r}_t] && (46.100)\\
&= \sum_{a'} P(\underline{a}_t = a'|\lambda_t) E_{\underline{r}_t|\underline{a}_t=a'}[\underline{r}_t(\mathbb{1}(a'=a) - \pi_t(a))] && (46.101)\\
&= E_{\underline{r}_t,\underline{a}_t|\lambda_t}[g(\underline{r}_t, \underline{a}_t|a, \lambda_t)] && (46.102)
\end{aligned}
$$

QED

Eq.(46.88) can be written as

$$
\lambda_{t+1}(a) = \lambda_t(a) + \eta g(r_t, a_t|a, \lambda_t) \tag{46.103}
$$

Chapter 47

Naive Bayes

Figure 47.1: bnet for Naive Bayes with 4 features

Class node $\underline{c} \in S_{\underline{c}}$. $|S_{\underline{c}}| = n_{\underline{c}}$= number of classes.
Feature nodes $\underline{x}_i \in S_{\underline{x}_i}$ for $i = 0, 1, 2, \ldots, F-1$. F=number of features.
Define

$$x. = [x_0, x_1, \ldots, x_{F-1}] \ . \tag{47.1}$$

For the bnet of Fig.47.1,

$$P(c, x.) = P(c) \prod_{i=0}^{F-1} P(x_i|c) \ . \tag{47.2}$$

Given $x.$ values, find most likely class $c \in S_{\underline{c}}$.
Maximum a Posteriori (MAP) estimate:

$$\begin{aligned} c^* &= \operatorname*{argmax}_c P(c|x.) && (47.3) \\ &= \operatorname*{argmax}_c \frac{P(c, x.)}{P(x.)} && (47.4) \\ &= \operatorname*{argmax}_c P(c, x.) \ . && (47.5) \end{aligned}$$

Chapter 48

Neural Networks

In this chapter, we discuss Neural Networks (NNs) of the feedforward kind, which is the most popular kind. In their plain, vanilla form, NNs only have deterministic nodes. But the nodes of a bnet can be deterministic too, because the TPM of a node can reduce to a delta function. Hence, NNs should be expressible as bnets. We will confirm this in this chapter.

Henceforth in this chapter, if we replace an index of an indexed quantity by a dot, it will mean the collection of the indexed quantity for all values of that index. For example, $x.$ will mean the array of x_i for all i.

Figure 48.1: Neural Network (feed forward) with 4 layers: input layer $\underline{x}.$, 2 hidden layers $\underline{h}^0.$, $\underline{h}^1.$ and output layer $\underline{Y}.$

Consider Fig.48.1.

$\underline{x}_i \in \{0,1\}$ for $i = 0,1,2,\ldots,nx-1$ is the **input layer**.

$\underline{h}^\lambda_i \in \mathbb{R}$ for $i = 0,1,2,\ldots,nh(\lambda)-1$ is the **λ-th hidden layer**. $\lambda = 0,1,2,\ldots,\Lambda-2$. A NN is said to be **deep** if $\Lambda > 2$; i.e., if it has more than one hidden layer.

$\underline{Y}_i \in \mathbb{R}$ for $i = 0,1,2,\ldots,ny-1$ is the **output layer**. We use a upper case y here because in the training phase, we will use pairs $(x.[\sigma], y.[\sigma])$ where $y_i[\sigma] \in \{0,1\}$

for $i = 0, 1, \ldots, ny - 1$. $Y = \hat{y}$ is an estimate of y. Note that lower case y is either 0 or 1, but upper case y may be any real. Often, the activation functions are chosen so that $Y \in [0, 1]$.

The number of nodes in each layer and the number of layers are arbitrary. Fig.48.1 is fully connected (aka dense), meaning that every node of a layer is impinged arrow coming from every node of the preceding layer. Later on in this chapter, we will discuss non-dense layers.

Let $w^{\lambda}_{i|j}, b^{\lambda}_{i} \in \mathbb{R}$ be given, for $i \in \mathbb{Z}_{[0,nh(\lambda))}$, $j \in \mathbb{Z}_{[0,nh(\lambda-1))}$, and $\lambda \in \mathbb{Z}_{[0,\Lambda)}$.

These are the TPMs, printed in blue, for the nodes of the bnet Fig.48.1:

$$P(x_i \mid x_{i-1}, x_{i-1}, \ldots, x_0) = \text{given} \tag{48.1}$$

$$P(h^{\lambda}_{i} \mid h^{\lambda-1}_{\cdot}) = \delta\left(h^{\lambda}_{i}, \mathcal{A}^{\lambda}_{i}(\sum_{j} w^{\lambda}_{i|j} h^{\lambda-1}_{j} + b^{\lambda}_{i})\right) , \tag{48.2}$$

where $P(h^0_i|h^{-1}) = P(h^0_i|x)$.

$$P(Y_i \mid h^{\Lambda-2}_{\cdot}) = \delta\left(Y_i, \mathcal{A}^{\Lambda-1}_{i}(\sum_{j} w^{\Lambda-1}_{i|j} h^{\Lambda-2}_{j} + b^{\Lambda-1}_{i})\right) . \tag{48.3}$$

48.1 Activation Functions $\mathcal{A}^{\lambda}_{i} : \mathbb{R} \to \mathbb{R}$

Activation functions must be nonlinear.

- **Step function (Perceptron)**

$$\mathcal{A}(x) = \mathbb{1}(x > 0) \tag{48.4}$$

  Zero for $x \leq 0$, one for $x > 0$.

- **Sigmoid function**

$$\mathcal{A}(x) = \frac{1}{1 + e^{-x}} = \text{smoid}(x) \tag{48.5}$$

  Smooth, monotonically increasing function. $\text{smoid}(-\infty) = 0$,$\text{smoid}(0) = 1/2$, $\text{smoid}(\infty) = 1$.

- **Hyperbolic tangent**

$$\mathcal{A}(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{48.6}$$

Smooth, monotonically increasing function. $\tanh(-\infty) = -1$,$\tanh(0) = 0$, $\tanh(\infty) = 1$.

Odd function:

$$\tanh(-x) = -\tanh(x) \tag{48.7}$$

Whereas $\text{smoid}(x) \in [0, 1]$, $\tanh(x) \in [-1, 1]$.

- **ReLU (Rectified Linear Unit)**

$$\mathcal{A}(x) = \underbrace{x\mathbb{1}(x > 0)}_{x_+} = \max(0, x) \ . \tag{48.8}$$

Compare this to the step function $\mathbb{1}(x > 0)$.

- **Swish**

$$\mathcal{A}(x) = x \ \text{smoid}(x) \tag{48.9}$$

- **Softmax**

$$\mathcal{A}(x_i|x.) = \frac{e^{x_i}}{\sum_i e^{x_i}} = \text{softmax}(x.)_i \tag{48.10}$$

The softmax definition implies that the bnet nodes within a softmax layer are fully connected by arrows to form a "clique".

48.2 Weight optimization via supervised training and gradient descent

The bnet of Fig.48.1 is used for classification of a single data point x.. It assumes that the weights $w^{\lambda}_{i|j}, b^{\lambda}_i$ are given.

To find the optimum weights via supervised training and gradient descent, one uses the bnet Fig.48.2.

In Fig.48.2, the nodes in Fig.48.1 become sampling space vectors. For example, $\underline{x}.$ becomes $\underline{\vec{x}.}$, where the components of $\underline{\vec{x}.}$ in sampling space are $\underline{x}.[\sigma] \in \{0, 1\}^{nx}$ for $\sigma = 0, 1, \ldots, nsam(\vec{x}) - 1$.

$nsam(\vec{x})$ is the number of samples used to calculate the gradient during each **stage (aka iteration)** of Fig.48.2. We will also refer to $nsam(\vec{x})$ as the **mini-batch size**. A **mini-batch** is a subset of the training dataset.

To train a bnet with a data set (d-set), the standard procedure is to split the d-set into 3 parts:

1. **training d-set**,

2. **testing1 d-set**, for tuning of hyperparameters like $nsam(\vec{x})$, Λ, and $nh(i)$ for each i.

3. **testing2 d-set**, for measuring how well the model tuned with the testing1 d-set performs.

The training d-set is itself split into mini-batches. An **epoch** is a pass through all the training d-set.

Define

$$W^{\lambda}_{i|j} = [w^{\lambda}_{i|j}, b^{\lambda}_{i}] \ . \tag{48.11}$$

These are the TPMs, printed in blue, for the nodes of the bnet Fig.48.2:

$$\begin{tikzcd}
 & & \underline{\vec{x}.} \arrow[r] \arrow[d] & \underline{\vec{y}.} \arrow[dd] & \\
 & \underline{W}^{0}_{.|.}, \arrow[r] & \underline{\vec{h}^{0}_{.}} \arrow[d] & & \\
 & \underline{W}^{1}_{.|.} \arrow[r] & \underline{\vec{h}^{1}_{.}} \arrow[d] & & \\
 & \underline{W}^{2}_{.|.} \arrow[r] & \underline{\vec{h}^{2}_{.}} \arrow[d] & & \\
\underline{W}^{\cdot}_{.|.} \arrow[uuur] \arrow[uur] \arrow[ur] \arrow[rrr, bend right] & & \underline{\vec{Y}}. \arrow[r] & \mathcal{E} \arrow[r] & (\underline{W'})^{\cdot}_{.|.}
\end{tikzcd}$$

Figure 48.2: bnet for finding optimum weights of the bnet Fig.48.1 via supervised training and gradient descent.

$$P(x.[\sigma]) = \text{given} \ . \tag{48.12}$$

$$P(y.[\sigma] \mid x.[\sigma]) = \text{given} \ . \tag{48.13}$$

$$P(h^{\lambda}_{i}[\sigma] \mid h^{\lambda-1}_{\cdot}[\sigma]) = \delta\left(h^{\lambda}_{i}[\sigma], \mathcal{A}^{\lambda}_{i}(\sum_{j} w^{\lambda}_{i|j} h^{\lambda-1}_{j}[\sigma] + b^{\lambda}_{i})\right) \tag{48.14}$$

$$P(Y_i[\sigma] \mid h_{\cdot}^{\Lambda-2}[\sigma]) = \delta\left(Y_i[\sigma], \mathcal{A}_i^{\Lambda-1}(\sum_j w_{i|j}^{\Lambda-1} h_j^{\Lambda-2}[\sigma] + b_i^{\Lambda-1})\right) \tag{48.15}$$

$$P(W_{\cdot|\cdot}) = \text{given} \tag{48.16}$$

The first time it is used, $W_{\cdot|\cdot}$ is arbitrary. After the first time, it is determined by previous stage.

$$P(W_{\cdot|\cdot}^{\lambda}|W_{\cdot|\cdot}) = \delta(W_{\cdot|\cdot}^{\lambda}, (W_{\cdot|\cdot})^{\lambda}) \tag{48.17}$$

$$P(\mathcal{E}|\vec{y}\cdot, \vec{Y}\cdot) = \frac{1}{nsam(\vec{x})} \sum_{\sigma} \sum_{i} d(y_i[\sigma], Y_i[\sigma]) \ , \tag{48.18}$$

where

$$d(y, Y) = |y - Y|^2 \ . \tag{48.19}$$

If $y, Y \in [0, 1]$, one can use this instead

$$d(y, Y) = XE(y \to Y) = -y \ln Y - (1 - y) \ln(1 - Y) \ . \tag{48.20}$$

$$P((W')_{i|j}^{\lambda}|\mathcal{E}, W_{\cdot|\cdot}) = \delta((W')_{i|j}^{\lambda}, W_{i|j}^{\lambda} - \eta \partial_{W_{i|j}^{\lambda}} \mathcal{E}) \tag{48.21}$$

$\eta > 0$ is called the learning rate. This method of minimizing the error $\mathcal{E}$ is called gradient descent. $W' - W = \Delta W = -\eta \partial_W \mathcal{E}$ so $\Delta \mathcal{E} = \frac{-1}{\eta}(\Delta W)^2 < 0$.

48.3 Non-dense layers

The TPM for a non-dense layer is of the form:

$$P(h_i^{\lambda}[\sigma] \mid h_{\cdot}^{\lambda-1}[\sigma]) = \delta(h_i^{\lambda}[\sigma], H_i^{\lambda}[\sigma]) \ , \tag{48.22}$$

where $H_i^{\lambda}[\sigma]$ will be specified below for each type of non-dense layer.

- **Dropout Layer**

  The dropout layer was invented in Ref.[48]. To dropout nodes from a fixed layer λ: For all i of layer λ, define a new node $\underline{r}_i^{\lambda}$ with an arrow $\underline{r}_i^{\lambda} \to \underline{h}_i^{\lambda}$. For $r \in \{0, 1\}$, and some $p \in (0, 1)$, define

$$P(r_i^{\lambda} = r) = [p]^r [1 - p]^{1-r} \text{ (Bernoulli dist.)} \ . \tag{48.23}$$

Now one has

$$P(h_i^\lambda[\sigma] \mid h_\cdot^{\lambda-1}[\sigma], r_i^\lambda) = \delta(h_i^\lambda[\sigma], H_i^\lambda[\sigma]) \ , \tag{48.24}$$

where

$$H_i^\lambda[\sigma] = \mathcal{A}_i^\lambda(r_i^\lambda \sum_j w_{i|j}^\lambda h_j^{\lambda-1}[\sigma] + b_i^\lambda) \ . \tag{48.25}$$

This reduces overfitting. Overfitting might occur if the weights follow too closely several similar minibatches. This dropout procedure adds a random component to each minibatch making groups of similar minibatches less likely.

The random $\underline{r}_i^\lambda$ nodes that induce dropout are only used in the training bnet Fig.48.2, not in the classification bnet Fig.48.1. We prefer to remove the $\underline{r}_i^\lambda$ stochasticity from classification and for Fig.48.1 to act as an average over sampling space of Fig.48.2. Therefore, if weights $w_{i|j}^\lambda$ are obtained for a dropout layer λ in Fig.48.2, then that layer is used in Fig.48.1 with no $\underline{r}_i^\lambda$ nodes but with weights $\langle r_i^\lambda \rangle w_{i|j}^\lambda = p w_{i|j}^\lambda$.

Note that dropout adds non-deterministic nodes to a NN, which in their vanilla form only have deterministic nodes.

- **Convolutional Layer**

  - 1-dim
    Filter function $\mathcal{F} : \{0, 1, \ldots, nf - 1\} \to \mathbb{R}$.
    σ=stride length
    For $i \in \{0, 1, \ldots, nh(\lambda) - 1\}$, let

    $$H_i^\lambda[\sigma] = \sum_{j=0}^{nf-1} h_{j+i\sigma}^{\lambda-1}[\sigma] \mathcal{F}(j) \ . \tag{48.26}$$

    For the indices not to go out of bounds in Eq.(48.26), we must have

    $$nh(\lambda - 1) - 1 = nf - 1 + (nh(\lambda) - 1)\sigma \tag{48.27}$$

    so

    $$nh(\lambda) = \frac{1}{\sigma}[nh(\lambda - 1) - nf] + 1 \ . \tag{48.28}$$

  - 2-dim
    $h_i^\lambda[\sigma]$ becomes $h_{(i,j)}^\lambda[\sigma]$. Do 1-dim convolution along both i and j axes.

- **Pooling Layers (MaxPool, AvgPool)**

  Here each node i of layer λ is impinged by arrows from a subset $Pool(i)$ of the set of all nodes of the previous layer $\lambda - 1$. Partition set $\{0, , 1, \ldots, nh(\lambda - 1) - 1\}$ into $nh(\lambda)$ mutually disjoint, nonempty sets called $Pool(i)$, where $i \in \{0, 1, \ldots, nh(\lambda) - 1\}$.

  - AvgPool
  $$H_i^\lambda[\sigma] = \frac{1}{|Pool(i)|} \sum_{j \in Pool(i)} h_j^{\lambda-1}[\sigma] \tag{48.29}$$
  - MaxPool
  $$H_i^\lambda[\sigma] = \max_{j \in Pool(i)} h_j^{\lambda-1}[\sigma] \tag{48.30}$$

48.4 Autoencoder NN

If the sequence

$$nx, nh(0), nh(1), \ldots, nh(\Lambda - 2), ny \tag{48.31}$$

first decreases monotonically up to layer λ_{min}, then increases monotonically until $ny = nx$, then the NN is called an **autoencoder NN**. Autoencoders are useful for unsupervised learning and feature reduction. In this case, Y estimates x. The layers before layer λ_{min} are called the **encoder**, and those after λ_{min} are called the **decoder**. Layer λ_{min} is called the **code**.

Chapter 49

Noisy-OR gate

The Noisy-OR gate was first proposed by Judea Pearl in his 1988 book Ref.[35].

Figure 49.1: Noisy-OR gate $\underline{y} \in \{0,1\}$ with $n = 3$, Boolean inputs $(\underline{x}_i)_{i=0,1,2}$ and parameters $\underline{\lambda}, (\underline{\pi})_{i=0,1,2}$.

Let
$\underline{\lambda} \in [0,1]$ =**gate leakage**.
$y \in \{0,1\}$ = gate output
$\underline{x}^n = (\underline{x}_i)_{i=0,1,\ldots,n-1}$, where $\underline{x}_i \in \{0,1\}$ are gate inputs.
$\underline{\pi}^n = (\underline{\pi}_i)_{i=0,1,\ldots,n-1}$, where $\underline{\pi}_i \in [0,1]$ are gate parameters.
The TPM, printed in blue, for the Noisy-OR gate $\underline{y}$ shown in Fig.49.1, is

$$P(y=1|x^n,\lambda,\pi^n) = 1-(1-\lambda)\prod_i[1-\pi_i x_i] \tag{49.1}$$

$$P(y=0|x^n,\lambda,\pi^n) = 1-P(y=1|x^n,\lambda,\pi^n) \tag{49.2}$$

Note that if $\lambda = 0$ and $\pi_i = 1$ for all i, then this becomes a deterministic OR-gate. Indeed,

$$P(y=1|x^n,\lambda=0,\pi^n=1^n) = 1-\prod_i[1-x_i] = \vee_{i=0}^{n-1} x_i \ , \tag{49.3}$$

so

$$P(y|x^n, \lambda = 0, \pi^n = 1^n) = \delta(y, \vee_{i=0}^{n-1} x_i) \ . \tag{49.4}$$

49.1 3 ways to interpret the parameters π_i

1. Note that if $\lambda = 0$ and x^n is one hot (i.e., $x^n = e_i^n$, where e_i^n is the vector with all components zero except for the i-th component which equals 1), then

$$P(y = 1|x^n = e_i^n, \lambda = 0, \pi^n) = 1 - [1 - \pi_i] = \pi_i \ . \tag{49.5}$$

   This gives an interpretation to the parameters π_i.

Figure 49.2: Fig.49.1 after replacing parameters $(\underline{\pi}_i)_{i=0,1,2}$ by hidden nodes $(\underline{h}_i)_{i=0,1,2}$.

2. Another way of interpreting the parameters π_i is to associate each of them with a hidden variable $\underline{h}_i \in \{0, 1\}$ whose average equals π_i. More precisely, consider Fig.49.2.

   Let $\underline{x}_i, \underline{h}_i, \underline{A}_i, \underline{y} \in \{0, 1\}$.

   The TPMs, printed in blue, for the nodes of the bnet Fig.49.2, are as follows:

$$P(h_i) = \pi_i \delta(h_i, 1) + (1 - \pi_i)\delta(h_i, 0) \tag{49.6}$$

$$P(A_i|h_i, x_i) = \delta(A_i, h_i \wedge x_i) = \delta(A_i, h_i x_i) \tag{49.7}$$

$$P(y=1|A^n) = 1-(1-\lambda)\wedge_{i=0}^{n-1}\overline{A}_i \tag{49.8}$$

$$= 1-(1-\lambda)\prod_i(1-A_i) \tag{49.9}$$

$$P(y=0|A^n) = 1-P(y=1|A^n) \tag{49.10}$$

Note that

$$P(y=1|x^n,\lambda) = \sum_{h^n}\sum_{A^n}\left[1-(1-\lambda)\prod_i(1-A_i)\right][\prod_i\delta(A_i,h_ix_i)]P(h^n) \tag{49.11}$$

$$= E_{\underline{h}^n}\left[[1-(1-\lambda)\prod_i(1-h_ix_i)\right] . \tag{49.12}$$

But

$$E_{\underline{h}_i}[h_ix_i] = \sum_{h_i=0,1}P(h_i)h_ix_i = \pi_ix_i \tag{49.13}$$

so

$$P(y=1|x^n,\lambda) = 1-(1-\lambda)\prod_i(1-\pi_ix_i) . \tag{49.14}$$

Figure 49.3: Fig.49.2 after replacing the hidden nodes $(\underline{h}_i)_{i=0,1,2}$ by vectors of samples $(\vec{\underline{h}}_i)_{i=0,1,2}$.

3. Another way to interpret the parameters π_i is to associate each of them with a vector of samples $\vec{\underline{h}}_i$ whose average is π_i. More precisely, consider Fig.49.3.

   Suppose $\underline{h}_i \in \{0,1\}$ and define

$$P_{\underline{h}_i}(h_i) = \pi_i \delta(h_i, 1) + (1-\pi_i)\delta(h_i, 0) \ . \tag{49.15}$$

   Suppose $\vec{\underline{h}}_i = (\underline{h}_i[\sigma])_{s=0,1,\ldots,nsam-1}$ and the Boolean samples $\underline{h}_i[\sigma] \in \{0,1\}$ are i.i.d. with $\underline{h}_i[\sigma] \sim P_{\underline{h}_i}$ for all σ.

   Note that for each i, an estimate $\hat{P}_{\underline{h}_i}(h_i)$ of $P_{\underline{h}_i}(h_i)$ can be obtained from the vector of samples $\vec{\underline{h}}_i$ as follows:

$$\hat{P}_{\underline{h}_i}(h_i) = \frac{1}{nsam} \sum_{\sigma=0}^{nsam-1} \mathbb{1}(h_i[\sigma] = h_i) \ . \tag{49.16}$$

   Let $\underline{x}_i, \underline{h}_i[\sigma], \underline{A}_i, \underline{y} \in \{0,1\}$.

   The TPMs, printed in blue, for the nodes of the bnet Fig.49.3, are as follows:

$$P(\vec{h}_i) = \prod_{\sigma=0}^{nsam-1} P_{\underline{h}}(h_i[\sigma]) \tag{49.17}$$

$$\begin{aligned} P(A_i \mid \vec{h}_i, x_i) &= \delta(A_i, \frac{1}{nsam} \sum_{\sigma} h_i[\sigma] \wedge x_i) && (49.18) \\ &= \delta(A_i, \pi_i x_i) && (49.19) \end{aligned}$$

$$\begin{aligned} P(y=1|A^n) &= 1 - (1-\lambda) \wedge_{i=0}^{n-1} \overline{A}_i && (49.20) \\ &= 1 - (1-\lambda) \prod_i (1 - A_i) && (49.21) \end{aligned}$$

$$P(y=0|A^n) = 1 - P(y=1|A^n) \tag{49.22}$$

   Note that

$$\begin{aligned} P(y=1|x^n, \lambda, \vec{h}^n) &= \sum_{A^n} \left[1 - (1-\lambda) \prod_i (1 - A_i) \right] \prod_i \delta(A_i, \pi_i x_i) && (49.23) \\ &= 1 - (1-\lambda) \prod_i (1 - \pi_i x_i) \ . && (49.24) \end{aligned}$$

Chapter 50

Non-negative Matrix Factorization

Based on Ref.[112].

Given matrix V, factor it into product of two matrices

$$V = WH \ , \tag{50.1}$$

where all 3 matrices have non-negative entries.

$V \in \mathbb{R}_{\geq 0}^{nv \times na}$: visible info matrix

$W \in \mathbb{R}_{\geq 0}^{nv \times nh}$: weight info matrix

$H \in \mathbb{R}_{\geq 0}^{nh \times na}$: hidden info matrix

Usually, $nv > nh < na$ so compression of information (aka dimensional reduction, clustering)

50.1 Bnet interpretation

Express node $\underline{v}$ as a chain of two nodes.

$$\underline{v} \longleftarrow \underline{a} \qquad = \qquad \underline{w} \longleftarrow \underline{h} \longleftarrow \underline{a}$$

Figure 50.1: Bnet interpretation of non-negative matrix factorization.

Node TPMs, printed in blue, for Fig.50.1.

$$P(\underline{v} = w|a) = \frac{V_{w,a}}{\sum_w V_{w,a}} \tag{50.2}$$

$$P(w|h) = \frac{W_{w,h}}{\sum_w W_{w,h}} \tag{50.3}$$

$$P(h|a) = \frac{\sum_w W_{w,h}}{\sum_w V_{w,a}} H_{h,a} \tag{50.4}$$

50.2 Simplest recursive algorithm

Initialize: Choose nh. Choose $W^{(0)}$ and $H^{(0)}$ that have non-negative entries.

Update: For $n = 0, 1, \ldots,$ do

$$H_{i,j}^{(n+1)} \leftarrow H_{i,j}^{(n)} \frac{[(W^{(n)})^T V]_{i,j}}{[(W^{(n)})^T \underbrace{W^{(n)} H^{(n)}}_{\approx V}]_{i,j}} \tag{50.5}$$

and

$$W_{i,j}^{(n+1)} \leftarrow W_{i,j}^{(n)} \frac{[V(H^{(n+1)})^T]_{i,j}}{[\underbrace{W^{(n)} H^{(n+1)}}_{\approx V}(H^{(n+1)})^T]_{i,j}} \ . \tag{50.6}$$

After each step, record error defined by

$$\mathcal{E}^{(n)} = \| \ V - W^{(n)} H^{(n)} \ \|_2 \ \ . \tag{50.7}$$

Using 2-norm, aka Frobenius matrix norm. Continue until reach acceptable error.

Can also use Kullback-Liebler divergence for error:

$$\mathcal{E} = \sum_a P(a) D_{KL}(P(\underline{v} = w|a) \ \| \sum_h P(w|h) P(h|a)) \ , \tag{50.8}$$

for some arbitrary choice of prior $P(a)$. For example, can choose $P(a)$ uniform.

Chapter 51

Observationally Equivalent DAGs

This chapter is based on Chapter 1 of Ref.[37] and on a blog post by Bruno Gonçalves (Ref.[11]).

A probability distribution P **is compatible with a DAG** G if P and G have the same random variables, and they can be combined to form a bnet without contradictions; i.e., one can calculate all the TPMs from P and multiply them together to obtain P again. Let

$$\mathcal{P}(G) = \{P : P \text{ is compatible with } G\} \ . \tag{51.1}$$

Two DAGs G **and** G' **are observationally equivalent (OE)** if $\mathcal{P}(G) = \mathcal{P}(G')$. Hence, any total probability distribution that is compatible with one of them is compatible with the other. For example, $\underline{a} \to \underline{b}$ and $\underline{a} \leftarrow \underline{b}$ are OE because

$$P(a|b)P(b) = P(a,b) = P(b|a)P(a) \ . \tag{51.2}$$

We'll say two bnets are OE if their DAGs are OE.

Two DAGs G and G' are **d-separation equivalent** if $DS(G) = DS(G')$. See Chapter 17 for definition of $DS(G)$.

Claim 58 *Two DAGs are OE iff their DAGs are d-separation equivalent.*

The **skeleton** of a DAG is its underlying undirected graph.

A **v-structure** in a DAG consists of two arrows converging to a node and such that their tails are not connected by a third arrow. Fig.51.1 shows in red all the v-structures of a particular DAG.

Claim 59 *Observational Equivalence Theorem (by Verma and Pearl, 1990)*
Two DAGs are OE iff they have the same skeletons and the same v-structures.

51.1 Examples

The 3 DAGs in Fig.51.2 are OE. They form an equivalence class of OE DAGs that represent the same probability distribution. This equivalence class of DAGs can be

Figure 51.1: Example showing in red all v-structures of a particular DAG.

Figure 51.2: These 3 DAGs are observationally equivalent (OE).

Figure 51.3: This partially directed graph represents the 3 DAGs in Fig.51.2.

represented by the partially directed graph Fig.51.3. These 3 DAGs can be proven to be OE in the following 3 ways:

1. Write the generic probability distributions represented by the 3 DAGs, and show that they are equal, as we did in Eq.(51.2). That is the low brow way of proving OE.

2. Use d-separation (see Chapter 17). Consider DAG (a) first. Rename the nodes as $\underline{\tau}_j$ with $j = 1, 2, \ldots$ so that the names are in topological order (i.e., so that the parents of $\underline{\tau}_j$ have indices that are smaller than j). The node names $\underline{x}_j$ of DAG (a) are already in topological order, so we skip this step for DAG (a). Now write down its total probability distribution and notice which parents of a fully connected DAG were omitted.

$$P(x_1, x_2, x_3, x_4, x_5) = \underbrace{P(x_5|x_4)}_{x_3, x_2, x_1 \text{ omitted}} \underbrace{P(x_4|x_3, x_2)}_{x_1 \text{ omitted}} \underbrace{P(x_3|x_1)}_{x_2 \text{ omitted}} P(x_2|x_1)P(x_1) \quad (51.3)$$

The observations of which parents were omitted can be stated in d-separation lingo as the following 3 orthogonality relations:[1]

$$\underline{x}_3 \perp_P \underline{x}_2 \quad | \quad \underline{x}_1 \quad (51.4a)$$
$$\underline{x}_4 \perp_P \underline{x}_1 \quad | \quad \underline{x}_2, \underline{x}_3 \quad (51.4b)$$
$$\underline{x}_5 \perp_P (\underline{x}_1, \underline{x}_2, \underline{x}_3) \quad | \quad \underline{x}_4 \ . \quad (51.4c)$$

Going through the same procedure for the other 2 DAGs yields, for each of them, an equivalent set of 3 orthogonality equations.[2]

This is enough to conclude that the 3 DAGs of Fig.51.2 are OE.

Note that Eqs.(51.4) encompass all that there is to say about the observability of DAG (a). These 3 equations can be checked empirically to assess how well the DAG fits the data. For example, one can do OLS (ordinary least squares) regression $\underline{x}_5 \sim \underline{x}_1 + \underline{x}_2 + \underline{x}_3 + \underline{x}_4$ on the data, i.e., try to fit $x_5 = \beta_0 + \sum_{i=1}^{4} \beta_i x_i$ to the data, and find that, to a good approximation, $\beta_1 = \beta_2 = \beta_3 = 0$.

3. Use the OE Theorem. All three DAGs have the same skeleton, and the same single v-structure $\underline{x}_2 \rightarrow \underline{x}_4 \leftarrow \underline{x}_3$.

[1] Normally, if we had changed from the original node names to the $\underline{\tau}_j$ node names, these orthogonality relations would first be stated in terms of the $\underline{\tau}_j$ names, and we could translate them so that they were stated in terms of the original node names. But for DAG (a) there was no need to use the $\underline{\tau}_j$ names.

[2] The $\underline{x}_j$ node names are no longer in topological order for DAGs (b) and (c) so for them you should go through the intermediate step of renaming the nodes $\underline{\tau}_j$, and then, after obtaining the orthogonality relations in terms of the $\underline{\tau}_j$ names, translating them back to the original $\underline{x}_j$ names.

Chapter 52

Potential Outcomes

This chapter is based on Ref.[6], a book by Stephen Cunningham entitled "Causal inference: the mixtape".

The theory of potential outcomes (PO) was for the most part invented in a seminal 1974 paper by Donald B. Rubin. Rubin has also made important extensions to PO theory since 1974. However, he does not use Pearl's causal DAGs to discuss PO theory. Pearl has shown that PO theory can be substantially clarified and extended by using the language of causal DAGs. The d-separation theorem and do operator that we discuss in Chapters 17 and 52 are especially useful in this regard. In this chapter, we stress the connection of PO theory to Pearl's causal DAGs and bnets.

σ	d^σ	y^σ	$y^\sigma(0)$	$y^\sigma(1)$
Edith	0	5	5	.
Frank	0	7	7	.
George	0	8	8	.
Hank	0	10	10	.
Andy	1	10	.	10
Ben	1	5	.	5
Chad	1	16	.	16
Daniel	1	3	.	3

Table 52.1: PO dataset describing whether individual σ took a treatment dose ($d^\sigma = 1$) or didn't ($d^\sigma = 0$). The treatment outcome is measured by the real number y^σ.

Suppose a **population of individuals** $\sigma = 0, 1, 2, \ldots, nsam - 1$ is given ($d^\sigma = 1$) or is not given ($d^\sigma = 0$) a **treatment discrete drug dose** d^σ, and that the **treatment outcome (i.e., response)** is measured by a real number y^σ. Table 52.1 gives a possible **PO dataset** for this scenario. As you can see from that table, each individual either takes a drug dose or doesn't, but not both. PO theory can be viewed as a **missing data (MD) problem**. MD problems are discussed in Chapter 44. However, the PO MD problem is much more specialized than the generic MD problems discussed in Chapter 44. In the PO MD problem, we can fill in the blank

cells by matching each individual that took the drug with another *similar* individual that didn't. We will have much more to say about this matching strategy later in this chapter.

One can define similar individuals as individuals that have the same value for nx features $x^\sigma = (x_i^\sigma)_{i=0,1,\ldots,nx-1}$. One can add to Table 52.1 nx extra columns giving the value of the feature vector x^σ for each individual. Members of a population with the same x^σ are referred to as a **subpopulation or stratum (i.e., layer)**.

In a **randomized controlled trial (RCT)** [1], the effect of the variable x^σ on the value of d^σ is eliminated by randomizing the population and therefore making the effect of x^σ on d^σ average out to zero. However, there are many situations in which carrying out a RCT is not possible or desirable. PO theory is a way of predicting the result of a RCT in situations where doing a real RCT is not possible or desirable.

In this chapter, x^σ will be called the confounders. Implicit throughout this chapter is the assumption that there are **no unmeasured confounders**. Because if there are some unmeasured confounders, those can send secret messages that influence the value that d^σ takes. This would ruin the predictions of someone trying to predict the results of a RCT without being privy to those secret messages. When there are **some unmeasured confounders**, it might still be possible to predict the effect of a RCT. This might be possible using instrumental variables. See Chapter 31 for a discussion of **instrumental variables**.

52.1 G and G_{den} bnets, the starting point bnets

Figure 52.1: Bnets G and G_{den} are our starting point in discussing PO theory. G is for a single individual σ of the population. Bnet G_{den} is the DEN counterpart to G. DEN (Deterministic with External Noise) bnets are discussed in Chapter 36.

In this chapter, we will abbreviate $\underline{X}[\sigma] = \underline{X}^\sigma$ for $X \in \{d, x, y\}$, where $\sigma \in$

[1] The term **A/B test** is often used to mean a RCT where A and B are the treated and control groups. However, sometimes the term is used to refer to an experiment that conditions on confounders, which violates the definition of a RCT, and is the same as a PO test.

$\{0, 1, 2, \ldots, nsam - 1\}$.

For each individual (aka unit, sample) $\sigma = 0, 1, 2, \ldots nsam - 1$, let:

$\underline{d}^\sigma \in \{0, 1\}$ be the treatment discrete drug dose. It equals 1 if treated and 0 if untreated.

$\underline{y}^\sigma \in \mathbb{R}$ be the treatment potential outcome

$\underline{x}^\sigma$ be the column vector of treatment confounders (aka covariates because they are often used as covariates (i.e., independent variables) in linear regression.)

Consider bnets G and G_{den} in Fig.52.1. G reflects the language used in Ref.[6] to discuss PO theory. And G_{den} reflects the language that Judea Pearl prefers to use to discuss PO theory. Both languages are equivalent. To go from one language to the other, one need only perform the following swaps, where $\underline{u}$ is the external noise of the DEN bnet.

$\underline{X}^\sigma \leftrightarrow \underline{X}(\underline{u})$ for $X \in \{d, x, y\}$.

$P(\sigma) = \frac{1}{nsam} \leftrightarrow P(u)$

$\sum_\sigma P(\sigma)(\cdot) \leftrightarrow \sum_u P(u)(\cdot)$

The TPMs, printed in blue, for the bnet G in Fig.52.1, are as follows:

$$P(x^\sigma) = P_{\underline{x}}(x^\sigma) \tag{52.1}$$

$$P(d^\sigma | x^\sigma) = P_{\underline{d}|\underline{x}}(d^\sigma | x^\sigma) \tag{52.2}$$

$$P(y^\sigma | d^\sigma, x^\sigma) = P_{\underline{y}|\underline{d},\underline{x}}(y^\sigma | d^\sigma, x^\sigma) \tag{52.3}$$

Now let:

$\underline{d} \in \{0, 1\}$ be the treatment discrete drug dose. It equals 1 if treated and 0 if untreated

$\underline{y} \in \mathbb{R}$ be the treatment potential outcome

$\underline{x}$ be the column vector of treatment confounders (aka covariates)

$\underline{u} = (\underline{u}_{\underline{d}}, \underline{u}_{\underline{x}}, \underline{u}_{\underline{y}})$ be the external noise

The TPMs, printed in blue, for the bnet G_{den} in Fig.52.1, are as follows:

$$P(x | u_{\underline{x}}) = \mathbb{1}(\ \ x = u_{\underline{x}}\ \ ) \tag{52.4}$$

$$P(d | x, u_{\underline{d}}) = \mathbb{1}(\ \ d = f_{\underline{d}}(x, u_{\underline{d}})\ \ ) \tag{52.5}$$

$$P(y | d, x, u_{\underline{y}}) = \mathbb{1}(\ \ y = f_{\underline{y}}(d, x, u_{\underline{y}})\ \ ) \tag{52.6}$$

If we linearize $f_{\underline{y}}$ in Eq.(52.6), we get

$$\underline{y} = \delta \underline{d} + \beta \underline{x} + \underline{u}_{\underline{y}} \ , \tag{52.7}$$

where $\delta, \beta \in \mathbb{R}$. Assuming that $\underline{x}, \underline{y} \in \mathbb{R}$ and $\underline{d} \in \{0, 1\}$, Eq.(52.7) can be plotted. The resulting plot is given in Fig.52.2. This plot is a very special case of the PO problem, but it gives a crude idea of the "effects" $\delta = y(1) - y(0)$ that PO theory gives estimates for. Any individual participating in the experiment experiences either $y(1)$ or $y(0)$, but not both.

Figure 52.2: Plot of Eq.(52.7)

52.2 G bnet with nodes $y^\sigma(0), y^\sigma(1)$ added to it.

Figure 52.3: Bnet G_+ is bnet G with two new nodes $\underline{y}^\sigma(0)$ and $\underline{y}^\sigma(1)$ added to it. The tuple node $[\underline{y}^\sigma(0), \underline{y}^\sigma(1)]$ can also be represented by two nodes $\underline{c} \to \underline{y}(\underline{c})$, where $\underline{c} \in \{0, 1\}$.

Consider Fig.52.3. Bnet G_+ was obtained by adding two new nodes $\underline{y}^\sigma(0)$ and $\underline{y}^\sigma(1)$ to bnet G. The TPMs, printed in blue, for bnet G_+, are as follows. Note that

we define them in terms of the TPMs for bnet G.

$$P(x^\sigma) = P_{\underline{x}}(x^\sigma) \tag{52.8}$$

$$P(d^\sigma|x^\sigma) = P_{\underline{d}|\underline{x}}(d^\sigma|x^\sigma) \tag{52.9}$$

For $c \in \{0,1\}$,

$$P(y^\sigma(c)|d^\sigma, x^\sigma) = \begin{cases} P_{\underline{y}(c)|\underline{d},\underline{x}}(y^\sigma(c)|d^\sigma, x^\sigma) & \text{if INCLUDE arrow with question mark} \\ P_{\underline{y}(c)|\underline{x}}(y^\sigma(c)|x^\sigma) & \text{if EXCLUDE arrow wit question mark} \end{cases} \tag{52.10}$$

$$\begin{aligned} P(y^\sigma|y^\sigma(0), y^\sigma(1), d^\sigma) = &= \mathbb{1}(y^\sigma = d^\sigma y^\sigma(1) + (1-d^\sigma)y^\sigma(0)) && (52.11a) \\ &= \mathbb{1}(y^\sigma = y^\sigma(d^\sigma)) && (52.11b) \end{aligned}$$

Eq.(52.11) is often referred to as the **SUTVA assumption**.

If we sum over the nodes $\underline{y}(0)$ and $\underline{y}(1)$ of this bnet, we should get the bnet G. This is easy to check. Indeed,

$$P(y^\sigma|d^\sigma, x^\sigma) = \sum_{y^\sigma(0)} \sum_{y^\sigma(1)} \mathbb{1}(y^\sigma = y^\sigma(d^\sigma)) P(y^\sigma(0)|d^\sigma, x^\sigma) P(y^\sigma(1)|d^\sigma, x^\sigma) \tag{52.12}$$

$$= \begin{cases} P_{\underline{y}(0)|\underline{d},\underline{x}}(y^\sigma|d^\sigma, x^\sigma) & \text{if } d^\sigma = 0 \\ P_{\underline{y}(1)|\underline{d},\underline{x}}(y^\sigma|d^\sigma, x^\sigma) & \text{if } d^\sigma = 1 \end{cases} . \tag{52.13}$$

Henceforth, we will refer to the case where the question mark arrow is included as the general case, and to the case when it's excluded as the **weak-d limit**. Henceforth, we will first present results for the general case, and then describe how those results change for the weak-d limit. Rubinologists always assume the weak-d limit, but we find that with little effort, we can derive many results for general case, and then compare those results to their weak-d limit. I find such comparisons instructive.

Note that in the general case, $P(\underline{y}(c) = y|\underline{d} = d, x)$ for $c, d \in \{0,1\}$ are four different probability distributions, and that $P(\underline{y} = y|d = d, x)$ is defined in terms of two of them, the so called **factual distributions** with $c = d$. By measuring $\underline{y}$, we can't access the other 2 probability distributions, the so called **counter-factual distributions** with $c \neq d$.

In the weak-d limit, $P(\underline{y}(c) = y|\underline{d} = d, x) = P(\underline{y}(c) = y|x)$ are two probability distributions, and they both can be accessed by measuring $\underline{y}$.

52.3 Expected Values of treatment outcome y^σ

It is convenient to define the following expected values of $\underline{y}^\sigma$ in terms of the TPMs of bnet G_+:

$$\mathcal{Y}_{c|d,x} = E_{\sigma|d,x}[\underline{y}^\sigma(c)] \to E_{\underline{y}(c)|d,x}[\underline{y}(c)] = \sum_y yP(\underline{y}(c) = y|\underline{d} = d, x) \tag{52.14}$$

$$\mathcal{Y}_{c|d} = E_{\sigma|d}[\underline{y}^\sigma(c)] \to E_{\underline{y}(c)|d}[\underline{y}(c)] = \sum_x \mathcal{Y}_{c|d,x}P(x|d) \tag{52.15}$$

$$\mathcal{Y}_{c|x} = E_{\sigma|x}[\underline{y}^\sigma(c)] \to E_{\underline{y}(c)|x}[\underline{y}(c)] = \sum_d \mathcal{Y}_{c|d,x}P(d|x) \tag{52.16}$$

$$\mathcal{Y}_c = E_\sigma[\underline{y}^\sigma(c)] \to E_{\underline{y}(c)}[\underline{y}(c)] = \sum_{x,d} \mathcal{Y}_{c|d,x}P(x,d) \tag{52.17}$$

Note that in the weak-d limit,

$$\mathcal{Y}_c = \sum_x \mathcal{Y}_{c|d,x}P(x) \ . \tag{52.18}$$

Note also that in the weak-d limit, $\mathcal{Y}_{c|d,x}$ is independent of d, but $\mathcal{Y}_{c|d}$ can depend on d if $P(x|d)$ depends on d.

$\mathcal{Y}_{0|0}, \mathcal{Y}_{1|1}$ are said to be **factual** (indicating compliant patients) whereas $\mathcal{Y}_{0|1}, \mathcal{Y}_{1|0}$ are said to be **counterfactual** (indicating non-compliant patients).

Also let

$$\mathcal{Y}_{|d,x} = E_{\sigma|d,x}[\underline{y}^\sigma] \to E_{\underline{y}|d,x}[\underline{y}] = \sum_y yP(\underline{y} = y|\underline{d} = d, x) \tag{52.19}$$

$$\mathcal{Y}_{|d} = E_{\sigma|d}[\underline{y}^\sigma] \to E_{\underline{y}|d}[\underline{y}] = \sum_x \mathcal{Y}_{|d,x}P(x|d) \tag{52.20}$$

$$\mathcal{Y}_{|x} = E_{\sigma|x}[\underline{y}^\sigma] \to E_{\underline{y}|x}[\underline{y}] = \sum_d \mathcal{Y}_{|d,x}P(d|x) \tag{52.21}$$

$$\mathcal{Y} = E_\sigma[\underline{y}^\sigma] \to E_{\underline{y}}[\underline{y}] = \sum_d \mathcal{Y}_{|d,x}P(d,x) \tag{52.22}$$

In the weak-d limit, $\mathcal{Y}_{|d,x}$ is independent of d, but $\mathcal{Y}_{|d}$ can still depend if $P(x|d)$ depends on d, then $\mathcal{Y}_{|d}$ depends on d too.

52.4 Translation Dictionary

Table 52.2 gives a dictionary for translating from the standard PO notation of Ref.[6] to our notation.

In standard PO notation	In our notation
i, individual (i.e., unit, sample) index	σ
$D_i = d_i$, treatment dose	$\underline{d}^\sigma = d^\sigma$
$Y_i = y_i$, treatment outcome	$\underline{y}^\sigma = y^\sigma$
$X_i = x_i$, treatment confounders	$\underline{x}^\sigma = x^\sigma$
$E[Y_i(c)]$	$E_\sigma[\underline{y}^\sigma(c)] = \mathcal{Y}_c$
$E[Y_i(c)\|D_i = d]$	$E_{\sigma\|d}[\underline{y}^\sigma(c)] = \mathcal{Y}_{c\|d}$
$E[Y_i(c)\|D_i = d, X_i = x]$	$E_{\sigma\|d,x}[\underline{y}^\sigma(c)] = \mathcal{Y}_{c\|d,x}$
$E[Y_i]$	$E_\sigma[\underline{y}^\sigma] = \mathcal{Y}$
$E[Y_i\|D_i = d]$	$E_{\sigma\|d}[\underline{y}^\sigma] = \mathcal{Y}_{\|d}$
$E[Y_i\|D_i = d, X_i = x]$	$E_{\sigma\|d,x}[\underline{y}^\sigma] = \mathcal{Y}_{\|d,x}$

Table 52.2: Dictionary for translating from standard PO notation of Ref.[6] to our notation. $c, d \in \{0, 1\}$.

52.5 $\mathcal{Y}_{|d,x} = \mathcal{Y}_{d|d,x}$ (SUTVA)

Claim 60 [2]

$$\mathcal{Y}_{|d,x} = \mathcal{Y}_{d|d,x} \tag{52.23}$$

$$\mathcal{Y}_{|d} = \mathcal{Y}_{d|d} \tag{52.24}$$

proof:

$$\begin{aligned} \mathcal{Y}_{|d,x} &= \sum_y y P(\underbrace{\underline{y}}_{=\underline{y}(d) \text{ by Eq.(52.11)}} = y|d,x) && (52.25) \\ &= \mathcal{Y}_{d|d,x} \ . && (52.26) \end{aligned}$$

Applying $\sum_x P(x|d)$ to both sides of Eq.(52.23) gives Eq.(52.24).
QED

[2] In the standard PO notation, this is the frequently used identity

$$E[Y|D = d, x] = E[Y(d)|D = d, x]$$

.

52.6 Conditional Independence Assumption (CIA)

The **Conditional Independence Assumption** (CIA) is said to hold if

$$(\underline{y}^\sigma(0), \underline{y}^\sigma(1), \underline{y}^\sigma) \perp_P \underline{d}^\sigma | \underline{x}^\sigma \ . \tag{52.27}$$

This is satisfied by G_+ in the weak-d limit. To prove this, check that

$$(\underline{y}^\sigma(0), \underline{y}^\sigma(1), \underline{y}^\sigma) \perp_{G_+} \underline{d}^\sigma | \underline{x}^\sigma \tag{52.28}$$

and then invoke the d-separation theorem (see Chapter 17).

I think CIA only makes sense if the individuals are treatment blind (i.e., have no knowledge of whether they are in the treated or control groups.) Otherwise, that extra knowledge becomes a confounder not being included in x.

A **Randomized Controlled Trial (RCT)** is defined to satisfy Eq.(52.27) without the $\underline{x}^\sigma$ conditioning; i.e., it satisfies

$$(\underline{y}^\sigma(0), \underline{y}^\sigma(1), \underline{y}^\sigma) \perp_P \underline{d}^\sigma \ . \tag{52.29}$$

This means that in a RCT, the arrow from $\underline{x}^\sigma$ to $\underline{d}^\sigma$ in G_+ is omitted.

Note that if we assume both CIA (i.e., weak-d limit) and SUTVA, we get

$$\begin{aligned}
\mathcal{Y}_{|d,x} &= E_{\sigma|d,x}[\underline{y}^\sigma] && (52.30a)\\
&= E_\sigma[\underline{y}^\sigma | \underline{d}^\sigma = d, \underline{x}^\sigma = x] && (52.30b)\\
&= E_\sigma[\underline{y}^\sigma(d) | \underline{d}^\sigma = d, \underline{x}^\sigma = x] \text{ (by SUTVA)} && (52.30c)\\
&= E_\sigma[\underline{y}^\sigma(d) | \underline{x}^\sigma = x] \text{ (by CIA)} && (52.30d)\\
&= E_{\sigma|x}[\underline{y}^\sigma(d)] && (52.30e)\\
&= \mathcal{Y}_{d|x} \ . && (52.30f)
\end{aligned}$$

In a RCT, Eq.(52.30f) is valid without the x conditioning.

52.7 Treatment Effects

A **treatment effect** is a a difference of two $\mathcal{Y}_{c|d}$. It is convenient to define the following treatment effects. See Figs.52.4 and 52.5.

- average treatment effect (ATE).

$$ATE = \mathcal{Y}_1 - \mathcal{Y}_0 = \delta \tag{52.31}$$

- average treatment effect of the treated (ATT)

$$ATT = \mathcal{Y}_{1|1} - \mathcal{Y}_{0|1} \tag{52.32}$$

Figure 52.4: Different treatment effects. A treatment effect is a difference of two $\mathcal{Y}_{c|d}$.

Figure 52.5: Alternative representation of the same information that is contained in Fig.52.4.

- average treatment effect of the untreated (ATU)

$$ATU = \mathcal{Y}_{1|0} - \mathcal{Y}_{0|0} \tag{52.33}$$

- selection bias (SB)

$$SB = \mathcal{Y}_{0|1} - \mathcal{Y}_{0|0} \tag{52.34}$$

- simple difference in outcomes (SDO)

$$SDO = \mathcal{Y}_{1|1} - \mathcal{Y}_{0|0} \tag{52.35}$$

Let

$$\pi_d = P(\underline{d} = d) \tag{52.36}$$

for $d \in \{0, 1\}$.

Note that there exist some linear constraints between these treatment effects.

$$\underbrace{\mathcal{Y}_1 - \mathcal{Y}_0}_{ATE} = \underbrace{\mathcal{Y}_{1|1}\pi_1 + \mathcal{Y}_{1|0}\pi_0}_{\mathcal{Y}_1} - \underbrace{(\mathcal{Y}_{0|1}\pi_1 + \mathcal{Y}_{0|0}\pi_0)}_{\mathcal{Y}_0} \tag{52.37}$$

$$= \underbrace{(\mathcal{Y}_{1|1} - \mathcal{Y}_{0|1})}_{ATT}\pi_1 + \underbrace{(\mathcal{Y}_{1|0} - \mathcal{Y}_{0|0})}_{ATU}\pi_0 \tag{52.38}$$

$$\underbrace{\mathcal{Y}_{1|1} - \mathcal{Y}_{0|0}}_{SDO} = \underbrace{(\mathcal{Y}_{1|1} - \mathcal{Y}_{0|1})}_{ATT} + \underbrace{\mathcal{Y}_{0|1} - \mathcal{Y}_{0|0}}_{SB} \tag{52.39}$$

$$\begin{aligned}\underbrace{\mathcal{Y}_{1|1} - \mathcal{Y}_{0|0}}_{SDO} = & \underbrace{(\mathcal{Y}_{1|1} - \mathcal{Y}_{0|1})\pi_1 + (\mathcal{Y}_{1|0} - \mathcal{Y}_{0|0})\pi_0}_{ATE} \\ & + \underbrace{\mathcal{Y}_{0|1} - \mathcal{Y}_{0|0}}_{SB} \\ & + \underbrace{(\mathcal{Y}_{1|1} - \mathcal{Y}_{0|1})}_{ATT}\pi_0 \\ & - \underbrace{(\mathcal{Y}_{1|0} - \mathcal{Y}_{0|0})}_{ATU}\pi_0 \end{aligned} \tag{52.40}$$

By virtue of Eq.(52.38),

$$ATT = ATU \implies ATT = ATU = ATE \tag{52.41}$$

and

$$ATE = 0 \iff \frac{ATU}{ATT} = -\left(\frac{\pi_1}{\pi_0}\right) . \tag{52.42}$$

Whenever $ATT = ATU$, we will say there is **T-U symmetry**.

In general, $SDO = ATT + SB$, but if there is T-U symmetry, then $SDO = ATE + SB$.

If there is T-U symmetry and zero bias $SB = 0$, then $SDO = ATE = ATT = ATU$.

If there is a null result for a RCT (i.e., $ATE = 0$), T-U symmetry and zero bias $SB = 0$, then $SDO = ATE = ATT = ATU = 0$.

Let

$$\mathcal{Y}_{c,d|x} = \mathcal{Y}_{c|d,x}P(d|x) \tag{52.43}$$

For each $\mathcal{E} \in \{ATE, ATT, ATU, SB, SDO\}$, we can define its restriction $\mathcal{E}_x$ to a fixed stratum x by replacing each $\mathcal{Y}_{c|d}$ with $\mathcal{Y}_{c,d|x}$. For example,

$$ATT = \mathcal{Y}_{1|1} - \mathcal{Y}_{0|1} \text{ so } ATT_x = \mathcal{Y}_{1,1|x} - \mathcal{Y}_{0,1|x} . \tag{52.44}$$

We can calculate $\mathcal{E}$ from $\mathcal{E}_x$ using

$$\mathcal{E} = E_x[\mathcal{E}_x] = \sum_x P(x)\mathcal{E}_x \ . \tag{52.45}$$

52.8 Insights into what makes treatment effects equal and $\mathcal{Y}_{1|0} = \mathcal{Y}_1$

Figure 52.6: Figure 52.4 with added information about probability distributions used to obtain each expected value $\mathcal{Y}_{c|d}$.

1. Is it possible for $SDO = 0$ but $ATE \neq 0$ or vice versa, and what is going on when this is true?

2. What is going on when two treatment effects are equal; for instance, when $ATT = ATU$?

3. When is $\mathcal{Y}_{1|0} = \mathcal{Y}_1$, and what is going on when this is true?

Fig.52.6 gives some intuition about what is going on when any of these things happen.

Recall that each expected value $\mathcal{Y}_{c|d}$ is associated with a probability distribution $P_{\underline{y}(c)|\underline{d},x}$.

$$\mathcal{Y}_{c|d} = \sum_y y \underbrace{\sum_x P_{\underline{y}(c)|\underline{d},x}(y|d,x)P(x|d)}_{P_{\underline{y}(c)|\underline{d}}(y|d)} \tag{52.46}$$

for $c, d \in \{0, 1\}$. Fig.52.6 reminds us of which P is used to generate each $\mathcal{Y}$. From this figure, we see that

1. A sufficient condition for $SDO = 0$ is that $P_{\underline{y}(1)|\underline{d}=1} = P_{\underline{y}(0)|\underline{d}=0}$. A sufficient condition for $ATE = 0$ is that $P_{\underline{y}(1)|x} = P_{\underline{y}(0)|x}$.

2. A sufficient condition for $ATT = ATU$ is that $P_{\underline{y}(1)|\underline{d}=1,x} - P_{\underline{y}(0)|\underline{d}=1,x}$ equals $P_{\underline{y}(1)|\underline{d}=0,x} - P_{\underline{y}(0)|\underline{d}=0,x}$.

3. A sufficient condition for $\mathcal{Y}_{1|0} = \mathcal{Y}_1$ is that $P_{\underline{y}(1)|\underline{d}=0} = P_{\underline{y}(1)}$. Note that the CIA implies that $P_{\underline{y}(1)|\underline{d}=0,x} = P_{\underline{y}(1)|x}$ always, but this does not imply that $P_{\underline{y}(1)|\underline{d}=0} = P_{\underline{y}(1)}$.

52.9 G_{do+} bnet

Figure 52.7: Bnet $G_{do} = \mathcal{D}_{\underline{d}^\sigma}(\tilde{d}^\sigma)G$ is obtained by applying the do operator to node $\underline{d}^\sigma$ of bnet G. Bnet G_{do+} is obtained by adding a prior probability distribution $P(\tilde{d}^\sigma)$ to node $\mathcal{D}\underline{d}^\sigma$ of bnet G_{do}.

Fig.52.7 shows how bnet G_{do} is obtained by applying the do operator to bnet G, and how bnet G_{do+} is obtained by adding a prior probability distribution to one of the nodes of G_{do}. In bnet G_{do}, node $\underline{d}^\sigma$ has been stripped of all outside influences and fixed to a specific state $\tilde{d}^\sigma$. This is what a RCT does.

The TPMs, printed in blue, for the bnets G_{do} and G_{do+}, are as follows. Note that the TPMs for bnets G_{do} and G_{do+} are defined in terms of the TPMs of bnet G.

$$P(x^\sigma) = P_{\underline{x}}(x^\sigma) \tag{52.47}$$

$$P_{\mathcal{D}\underline{d}}(\tilde{d}) = \sum_x P_{\underline{d}|\underline{x}}(\tilde{d}|x)P_{\underline{x}}(x) \tag{52.48}$$

$$P(\tilde{d}^\sigma) = \begin{cases} \delta(\tilde{d}^\sigma, (\tilde{d}^\sigma)') & \text{for } G_{do} \\ P_{\mathcal{D}\underline{d}}(\tilde{d}^\sigma) & \text{for } G_{do+} \end{cases} \tag{52.49}$$

$$P(y^\sigma|\tilde{d}^\sigma, x^\sigma) = P_{\underline{y}|\underline{d},\underline{x}}(y^\sigma|\tilde{d}^\sigma, x^\sigma) \tag{52.50}$$

Note that in G_{do},

$$P(\underline{y} = y|\mathcal{D}\underline{d} = d, \underline{x} = x) = P(y|\underline{d} = d, x) \tag{52.51}$$

because, by the d-separation theorem, when we condition on the confounder $\underline{x}$, we block information from being transmitted from $\underline{d}$ to $\underline{y}$ through $\underline{x}$, and this is equivalent to amputating the arrow $\underline{x} \to \underline{d}$.

Using Eq.(52.51), we get

$$\begin{aligned} P(\underline{y} = y|\mathcal{D}\underline{d} = d) &= \sum_x P(\underline{y} = y|\mathcal{D}\underline{d} = d, x)P(x|\mathcal{D}\underline{d} = d) && (52.52) \\ &= \sum_x P(y|d, x)P(x) && (52.53) \end{aligned}$$

Eq.(52.53) is called the **backdoor adjustment formula**. It allows us to express a probability with a do operator in its definition in terms of probabilities without do operators.

52.10 $ACE = ATE$

Define the Average Causal Effect (ACE) by

$$\begin{aligned} ACE &= \sum_y y[P(y|\mathcal{D}\underline{d} = 1) - P(y|\mathcal{D}\underline{d} = 0)] && (52.54) \\ &= \sum_x P(x) \sum_y y[P(y|\underline{d} = 1, x) - P(y|\underline{d} = 0, x)] \text{ . (by Eq.(52.53)} && (52.55) \end{aligned}$$

Claim 61 *If we assume both SUTVA and CIA (i.e., weak-d limit), then*

$$ACE = ATE \tag{52.56}$$

proof:

$$\begin{aligned} ACE &= \sum_x P(x) \sum_y y[P(\underline{y} = y|\underline{d} = 1, x) - P(\underline{y} = y|\underline{d} = 0, x)] && (52.57) \\ &= \sum_x P(x)[\mathcal{Y}_{1|1,x} - \mathcal{Y}_{0|0,x}] \text{ (by SUTVA)} && (52.58) \\ &= \sum_x P(x)[\mathcal{Y}_{1|x} - \mathcal{Y}_{0|x}] \text{ (by CIA)} && (52.59) \\ &= \mathcal{Y}_1 - \mathcal{Y}_0 && (52.60) \\ &= ATE && (52.61) \end{aligned}$$

QED

We will say there is a **null result in a RCT** when $ACE = 0$. By the previous claim, this is true iff $ATE = 0$ (assuming weak-d limit).

52.11 Good, Bad Controls

The bnet G_+ of Fig.52.3, —the cornerstone of Rubin's PO theory— is limited in scope and is easily misapplied, leading to incorrect results. The problem is some features that are available and coud be conditioned on shouldn't because they would introduce spurious contributions to ATE. Such features are called "bad controls".

Examples:

Figure 52.8: In this bnet, $\underline{x}$ is a bad control (i.e., should not be conditioned on). Nodes $\underline{h}_1$ and $\underline{h}_2$ are hidden and therefore cannot be conditioned on.

1. Consider the bnet Fig.52.8, which Pearl calls M-bias, because it looks like an M. In that figure, $\underline{x}$ is a "bad control" because calculating ATE by conditioning on it, and using the formula $ATE = \sum_x P(x)[\mathcal{Y}_{1|x} - \mathcal{Y}_{0|x}]$, yields a value of ATE that is different from ACE. This value for ATE is unacceptable because it does not give the result of a RCT whereas ACE defined in terms of do operators always does. The reason[3] $ATE \neq ACE$ for this figure is that in it, $\underline{x}$ is a collider node, and conditioning on it allows rather than prevents information to flow from $\underline{d}$ to $\underline{y}$ via the path $\underline{d} - \underline{h}_1 - \underline{x} - \underline{h}_2 - \underline{y}$.

2. In Fig.52.9, node $\underline{c}_1$ is a good control and nodes $\underline{c}_2, \underline{c}_3, \underline{c}_4$ are bad ones.

   Conditioning on $\underline{c}_1$ blocks path $\underline{d} - \underline{c}_1 - \underline{y}$, good

   Conditioning on $\underline{c}_2$ opens path $\underline{d} - \underline{c}_2 - \underline{y}$, bad

   Conditioning on $\underline{c}_3$ blocks path $\underline{d} - \underline{c}_3 - \underline{y}$, bad

   Conditioning on $\underline{c}_4$ opens path $\underline{d} - \underline{c}_4 - \underline{y}$, bad

[3] We are using here arguments based on the d-separation theorem which is discussed in Chapter 17.

Figure 52.9: In this bnet, node $\underline{c}_1$ is a good control and nodes $\underline{c}_2, \underline{c}_3, \underline{c}_4$ are bad ones.

Pearl et al have a paper (Ref.[5]) that I highly recommend that gives dozens of examples of good, neutral and bad controls in an ACE calculation.

52.12 PO Confounder Sensitivity Analysis

Figure 52.10: LDEN bnet used to do PO confounder sensitivity analysis. Node $\underline{u}$ is an unobserved confounder.

Consider the LDEN bnet of Fig.52.10. whose structure equations, printed in blue, are as follows:

$$\underline{d} = \alpha \underline{x} + \alpha' \underline{u} \tag{52.62}$$

$$\underline{y} = \beta \underline{x} + \beta' \underline{u} + \delta \underline{d} \tag{52.63}$$

Therefore,

$$\underline{u} = \frac{1}{\alpha'}(\underline{d} - \alpha \underline{x}) \tag{52.64}$$

$$\underline{y} = \left(\beta - \frac{\alpha\beta'}{\alpha'}\right) \underline{x} + \left(\delta + \frac{\beta'}{\alpha'}\right) \underline{d} \tag{52.65}$$

$$\mathcal{Y}_{d|x,u} = E[\underline{y}(d)|x,u] = E[\underline{y}|d,x,u] = \beta x + \beta' u + \delta d \tag{52.66}$$

$$\mathcal{Y}_{d|x} = E[\underline{y}(d)|x] = E[\underline{y}|d,x] = \left(\beta - \frac{\alpha\beta'}{\alpha'}\right) x + \left(\delta + \frac{\beta'}{\alpha'}\right) d \tag{52.67}$$

$$ATE = E_x[ATE_x] = E_x[\mathcal{Y}_{1|x} - \mathcal{Y}_{0|x}] = \delta + \frac{\beta'}{\alpha'} \tag{52.68}$$

$$ATE|_{\beta'=0} = \delta = \frac{d\underline{y}}{d\underline{d}} \tag{52.69}$$

$$ATE - ATE|_{\beta'=0} = \frac{\beta'}{\alpha'} = \frac{\frac{d\underline{y}}{d\underline{u}}}{\frac{d\underline{d}}{d\underline{u}}} \tag{52.70}$$

52.13 (SDO, ATE) space

If we substitute $y^\sigma \to y^\sigma(d^\sigma)$ and $y^{m(\sigma)} \to y^\sigma(1-d^\sigma)$ into the estimator Eq.(52.83) for ATE and the estimator Eq.(52.89) for SDO, we get

$$\begin{aligned} \widehat{ATE}_x &= \frac{1}{N_x} \sum_{\sigma \in A_x} (2d^\sigma - 1)[y^\sigma(d^\sigma) - y^\sigma(1-d^\sigma)] && (52.71) \\ &= \frac{1}{N_x} \sum_{\sigma \in A_x} [y^\sigma(1) - y^\sigma(0)] && (52.72) \end{aligned}$$

and

$$\begin{aligned} \widehat{SDO}_x &= \frac{1}{N_{1,x}} \sum_{\sigma \in A_x} d^\sigma y^\sigma(d^\sigma) - \frac{1}{N_{0,x}} \sum_{\sigma \in A_x} (1-d^\sigma) y^\sigma(d^\sigma) && (52.73) \\ &= \frac{1}{N_{1,x}} \sum_{\sigma \in A_{1,x}} y^\sigma(1) - \frac{1}{N_{0,x}} \sum_{\sigma \in A_{0,x}} y^\sigma(0) \ . && (52.74) \end{aligned}$$

Recall that $\hat{\mathcal{E}} = E_x[\hat{\mathcal{E}}_x] = \sum_x \frac{N_x}{N} \hat{\mathcal{E}}_x$ for $\mathcal{E} \in \{ATE, SDO\}$.

Recall also that $ACE = ATE = 0$ is the null result in a RCT.

Suppose that the treatment outcome y^σ has only two possible values, 0 and 1. Then, $-1 \leq ATE \leq 1$ and $-1 \leq SDO \leq 1$. But does $ATE = 0$ imply $SDO = 0$ or vice versa? Next, we answer that question and more by finding the region of accessibility in the (SDO, ATE) plane, assuming $y^\sigma \in \{0, 1\}$.

σ	d^σ	$y^\sigma(0)$	$y^\sigma(1)$
1	0	1	0
2	0	1	0
3	0	1	0
4	1	1	0
5	1	1	0
6	1	1	0

(a) $ATE = -1$ ($SDO = -1$) point A

σ	d^σ	$y^\sigma(0)$	$y^\sigma(1)$
1	0	0	1
2	0	0	1
3	0	0	1
4	1	0	0
5	1	0	0
6	1	0	0

(b) $ATE = \frac{1}{2}$ ($SDO = 0$) point B

σ	d^σ	$y^\sigma(0)$	$y^\sigma(1)$
1	0	0	1
2	0	0	1
3	0	0	1
4	1	0	1
5	1	0	1
6	1	0	1

(c) $ATE = 1$ ($SDO = 1$) point C

Figure 52.11: Examples of PO datasets. Exploring ATE extremes.

σ	d^σ	$y^\sigma(0)$	$y^\sigma(1)$
1	0	1	1
2	0	1	1
3	0	1	1
4	1	0	0
5	1	0	0
6	1	0	0

(a) $SDO = -1$ ($ATE = 0$) point D

σ	d^σ	$y^\sigma(0)$	$y^\sigma(1)$
1	0	1	0
2	0	1	0
3	0	1	0
4	1	1	1
5	1	1	1
6	1	1	1

(b) $SDO = 0$ ($ATE = -\frac{1}{2}$) point E

σ	d^σ	$y^\sigma(0)$	$y^\sigma(1)$
1	0	0	0
2	0	0	0
3	0	0	0
4	1	1	1
5	1	1	1
6	1	1	1

(c) $SDO = 1$ ($ATE = 0$) point F

Figure 52.12: Examples of PO datasets. Exploring SDO extremes.

Figure 52.13: Green parallelogram is accessible region in (SDO, ATE) plane, assuming $y^\sigma \in \{0, 1\}$. Each of the six points A, B, ... F corresponds to one of the six tables in Figs. 52.11 and 52.12. Segment DF corresponds to the null result in a RCT.

52.14 Strata-Matching

For a situation described by the bnet G_+ in the weak-d limit, we can match *similar* individuals to fill the blank cells of Table 52.1. By "similar", we mean that they have the same or almost the same value of $\underline{x}^\sigma$.

The reason the weak-d limit is required is because it implies that $P(y(c)|d = 0, x) = P(y(c)|d = 1, x)$ for $c \in \{0, 1\}$, Hence, we can sample from a known factual $(c = d)$ distribution to fill the missing data in the unknown counterfactual $(c \neq d)$ distribution.

52.14.1 Exact strata-matching

Estimates of Treatment Effects

For $d \in \{0, 1\}$ and all strata x, define the sets of individuals $A_{d,x} = \{\sigma : d^\sigma = d, x^\sigma = x\}$, $A_x = A_{0,x} \cup A_{1,x}$ and $A = \cup_x A_x$. Let $N_{d,x} = |A_{d,x}|$, $N_x = |A_x|$ and $N = |A|$.

In an exact strata-matching, we match each individual with $d^\sigma = d, x^\sigma = x$ with exactly one individual with $d^\sigma = 1 - d, x^\sigma = x$. Define a map $m : A \to A$ such that, for each x, and for $d \in \{0, 1\}$, if $\sigma \in A_{d,x}$, then $m(\sigma) \in A_{1-d,x}$ This assumes $A_{0,x}$ and $A_{1,x}$ are non-empty for all x. The purpose of map $m()$ is to fill in the missing data in the PO dataset. See Fig.52.3 for a pictorial representation of this.

	$y^\sigma(0)$	$y^\sigma(1)$
$d^\sigma = 0$	y^σ	$y^{m(\sigma)}$
$d^\sigma = 1$	$y^{m(\sigma)}$	y^σ

Table 52.3: Illustration of the purpose of the map $m()$. Note that $y^\sigma = y^\sigma(d^\sigma)$ and $y^{m(\sigma)} = y^\sigma(1 - d^\sigma)$.

Note that

$$\sum_{\sigma \in A_x} \frac{d^\sigma}{N_{1,x}} = \sum_{\sigma \in A_{1,x}} \frac{1}{N_{1,x}} = 1 \ . \tag{52.75}$$

Thus

$$\sum_{\sigma \in A_x} \frac{d^\sigma}{N_{1,x}} y^\sigma = E_{\sigma|d=1,x}[y^\sigma] = \mathcal{Y}_{1|1,x} \tag{52.76}$$

Table 52.4 gives estimates of $\mathcal{Y}_{c|d,x}$

Recall that

$$\mathcal{Y}_{c,d|x} = \mathcal{Y}_{c|d,x} P(d|x) \tag{52.77}$$

Hence,

	$y^\sigma(0)$	$y^\sigma(1)$
$d^\sigma = 0$	$\frac{1}{N_{0,x}} \sum_{\sigma \in A_x} (1 - d^\sigma) y^\sigma = \mathcal{Y}_{0\|0,x}$	$\frac{1}{N_{0,x}} \sum_{\sigma \in A_x} (1 - d^\sigma) y^{m(\sigma)} = \mathcal{Y}_{1\|0,x}$
$d^\sigma = 1$	$\frac{1}{N_{1,x}} \sum_{\sigma \in A_x} d^\sigma y^{m(\sigma)} = \mathcal{Y}_{0\|1,x}$	$\frac{1}{N_{1,x}} \sum_{\sigma \in A_x} d^\sigma y^\sigma = \mathcal{Y}_{1\|1,x}$

Table 52.4: Estimates of $\mathcal{Y}_{c|d,x}$.

	$y^\sigma(0)$	$y^\sigma(1)$
$d^\sigma = 0$	$\frac{1}{N_x} \sum_{\sigma \in A_x} (1 - d^\sigma) y^\sigma = \mathcal{Y}_{0,0\|x}$	$\frac{1}{N_x} \sum_{\sigma \in A_x} (1 - d^\sigma) y^{m(\sigma)} = \mathcal{Y}_{1,0\|x}$
$d^\sigma = 1$	$\frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma y^{m(\sigma)} = \mathcal{Y}_{0,1\|x}$	$\frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma y^\sigma = \mathcal{Y}_{1,1\|x}$

Table 52.5: Estimates of $\mathcal{Y}_{c,d|x}$.

$$\mathcal{Y}_{c,d|x} = (N_{d,x}\mathcal{Y}_{c|d,x})\frac{P(d|x)}{N_{d,x}} \tag{52.78}$$

$$= (N_{d,x}\mathcal{Y}_{c|d,x})\frac{1}{N_x} \tag{52.79}$$

Table 52.5 gives estimates of $\mathcal{Y}_{c,d|x}$

The treatment effects $\mathcal{E} \in \{ATE, ATT, ATU, SB, SDO\}$ can be estimated from the data via the following estimators.

$$\widehat{ATE}_x = \overbrace{\mathcal{Y}_{1|1,x}P(1|x) + \mathcal{Y}_{1|0,x}P(0|x)}^{\mathcal{Y}_{1|x}} - \overbrace{(\mathcal{Y}_{0|1,x}P(1|x) + \mathcal{Y}_{0|0,x}P(0|x))}^{\mathcal{Y}_{0|x}} \tag{52.80}$$

$$= \frac{1}{N_x}[\widehat{ATT}_x N_{1,x} + \widehat{ATU}_x N_{0,x}] \tag{52.81}$$

$$= \frac{1}{N_x}\left[\sum_{\sigma \in A_x} d^\sigma[y^\sigma - y^{m(\sigma)}] + \sum_{\sigma \in A_x}(1 - d^\sigma)[y^{m(\sigma)} - y^\sigma]\right] \tag{52.82}$$

$$= \frac{1}{N_x}\sum_{\sigma \in A_x}(2d^\sigma - 1)[y^\sigma - y^{m(\sigma)}] \tag{52.83}$$

$$\widehat{ATT}_x = \overbrace{\frac{1}{N_x}\sum_{\sigma \in A_x} d^\sigma y^\sigma}^{\mathcal{Y}_{1,1|x}} - \overbrace{\frac{1}{N_x}\sum_{\sigma \in A_x} d^\sigma y^{m(\sigma)}}^{\mathcal{Y}_{0,1|x}} \tag{52.84}$$

$$= \frac{1}{N_x}\sum_{\sigma \in A_x} d^\sigma[y^\sigma - y^{m(\sigma)}] \tag{52.85}$$

$$\widehat{ATU}_x = \overbrace{\frac{1}{N_x}\sum_{\sigma\in A_x}(1-d^\sigma)y^{m(\sigma)}}^{\mathcal{Y}_{1,0|x}} - \overbrace{\frac{1}{N_x}\sum_{\sigma\in A_x}(1-d^\sigma)y^\sigma}^{\mathcal{Y}_{0,0|x}} \tag{52.86}$$

$$= \frac{1}{N_x}\sum_{\sigma\in A_x}(1-d^\sigma)[y^{m(\sigma)} - y^\sigma] \tag{52.87}$$

$$\widehat{SB}_x = \overbrace{\frac{1}{N_x}\sum_{\sigma\in A_x} d^\sigma y^{m(\sigma)}}^{\mathcal{Y}_{0,1|x}} - \overbrace{\frac{1}{N_x}\sum_{\sigma\in A_x}(1-d^\sigma)y^\sigma}^{\mathcal{Y}_{0,0|x}} \tag{52.88}$$

$$\widehat{SDO}_x = \overbrace{\frac{1}{N_x}\sum_{\sigma\in A_x} d^\sigma y^\sigma}^{\mathcal{Y}_{1,1|x}} - \overbrace{\frac{1}{N_x}\sum_{\sigma\in A_x}(1-d^\sigma)y^\sigma}^{\mathcal{Y}_{0,0|x}} \tag{52.89}$$

Suppose we do linear regression to fit a hyperplane $y(x)$ to the dataset set $\{(x^\sigma, y^\sigma) : \sigma\}$, and then we calculate $\frac{dy}{d\underline{d}} = \delta$. Out of all the treatment effects, this δ is probably (?) closest to $ACE = ATE$. Note also that the linear regression method of estimating δ does imputation (guesses missing values) by doing a linear fit. One can also use machine learning to do a non-linear fit. In contrast, the estimators of treatment effects presented in this section do imputation by non-linear strata-matching.

Example, estimation of treatment effects

For $\sigma \in \{1, 2, \ldots, 10\}$, define

$$m(\sigma) = \begin{cases} \sigma + 5 & \text{if } \sigma \leq 5 \\ \sigma - 5 & \text{if } \sigma > 5 \end{cases} \tag{52.90}$$

Let $N(\mathcal{S})$ be the number of individuals σ that satisfy condition $\mathcal{S}$. For example, $N(\underline{d^\sigma} = d)$ is the number of individuals such that $\underline{d^\sigma} = d$.

$$N_1 = N(d^\sigma = 1) = 5 \tag{52.91}$$

$$N_0 = N(d^\sigma = 0) = 5 \tag{52.92}$$

$$N = N_0 + N_1 = 10 \tag{52.93}$$

$$\mathcal{Y}_{1|1} = \frac{1}{N_1}\sum_\sigma d^\sigma y^\sigma = \frac{4}{5} \tag{52.94}$$

σ	d^σ	y^σ	$d^\sigma y^\sigma$	$(1-d^\sigma)y^\sigma$	$d^\sigma y^{m(\sigma)}$	$(1-d^\sigma)y^{m(\sigma)}$
1	0	0	0	0	0	0
2	0	0	0	0	0	1
3	0	1	0	1	0	1
4	0	1	0	1	0	1
5	0	1	0	1	0	1
6	1	0	0	0	0	0
7	1	1	1	0	0	0
8	1	1	1	0	1	0
9	1	1	1	0	1	0
10	1	1	1	0	1	0

Table 52.6: Estimators of treatment effects are calculated for this example.

$N(d,y)$	$y=0$	$y=1$
$d=0$	2	3
$d=1$	1	4

Table 52.7: $N(\underline{d^\sigma}=d, \underline{y}^\sigma=y)$ for the data in Table 52.6.

$$\mathcal{Y}_{0|0} = \frac{1}{N_0}\sum_\sigma (1-d^\sigma)y^\sigma = \frac{3}{5} \tag{52.95}$$

$$\mathcal{Y}_{0|1} = \frac{1}{N_1}\sum_\sigma d^\sigma y^{m(\sigma)} = \frac{3}{5} \tag{52.96}$$

$$\mathcal{Y}_{1|0} = \frac{1}{N_0}\sum_\sigma (1-d^\sigma)y^{m(\sigma)} = \frac{4}{5} \tag{52.97}$$

$$ATT = \mathcal{Y}_{1|1} - \mathcal{Y}_{0|1} = \frac{1}{5} \tag{52.98}$$

$$ATE = ATT = ATU = SDO = \frac{1}{5}\ , \quad SB = 0 \tag{52.99}$$

This example is unusual in that it has a single stratum x, and for that stratum, the treated and untreated populations are **balanced** (of equal size). Also, the map $m()$ is 1-1 onto. If, for instance, $m(\sigma)=6$ for all $\sigma \in A_0$ and $m(\sigma)=5$ for $\sigma \in A_1$, then ATE, ATT, ATU, SDO would not all be same, and SB would not be zero.

In fact, whenever there is a single balanced stratum and the map $m()$ is 1-1 onto, Eq.(52.99) can be proven to be true using the methods of section 52.8.

52.14.2 Approximate strata-matching

It is very often the case that one can't find for a given individual σ another individual that has opposite d^σ but exactly the same value of x^σ. In such cases, one can discard all matchless individuals. But that would entail a loss of precious information. Instead of discarding orphans, a better way is to relax our demands and match individual σ with another individual $m(\sigma)$ such that x^σ and $x^{m(\sigma)}$ are very close in some metric. Alternatively, the matching individual might not be real; it might be a composite of individuals.

More precisely, for some arbitrary parameter $\epsilon > 0$, and an individual σ, define the **strata-matching set** $\mathcal{M}_\epsilon(\sigma)$ by[4]

$$\mathcal{M}_\epsilon(\sigma) = \{m : d^m = 1 - d^\sigma, dist(x^\sigma, x^m) \leq \epsilon\} \ , \tag{52.100}$$

where

$$dist(x^\sigma, x^m) = [x^\sigma]^T [\Sigma]^{-1} x^m \ , \tag{52.101}$$

where $\Sigma = \left\langle \underline{x}^\sigma, [\underline{x}^m]^T \right\rangle$. This metric $dist(x^\sigma, x^m)$ is called the **Mahalanobis distance**. We will call the case $\epsilon = 0$ an **exact strata-matching**, and the case $\epsilon \neq 0$ an **approximate strata-matching.**. To do an approximate strata-matching, replace $y^{m(\sigma)}$ by $\langle y \rangle^{\mathcal{M}_\epsilon(\sigma)}$ in the estimators given above for an exact strata-matching. $\langle y \rangle^{\mathcal{M}_\epsilon(\sigma)}$ is defined by

$$\langle y \rangle^{\mathcal{M}_\epsilon(\sigma)} = \frac{1}{|\mathcal{M}_\epsilon(\sigma)|} \sum_{m \in \mathcal{M}_\epsilon(\sigma)} y^m \ . \tag{52.102}$$

52.14.3 Unbiased strata-matching estimators

The estimators we obtained via strata-matching are biased because strata-matching, due to its non-uniqueness, introduces noise into the estimate. However, one can define new bias-corrected estimators. Following Ref.[6], we will next find an unbiased estimator of ATT_x using the biased estimator of ATT_x that we obtained by strata-matching.

Let $\widehat{\mathcal{Y}}_{|d,x}$ be an estimate of $\mathcal{Y}_{|d,x} = E_{|d,x}[\underline{y}]$ that is obtained, for example, via Linear Regression.

[4] One can use an ϵ that depends on σ. For example, let $\epsilon(\sigma, 5)$ satisfy $|\mathcal{M}_{\epsilon(\sigma,5)}(\sigma)| = 5$.

Claim 62 *The quantity*

$$\widehat{ATT}_x^{unbi} = \frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma \left[(y^\sigma - y^{m(\sigma)}) - (\widehat{\mathcal{Y}}_{|0,x^\sigma} - \widehat{\mathcal{Y}}_{|0,x^{m(\sigma)}}) \right] \tag{52.103}$$

is an unbiased estimate of ATT_x.

proof:

We begin by assuming a special case of SUTVA. Let

$$\underline{y}^\sigma = \underline{y}(d^\sigma) = \widehat{\mathcal{Y}}_{|d^\sigma,x^\sigma} + \underline{\epsilon}^\sigma \tag{52.104}$$

where

$$\left\langle \widehat{\mathcal{Y}}_{|d^\sigma,x^\sigma}, \underline{\epsilon}^\sigma \right\rangle_\sigma = 0 \tag{52.105}$$

Recall that the biased estimator of ATT_x obtained by strata-matching is

$$\widehat{ATT}_x^{bi} = \frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma (y^\sigma - y^{m(\sigma)}) \tag{52.106}$$

where σ and $m(\sigma)$ are matched (i.e., $x^\sigma \approx x^{m(\sigma)}$ and $d^{m(\sigma)} = 1 - d^\sigma$).

$$\widehat{ATT}_x^{bi} = \frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma \left(\widehat{\mathcal{Y}}_{|1,x^\sigma} - \widehat{\mathcal{Y}}_{|0,x^{m(\sigma)}} \right) \tag{52.107}$$

$$+ \frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma (\epsilon^\sigma - \epsilon^{m(\sigma)}) \tag{52.108}$$

$$\widehat{ATT}_x^{bi} = \underbrace{\frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma \left(\widehat{\mathcal{Y}}_{|1,x^\sigma} - \widehat{\mathcal{Y}}_{|0,x^\sigma} \right)}_{ATT_x^{unbi}} \tag{52.109}$$

$$+ \underbrace{\frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma \left(\widehat{\mathcal{Y}}_{|0,x^\sigma} - \widehat{\mathcal{Y}}_{|0,x^{m(\sigma)}} \right)}_{\Delta ATT_x} \tag{52.110}$$

$$+ \underbrace{\frac{1}{N_x} \sum_{\sigma \in A_x} d^\sigma (\epsilon^\sigma - \epsilon^{m(\sigma)})}_{\mathcal{E}_x} \tag{52.111}$$

$$ATT_x^{unbi} = \underbrace{\widehat{ATT}_x^{bi} - \Delta ATT_x}_{\widehat{ATT}_x^{unbi}} - \mathcal{E}_x \tag{52.112}$$

By the Central Limit Theorem, for large N_x, this sum over $\sigma \in A_x$ of i.i.d. summands is normally distributed

$$\sqrt{N_x} ATT_x^{unbi} \sim \mathcal{N}(x; 0, var) \tag{52.113}$$

The reason for the $\sqrt{N_x}$ normalization is that we want the variance to be proportional to N_x.

$$\begin{aligned} var &= N_x \left\langle ATT_x^{unbi}, ATT_x^{unbi} \right\rangle & (52.114) \\ &= N_x \left\langle \widehat{ATT}_x^{unbi}, \widehat{ATT}_x^{unbi} \right\rangle + N_x \left\langle \mathcal{E}_x, \mathcal{E}_x \right\rangle & (52.115) \end{aligned}$$

QED

52.15 Propensities

It is often the case that the discrete vector $\underline{x}^\sigma$ has too many possible values to make matching possible. In such cases, it is convenient to map the space of vectors $\underline{x}^\sigma$ to the real line. One very convenient choice for that map is the **propensity score**, which is defined as

$$g(x^\sigma) = P(\underline{d}^\sigma = 1 | x^\sigma) \ . \tag{52.116}$$

$P(\underline{d}^\sigma = 1 | x^\sigma)$ is easy to calculate from the dataset. $g(x) = N_{1,x}/N_x$.

Figure 52.14: Bnet G_{ps} used when doing propensity scoring.

To use the propensity score, one replaces the bnet G_+ by the bnet G_{ps} as shown in Fig.52.14. The TPMs, printed in blue, for the 2 nodes of G_{ps} that differ from the nodes of G_+, are as follows:

$$P(g^\sigma | x^\sigma) = \delta(g^\sigma, g(x^\sigma)) \tag{52.117}$$

$$P(d^\sigma|g^\sigma) = g^\sigma d^\sigma + (1-g^\sigma)(1-d^\sigma) \tag{52.118}$$

Note that these TPMs are self-consistent because

$$\begin{aligned}
P(d|x) &= \sum_g P(d|g)P(g|x) && (52.119)\\
&= g(x)d + [1-g(x)](1-d) && (52.120)\\
&= P(\underline{d}=1|x)d + [1-P(\underline{d}=1|x)](1-d) && (52.121)\\
&= P(d|x) && (52.122)
\end{aligned}$$

We would like to do **propensity score strata-matching** by matching g-strata instead of x-strata. PO calculations for x- strata-matching use the TPMs for $P(d|x)$, $P(x)$ and $P(y|d,x)$. To do g- strata-matching using the same equations, but with x replaced by g, we would need to solve for $P(d|g)$, $P(g)$ and $P(y|d,g)$ in terms of $P(d|x)$, $P(x)$ and $P(y|d,x)$. We solve for those next.

From the TPMs for G_{ps}, one has

$$\boxed{P(d|g) = gd + (1-g)(1-d)} \tag{52.123}$$

and

$$\boxed{P(g) = \sum_x \overbrace{\delta(g,g(x))}^{P(g|x)} P(x)}\,. \tag{52.124}$$

Next, note that

$$P(y|d,g) = \sum_x P(y|d,x)P(x|g) \tag{52.125}$$

so we need to find $P(x|g)$. Since

$$\begin{aligned}
P(x|g) &= \frac{P(g|x)P(x)}{P(g)} && (52.126)\\
&= \frac{\delta(g,g(x))P(x)}{P(g)} && (52.127)
\end{aligned}$$

we finally get

$$\boxed{P(y|d,g) = \sum_x P(y|d,x)\frac{\delta(g,g(x))P(x)}{P(g)}}\,. \tag{52.128}$$

Eq.(52.128) looks complicated, but all it is saying is that

$$P(y|d,g)P(g) = \sum_{x \in g^{-1}(g)} P(y|d,x)P(x) \ , \tag{52.129}$$

where $g^{-1}(g) = \{x : g(x) = g\}$. In general,

$$\sum_x \delta(g, g(x)) = \sum_{x \in g^{-1}(g)} \tag{52.130}$$

Recall that for any treatment effect $\mathcal{E} \in \{ATE, ATT, ATU, SB, SDO\}$, we can estimate $\mathcal{E}_x$, and then calculate $\mathcal{E}$ from it, using

$$\mathcal{E} = \sum_x P(x)\mathcal{E}_x \ . \tag{52.131}$$

Hence

$$\begin{aligned} \mathcal{E} &= \sum_g \sum_x \delta(g, g(x))P(x)\mathcal{E}_x & (52.132) \\ &= \sum_g \sum_{x \in g^{-1}(g)} P(x)\mathcal{E}_x & (52.133) \end{aligned}$$

Let

$$g_{d|x} = g_d(x) = P(d|x) \tag{52.134}$$

for $d \in \{0,1\}$. Note that $g_{0|x} + g_{1|x} = 1$. We will refer to $g_d(x)$ for $d \in \{0,1\}$ as a two **propensities** and to $g_1(x)$ as the **propensity score**.

Define

$$\delta_{y|x} = P(y|d=1,x) - P(y|d=0,x) \tag{52.135}$$

Note that

$$\begin{aligned} \delta_{y|x} &= \frac{P(y,d=1|x)}{g_{1|x}} - \frac{P(y,d=0|x)}{g_{0|x}} & (52.136) \\ &= \frac{P(y,d=1|x)g_{0|x} - P(y,d=0|x)g_{1|x}}{g_{0|x}g_{1|x}} & (52.137) \\ &= \frac{P(y,d=1|x) - P(y|x)g(x)}{g(x)(1-g(x))} & (52.138) \end{aligned}$$

Dividing each term in a sum over $d \in \{0,1\}$ by $g_d(x)$ is often called **inverse probability (or propensity) weighting** (IPW). $g_d(x) = P(\underline{d} = d|x)$ is the likelihood of strata x, so dividing each term by this likelihood increases the contribution to the sum by less likely strata and decreases the contribution by more likely strata.

Note that the backdoor adjustment formula can expressed as an IPW:

$$P(y|\mathcal{D}\underline{d}=d) = \sum_x P(y|d,x)P(x) \tag{52.139}$$

$$= \sum_x \frac{P(y,d,x)}{P(d|x)} \tag{52.140}$$

Note that $ATE = ACE$ can be expressed in terms of propensities:

$$ACE = \sum_y y[P(y|\mathcal{D}\underline{d}=1) - P(y|\mathcal{D}\underline{d}=0)] \tag{52.141}$$

$$= \sum_x P(x) \sum_y y\,[P(y|d=1,x) - P(y|d=0,x)] \tag{52.142}$$

$$= \sum_x P(x) \underbrace{\sum_y y\delta_{y|x}}_{ACE_x} \tag{52.143}$$

Note that

$$P(y(c)|d) = \sum_x P(y(c)|d,x)P(x|d) \tag{52.144}$$

$$= \sum_x P(y(c)|c,x)P(x|d) \text{ (by CIA)} \tag{52.145}$$

$$= \sum_x P(y|c,x)P(x|d) \text{ (by SUTVA)} \tag{52.146}$$

It's instructive to express all the other treatment effects besides ATE in terms of propensities:

$$ATT = \sum_y y[P(\underline{y}(1)=y|d=1) - P(\underline{y}(0)=y|d=1)] \tag{52.147}$$

$$= \sum_x P(x|d=1) \sum_y y\,[P(y|\underline{d}=1,x) - P(y|\underline{d}=0,x)] \text{ (by Eq.(52.146))} \tag{52.148}$$

$$= \sum_x P(x|d=1) \sum_y y\ \delta_{y|x} \tag{52.149}$$

$$= \sum_x P(x) \underbrace{\frac{1}{P(\underline{d}=1)} \sum_y y\ g_{1|x}\delta_{y|x}}_{ATT_x} \tag{52.150}$$

$$ATU = \sum_x P(x) \underbrace{\frac{1}{P(\underline{d}=0)} \sum_y y \; g_{0|x} \delta_{y|x}}_{ATU_x} \tag{52.151}$$

$$SB = \sum_y y[P(\underline{y}(0)=y|d=1) - P(\underline{y}(0)=y|d=0)] \tag{52.152}$$

$$= \sum_y y \sum_x P(y|\underline{d}=0,x)[P(x|d=1) - P(x|d=0)] \text{ (by Eq.(52.146))} \tag{52.153}$$

$$= \sum_x P(x) \underbrace{\sum_y y \frac{P(y,\underline{d}=0|x)}{g_{0|x}} \left[\frac{g_{1|x}}{P(\underline{d}=1)} - \frac{g_{0|x}}{P(\underline{d}=0)}\right]}_{SB_x} \tag{52.154}$$

$$SDO = \sum_y y[P(\underline{y}(1)=y|d=1) - P(\underline{y}(0)=y|d=0)] \tag{52.155}$$

$$= \sum_x P(x) \underbrace{\sum_y y \left[\frac{P(y,\underline{d}=1|x)}{P(\underline{d}=1)} - \frac{P(y,\underline{d}=0|x)}{P(\underline{d}=0)}\right]}_{SDO_x} \text{ (by SUTVA)} \tag{52.156}$$

52.16 Propensity based estimators of treatment effects

In the strata-matching section 52.14, we gave an estimate of ACE_x. In strata-matching, one fills in the missing counterfactual values via the map $m()$. This is justified because, by CIA (weak-d limit), the counterfactual distributions are assumed to equal the factual distributions (see Fig. 52.6). In this section, we give estimators of treatment effects that are based on the formulae derived in Section 52.15. The estimators given in this section do no require the map $m()$, because the identification of counterfactual distributions with factual ones was used to derive the formulae of Section 52.15.

Eq.(52.143) suggests the estimator

$$\widehat{ACE}_x = \frac{1}{N_x} \sum_{\sigma \in A_x} y^\sigma \left[\frac{d^\sigma - g(x^\sigma)}{g(x^\sigma)(1-g(x^\sigma))}\right] \tag{52.157}$$

$$= \frac{1}{N_x} \sum_{\sigma \in A_x} \left[\frac{d^\sigma y^\sigma}{g_{1|x^\sigma}} - \frac{(1-d^\sigma)y^\sigma}{g_{0|x^\sigma}}\right] \tag{52.158}$$

This estimator is unbiased (because it doesn't have a source of noise like the strata-matching estimators do), but it is still possible to improve it. We next define another

ACE estimator called **Doubly Robust Estimator** (DRE) that is also unbiased and has smaller variance.

Let $\widehat{\mathcal{Y}}_{|d,x}$ be an estimate of $\mathcal{Y}_{|d,x} = E_{\underline{y}|d,x}[\underline{y}]$ that is obtained, for example, via Linear Regression. Define the DRE estimator

$$\widehat{ACE}_x^{DRE} = \widehat{\mathcal{Y}}_{1|x}^{DRE} - \widehat{\mathcal{Y}}_{0|x}^{DRE} \tag{52.159}$$

where

$$\widehat{\mathcal{Y}}_{1|x}^{DRE} = \frac{1}{N_x} \sum_{\sigma \in A_x} \left[\frac{d^\sigma (y^\sigma - \widehat{\mathcal{Y}}_{|1,x^\sigma})}{g_{1|x^\sigma}} + \widehat{\mathcal{Y}}_{|1,x^\sigma}) \right] \tag{52.160}$$

and

$$\widehat{\mathcal{Y}}_{0|x}^{DRE} = \frac{1}{N_x} \sum_{\sigma \in A_x} \left[\frac{(1-d^\sigma)(y^\sigma - \widehat{\mathcal{Y}}_{|0,x^\sigma})}{g_{0|x^\sigma}} + \widehat{\mathcal{Y}}_{|0,x^\sigma}) \right] . \tag{52.161}$$

The DRE estimator $\widehat{\mathcal{Y}}_{1|x}^{DRE}$ requires first estimating 2 preparatory quantities, $\widehat{\mathcal{Y}}_{|1,x^\sigma}$ and $g_{1|x}$. It's called doubly robust because it remains unbiased even if one of the estimates of those 2 preparatory quantities is wrong, but not if both are wrong. Let's check this.

- Suppose the propensity $g_{1|x}$ is slightly wrong. So what because

$$E_{\sigma|1,x} \left[\frac{d^\sigma (y^\sigma - \widehat{\mathcal{Y}}_{|1,x})}{g_{1|x}} \right] = 0 \tag{52.162}$$

- Suppose $\widehat{\mathcal{Y}}_{|1,x}$ is slightly wrong. So what because

$$E_{\sigma|1,x} \left[\frac{-d^\sigma \widehat{\mathcal{Y}}_{|1,x} + g_{1|x} \widehat{\mathcal{Y}}_{|1,x}}{g_{1|x}} \right] = 0 \tag{52.163}$$

A similar argument can be used to show that $\widehat{\mathcal{Y}}_{0|x}^{DRE}$ is doubly robust too.

52.17 Positivity

Positivity or **non-zero overlap** is defined as the requirement that for all layers x,

$$0 < \underbrace{P(\underline{d}^\sigma = 1 | \underline{x}^\sigma = x)}_{g_{1|x}} < 1 \tag{52.164}$$

Figure 52.15: Pictorial representation of positivity. $g_{0|x} + g_{1|x} = 1$. $g_{0|x} = 0$ and $g_{1|x} = 0$ are forbidden.

or, equivalently,

$$\underbrace{P(\underline{d}^\sigma = 1|\underline{x}^\sigma = x)}_{g_{1|x}} > 0 \quad \text{and} \quad \underbrace{P(\underline{d}^\sigma = 0|\underline{x}^\sigma = x)}_{g_{0|x}} > 0 \ . \tag{52.165}$$

In other words, for each layer x, there is a non-zero probability of being both treated and untreated. See Fig.52.15 for a pictorial representation of positivity.

If positivity is violated for some layer x, then

- the propensity based estimate Eq.(52.158) for ACE_x (which equals ATE_x) is undefined.
- all strata-matching estimates of $\mathcal{E}_x$ that use the matching function $m()$ are undefined because that function is undefined if $A_{0,x} = \emptyset$ or $A_{1,x} = \emptyset$.

If a quantity (estimand) can be estimated, it is said to be **do-identifiable** (i.e., expressible without do() operators). If positivity is violated, then $ACE = ATE$ is not identifiable.

When $P(d|x)$ becomes 0 or 1 for some x, the arrow $\underline{x} \to \underline{d}$ becomes deterministic for that x. This situation is the very antithesis of RCTs, wherein the influence exerted by $\underline{x}^\sigma$ on $\underline{d}^\sigma$ is uniformly random and therefore ignorable. Hence, it is perhaps not too surprising that a violation of positivity makes $ACE = ATE$ not identifiable.

52.18 Multi-time PO bnets (Panel Data)

In this section, we will discuss Multi-time PO bnets (MT-PO).

A **time-series** is a function $f : D \to \mathbb{R}$ whose domain D is a discrete set of times. A time-series usually describes a single unit σ (i.e., an individual) in a population.

An **observational study (or analysis or model)** can be cross-sectional or longitudinal. A **cross-sectional study** collects and analyzes a **cross-sectional**

dataset; i.e., a dataset for a population at a single time. A **longitudinal study or panel study** collects and analyzes a **longitudinal dataset**; i.e., a dataset for a population at multiple times. Thus, a longitudinal study consists of one or more time-series.

Let $\mathcal{T} = \{t_0, t_1, \ldots, t_{nt-1}\}$. For any time-series $a_t : \mathcal{T} \to \mathbb{R}$, define

$$E_t a_t = \frac{1}{nt} \sum_{t \in \mathcal{T}} a_t \tag{52.166}$$

$$\Delta_t a_t = a_t - E_t a_t \tag{52.167}$$

$$\langle a_t, b_t \rangle_t = E_t \Delta_t a_t \Delta_t b_t \tag{52.168}$$

Consider a quantity a_t^σ that is a function of the time t and of the particular unit σ in a population. a_t^σ is said to be a **t-constant effect** if it is t-independent. a_t^σ is said to be a **homogeneous effect** (antonym: **heterogeneous effect**) if it is σ-independent. Henceforth, we will avoid using the word "effect" for these because that word has already been used for "treatment effect" in PO theory. Instead, we will use the word "quantity".

Figure 52.16: Example of multi-time PO bnet with t-constant quantities $\underline{x}^\sigma, \underline{u}^\sigma$. The 3 nodes $\underline{x}^\sigma$ should be identified as a single node. Likewise, the 3 nodes $\underline{u}^\sigma$ should be identified as a single node.

Fig.52.16 gives an example of a multi-time PO bnet (MT-PO). Note that in this example, $\underline{x}^\sigma$ and $\underline{u}^\sigma$ are t-constant quantities. $\underline{u}^\sigma$ is an unobserved confounder and $\underline{x}^\sigma$ is an observed confounder. For convenience and simplicity, we will assume linear deterministic TPMs. The TPMs for the bnet Fig.52.16, printed in blue, are as follows:

$$P(x^\sigma) = P_{\underline{x}}(x^\sigma) \tag{52.169}$$

$$P(u^\sigma) = P_{\underline{u}}(u^\sigma) \tag{52.170}$$

$$P(y_t^\sigma | d_t^\sigma, x^\sigma, u^\sigma) = \mathbb{1}(\ \ y_t^\sigma = \delta d_t^\sigma + \beta x^\sigma + u^\sigma\ \ ) \tag{52.171}$$

$$P(d_{t+1}^\sigma | d_t^\sigma, x^\sigma, u^\sigma) = \mathbb{1}(\ \ d_{t+1}^\sigma = \alpha d_t^\sigma + \gamma x^\sigma + u^\sigma\ \ ) \tag{52.172}$$

Taking time averages of the treatment dose and treatment outcome, we get

$$E_t \underline{y}_t^\sigma = \delta E_t \underline{d}_t^\sigma + \beta \underline{x}^\sigma + \underline{u}^\sigma\ , \tag{52.173}$$

$$E_t \underline{d}_{t+1}^\sigma = \alpha E_t \underline{d}_t^\sigma + \gamma \underline{x}^\sigma + \underline{u}^\sigma\ . \tag{52.174}$$

Subtracting the time averages from the quantities being averaged, we get

$$\Delta_t \underline{y}_t^\sigma = \delta \Delta_t \underline{d}_t^\sigma\ , \tag{52.175}$$

$$\Delta_t \underline{d}_{t+1}^\sigma = \alpha \Delta_t \underline{d}_t^\sigma\ . \tag{52.176}$$

This allows us to find estimators for δ and α:

$$E_\sigma \left\langle \underline{y}_t^\sigma, \underline{y}_t^\sigma \right\rangle_t = \delta E_\sigma \left\langle \underline{y}_t^\sigma, \underline{d}_t^\sigma \right\rangle_t \tag{52.177}$$

$$\delta = \frac{E_\sigma \left\langle \underline{y}_t^\sigma, \underline{y}_t^\sigma \right\rangle_t}{E_\sigma \left\langle \underline{y}_t^\sigma, \underline{d}_t^\sigma \right\rangle_t} \tag{52.178}$$

$$E_\sigma \left\langle \underline{d}_{t+1}^\sigma, \underline{d}_{t+1}^\sigma \right\rangle_t = \alpha E_\sigma \left\langle \underline{d}_{t+1}^\sigma, \underline{d}_t^\sigma \right\rangle_t \tag{52.179}$$

$$\alpha = \frac{E_\sigma \left\langle \underline{d}_{t+1}^\sigma, \underline{d}_{t+1}^\sigma \right\rangle_t}{E_\sigma \left\langle \underline{d}_{t+1}^\sigma, \underline{d}_t^\sigma \right\rangle_t} \tag{52.180}$$

As shown in Fig.52.17, subtraction of time averages from each node removes the confounder nodes from the bnet of Fig.52.16 (However, this assumes that the confounders are t-constant and that the TPMs are linear deterministic, two very strong assumptions).

$$
\begin{tikzcd}
\Delta_t \underline{d}_t^{\sigma} \arrow[r, "\alpha"'] \arrow[d, "\delta"'] & \Delta_t \underline{d}_{t+1}^{\sigma} \arrow[d] \\
\Delta_t \underline{y}_t^{\sigma} & \Delta_t \underline{y}_{t+1}^{\sigma}
\end{tikzcd}
$$

Figure 52.17: time-average-subtracted (TAS) bnet for the bnet of Fig.52.16.

Chapter 53

Program evaluation and review technique (PERT)

This chapter is based on Refs.[52] and [114].

PERT diagrams are used for scheduling a project consisting of a series of interdependent activities and estimating how long it will take to finish the project. PERT diagrams were invented by the NAVY in 1958 to manage a submarine project. Nowadays they are taught in many business and management courses.

A **PERT diagram** is a Directed Acyclic Graph (DAG) with the following properties. (See Fig.53.2 for an example of a PERT diagram). The nodes $\underline{E}_i$ for $i = 1, 2, \ldots, ne$ of a PERT diagram are called **events**. The edges $i \to j$ of a PERT diagram are called **activities**. An event represents the starting (kickoff) date of one or more activities. A PERT diagram has a single root node ($i = 1$, start event) and a single leaf node ($i = ne$, end event).

The PERT diagram user must initially provide a **Duration Times (DT) table** which gives $(DO_{i\to j}, DP_{i\to j}, DM_{i\to j})$ for each activity $i \to j$, where

$DO_{i\to j}$= optimistic duration time of activity $i \to j$

$DP_{i\to j}$= pessimistic duration time of activity $i \to j$

$DM_{i\to j}$= median duration time of activity $i \to j$

From the DT table, one calculates:

Duration time of activity $i \to j$

$$D_{i\to j} = \frac{1}{6}(DO_{i\to j} + DP_{i\to j} + 4DM_{i\to j}) \tag{53.1}$$

Duration Variance of activity $i \to j$

$$V_{i\to j} = \left(\frac{DO_{i\to j} - DP_{i\to j}}{DM_{i\to j}}\right)^2 \tag{53.2}$$

Often, it is convenient to define "dummy" edges with $D_{i\to j} = 0$. That is perfectly fine.

Define:

TES_i = Earliest start time for event i
TLS_i = Latest start time for event i
$slack_i = TLS_i - TES_i$ = slack for event i
$TEF_{i\to j} = TES_i + D_{i\to j}$ = Earliest finish time for activity $i \to j$.
$TLF_{i\to j} = TLS_j - D_{i\to j}$ = Latest finish time for activity $i \to j$. See footnote below. [1]

A **critical path** is a directed path (i.e., a chain of connected arrows, all pointing in the same direction) going from the start to the end node, such that slack equals zero at every node visited. In a DAG, the neighbors of a node is the union of its parent and children nodes. A critical path must also have all other nodes as neighbors; i.e, the union of the neighbors of every node in the path plus the nodes in the path itself, equals all nodes in the graph.

GOAL of PERT analysis: The main goal of PERT analysis is to find, based on the data of the DT table, the interval $[TES_i, TLS_i]$ giving a lower and an upper bound to the starting time of each node i. Another goal is to find a critical path for the PERT diagram (which represents an entire project). By adding the $D_{i\to j}$ of each edge of the critical path, one can get the mean value of the total duration of the entire project, and by adding the variances of each edge along the critical path, one can get an estimate of the total variance of the total duration. Knowing the mean and variance of the total duration and assuming a Normal distribution, one can predict the probability that the actual duration will deviate by a certain amount from its mean.

To calculate the interval $[TES_i, TLS_i]$, one follows the following two steps.

1. Assume $TES_1 = 0$ and solve

$$TES_i = \max_{a\in pa(i)} (\underbrace{TES_a + D_{a\to i}}_{TEF_{a\to i}}) \tag{53.3}$$

   for $i \in [2, ne]$. This recursive equation is solved by what is called "forward propagation", wherein one moves up the list of nodes i in order of increasing i starting at $i = 1$ with $TES_1 = 0$.

2. Assume $TLS_{ne} = TES_{ne}$ and solve

$$TLS_i = \min_{b\in ch(i)} (\underbrace{TLS_b - D_{i\to b}}_{TLF_{i\to b}}) \tag{53.4}$$

[1] In the popular educational literature, the edge variables $TEF_{i\to j}$ and $TLF_{i\to j}$ are sometimes associated with the nodes, but they are clearly edge variables. This makes things confusing. The reason this is done is that some software draws PERT diagrams as trees whereas other software draws them as DAGs. For trees, storing $TEF_{i\to j}$ and $TLF_{i\to j}$ in a node makes some sense but not for DAGs. You will notice that giving specific names to the variables $TEF_{i\to j}$ and $TLF_{i\to j}$ is unnecessary. It is possible to delete all mention of their names from this chapter without losing any details. I only declare their names in this chapter so as tell the reader what they are in case he/she hears them mentioned and wonders what they are equal to in our notation.

for $i \in [1, ne-1]$. This recursive equation is solved by what is called "backward propagation", wherein one moves down the list of nodes i in order of decreasing i starting at $i = ne$ with $TLS_{ne} = TES_{ne}$. TES_{ne} is known from step 1.

Eqs.(53.3) and (53.4) are illustrated in Fig.53.1.

Figure 53.1: TES_i defined from info received from parents of i and TLS_i defined from info received from children of i.

53.1 Example

To illustrate PERT analysis, we end with an example. We present the example in the form of an exercise question and then provide the answer. This example comes from Ref.[52], except for part (e) about bnets, which is our own.

Question: For the PERT diagram of Fig.53.2, calculate the following:

(a) Interval $[TES_i, TLS_i]$ for all i.

(b) A critical path for this PERT diagram.

(c) The mean and variance of the total duration of the critical path.

(d) The probability that the total duration will be 225 days or less.

(e) A bnet interpretation of this problem.

Answer to (a) $[TES_i, TLS_i]$ are given by Fig.53.3.

Answer to (b) The critical path is given in red in Fig.53.3. Note that this path does indeed have zero slack at each node it visits and the union of its neighborhood and the path itself encompasses all nodes.

Answer to (c) The mean and variance of the total duration are calculated in Table 53.1.

Answer to (d)

$$P(\underline{x} < 225) = P\left[\frac{\underline{x}-\mu}{\sigma} \leq \frac{225-220}{\sqrt{7.73}}\right] \tag{53.5}$$

$$= P[\underline{z} \leq 1.80] \tag{53.6}$$

$$= 0.9641 \tag{53.7}$$

Figure 53.2: Example of a PERT diagram. The numbers attached to the arrows are the duration times $D_{i\to j}$ in days followed by, enclosed in parentheses, the variance $V_{i\to j}$ of that duration. The info given in this PERT diagram was derived from a DT table in Ref.[52]. The info in this PERT diagram is sufficient for calculating TES_i and TLS_i for each node i. The results of that calculation are given in Fig.53.3.

TES_i (given after the node name) for node $\underline{E}_i$ for all i

TLS_i (given after the node name) for node $\underline{E}_i$ for all i

Figure 53.3: Results of calculating TES_i for all i via a forward pass, followed by calculating TLS_i for all i via a backward pass. Critical path indicated in red.

Answer to (e) Define 2 bnets.

1. The first PERT bnet is for calculating TES_i for all i and is given by Fig.53.4.

edge $i \to j$	duration $D_{i\to j}$	variance $V_{i\to j}$
A $(1 \to 2)$	50	2.56
D $(2 \to 5)$	60	0.44
G $(5 \to 7)$	30	1.78
J $(7 \to 8)$	40	1.31
K $(8 \to 9)$	10	0.64
L $(9 \to 10)$	30	1.00
Total	220	7.73

Table 53.1: Calculation of mean and variance of total duration along critical path.

Figure 53.4: bnet for TES_i calculation.

The node TPMs, printed in blue, for the bnet Fig.53.4 are given by (this equation is to be evaluated recursively by a forward pass through the bnet):

$$P(TES_i|(TES_a)_{a\in pa(i)}) = \delta(TES_i, \max_{a\in pa(i)}(TES_a + D_{a\to i})) \tag{53.8}$$

2. The second PERT bnet is for calculating TLS_i for all i and is given by Fig.53.5. Note that the directions of all the arrows in the PERT diagram Fig.53.2 have been reversed so Fig.53.5 is a time reversed graph.

   The node TPMs, printed in blue, for the bnet Fig.53.5 are given by (this equation is to be evaluate recursively by a backward pass through the bnet):

$$P(TLS_i|(TLS_b)_{b\in pa(i)}) = \delta(TLS_i, \min_{b\in pa(i)}(TLS_b - D^T_{b\to i}))\ , \tag{53.9}$$

   where $D^T_{i\to j} = D_{j\to i}$.

Figure 53.5: bnet for TLS_i calculation.

Chapter 54

Random Forest and Bagging

This chapter is based on Refs.[71] and [115].

Chapter 12 defines decision trees (dtrees) and explains how to construct them. A Random Forest (RF) is an ensemble of dtrees. The RF algorithm is a method of, given a dataset, constructing a RF and averaging over the classifier functions of the RF to produce an ensemble classifier.

The RF algo uses the method of Bootstrap Aggregating (aka Bagging), which is discussed in detail below. Bagging is a method of constructing an ensemble of datasets (called **bootstrap datasets or bags**) that are fairly uncorrelated. Each of these bags is used to train a **bag-classifier**. The bag-classifiers are averaged over to produce an **ensemble classifier, or e-classifier** for short. Bagging can be used to train any type of bag-classifier, but it was invented with dtrees in mind, and is still most commonly used to train dtrees.

Boosting (see Chapter 1 on AdaBoost and Chapter 76 on XGBoost) is another method, besides Bagging, of constructing a classifier function from an ensemble of classifier functions. These two methods are most commonly applied to dtrees: Boosting for an ensemble of small dtrees, and Bagging for a random forest (which is an ensemble of dtrees that are usually much more complicated than small dtrees).

54.1 Bagging (with fully-featured bags)

In this section we discuss the bagging algorithm. As already mentioned, bagging is usually used to train dtrees. In this section, we explain bagging in general, not just for dtrees.

Let $L = [0, 1, 2, \ldots, nsam-1]$ be a list of individuals (samples) in a population. In this chapter, we will use the notation $A^\sigma = A[\sigma]$ and $\vec{A} = [A^\sigma : \sigma \in L]$ for a list (vector, 1-D array) indexed by L. We will refer to $DS = (\vec{x}, \vec{y})$ where $x^\sigma \in S_{\underline{x}}$, $y^\sigma \in S_{\underline{y}}$, as a dataset. If L_j is a list (possibly with duplicate items) such that $set(L_j) \subset set(L)$, then define $DS_j = (\vec{x}, \vec{y})_{L_j} = ((x^\sigma)_{\sigma \in L_j}, (y^\sigma)_{\sigma \in L_j})$. We will refer to DS_j as the **restriction of $(\vec{x}, \vec{y})$ to L_j.**

Figure 54.1: Bnet for Random Forest (RF) with 5 bags.

We will refer to a function $Y : S_{\underline{x}} \to S_{\underline{c}}$ as a classifier. It maps vector of vector of features $x \in S_{\underline{x}}$ to a class $c \in S_{\underline{c}}$. Below, Y_b for all b and Y_{ens} are classifiers.

Fig.54.1 is a bnet that encapsulates the RF algo. The TPMs, printed in blue, for the nodes of bnet Ref.54.1, are as follows.

Let $b \in \{0, 1, 2, \ldots, nbags - 1\} = B$ and $\sigma \in L$. Let $L_b^\sigma \in L$ and

$$P(L_b^\sigma) = 1/nsam \tag{54.1}$$

In other words, each item in list L_b is chosen from the items of list L, uniformly at random with replacements. $|L_b| = |L|$ (same size as original). L_b can have duplicate items and be missing items from L.

$$P((\vec{x}, \vec{y})_{L_b} | (\vec{x}, \vec{y}), L_b) = \mathbb{1}(\quad (\vec{x}, \vec{y})_{L_b} = \text{restriction of } (\vec{x}, \vec{y}) \text{ to } L_b \quad) \tag{54.2}$$

We will refer to $(\vec{x}, \vec{y})$ as the **original dataset** and to the $(\vec{x}, \vec{y})_{L_b}$ for $b \in B$ as the **bootstrap datasets** or **bags**.

$$P(Y_b | (\vec{x}, \vec{y})_{L_b}) = \mathbb{1}(\quad Y_b(\cdot) = \text{classifier trained on dataset } (\vec{x}, \vec{y})_{L_b}. \quad) \tag{54.3}$$

$$P(Y_{ens} | (Y_b)_{b \in B}) = \prod_\sigma \mathbb{1}(\quad Y_{ens}(x^\sigma) = \texttt{majority}(\{Y_b(x^\sigma) : b \in B\}) \quad) \tag{54.4}$$

The `majority()` function can be replaced by an average $\frac{1}{nbags} \sum_b$ in case the set of classes $S_{\underline{c}}$ equals $\mathbb{R}$ rather than a finite set. We will refer to Y_{ens} as the **ensemble classifier (e-classifier)** and to the Y_b as the **bag-classifiers**.

Figure 54.2: $\Sigma(b)$ and $\Sigma^c(b)$ are disjoint sets whose union is Σ. $B(\sigma)$ and $B^c(\sigma)$ are disjoint sets whose union is B.

Define (these definitions are illustrated in Fig.54.2)

$$\Sigma = set(L)\ ,\ \ \Sigma(b) = set(L_b)\ ,\ \ \Sigma^c(b) = \Sigma - \Sigma(b) \tag{54.5}$$

and

$$B(\sigma) = \{b \in B : \sigma \in \Sigma(b)\}\ ,\ \ B^c(\sigma) = B - B(\sigma) \tag{54.6}$$

$\Sigma(b)$ is the set of **in-the-b-bag individuals** and $\Sigma^c(b)$ is the set of **out-of-the-b-bag (OOB) individuals**. $B(\sigma)$ is the set of bags that contain individual σ, and $B^c(\sigma)$ is the set of bags that don't.

The **OOB error** is defined as

$$err = \sum_{\sigma \in L} \mathbb{1}(B^c(\sigma) \neq \emptyset)\mathbb{1}(\quad y^\sigma \neq \texttt{majority}([Y_b(x^\sigma) : b \in B^c(\sigma)])\quad)\ . \tag{54.7}$$

Empirical results supposedly show that OOB error is comparable in accuracy to the error calculated by doing cross-validation (CV) (see Chapter 11), although CV error is considered more dependable.

54.2 Bagging (with randomly-shortened bags)

Suppose the feature vector x^σ in the dataset $DS = (\vec{x}, \vec{y})$ has nf components; i.e., $x^\sigma = (x^\sigma_0, x^\sigma_1, \ldots, x^\sigma_{nf-1}) \in S_{\underline{x}_0} \times S_{\underline{x}_1} \times \ldots \times S_{\underline{x}_{nf-1}} = S_{\underline{x}}$.

For each bag DS_b, one chooses at random $nf' = \sqrt{nf}$ out of the nf features, and discards the remaining features from DS_b, thus producing a new, **randomly-shortened-bag (rs-bag)** DS'_b. Each rs-bag is then used to train a bag-classifier, usually a dtree, using the methods for dtree SL described in Chapter 12. Using rs-bags is called the **random subspace method**. The reason for using rs-bags is that they further decorrelate the set of bags used to train bag-classifiers.

Chapter 55

Recurrent Neural Networks

This chapter is mostly based on Ref.[29].

This chapter assumes you are familiar with the material and notation of Chapter 48 on plain Neural Nets.

$$
\begin{tikzcd}
\underline{x}(\cdot) \arrow[d] \arrow[dr] \arrow[drr] & & \\
\underline{x}(0) \arrow[d] & \underline{x}(1) \arrow[d] & \underline{x}(2) \arrow[d] \\
\underline{h}(0) \arrow[r] \arrow[d] & \underline{h}(1) \arrow[r] \arrow[d] & \underline{h}(2) \arrow[d] \\
\underline{Y}(0) & \underline{Y}(1) & \underline{Y}(2)
\end{tikzcd}
$$

Figure 55.1: Simple example of RNN with $T = 3$

Suppose
T is a positive integer.
$t = 0, 1, \ldots, T-1$,
$\underline{x}_i(t) \in \mathbb{R}$ for $i = 0, 1, \ldots, nx-1$,
$\underline{h}_i(t) \in \mathbb{R}$ for $i = 0, 1, \ldots, nh-1$,
$\underline{Y}_i(t) \in \mathbb{R}$ for $i = 0, 1, \ldots, ny-1$,
$W^{h|x} \in \mathbb{R}^{nh \times nx}$,
$W^{h|h} \in \mathbb{R}^{nh \times nh}$,
$W^{y|h} \in \mathbb{R}^{ny \times nh}$,
$b^y \in \mathbb{R}^{ny}$,
$b^h \in \mathbb{R}^{nh}$.
Henceforth, $x(\cdot)$ will mean the array of $x(t)$ for all t.

The simplest kind of recurrent neural network (RNN) has the bnet Fig.55.1 with arbitrary T. The node TPMs, printed in blue, for this bnet, are as follows.

$$P(x(\cdot)) = \text{given} \tag{55.1}$$

$$P(x(t)) = \delta(x(t), [x(\cdot)]_t) \tag{55.2}$$

$$P(h(t) \mid h(t-1), x(t)) = \delta(h(t), \mathcal{A}(W^{h|x}x(t) + W^{h|h}h(t-1) + b^h)) \ , \tag{55.3}$$

where $h(-1) = 0$.

$$P(Y(t) \mid h(t)) = \delta(Y(t), \mathcal{A}(W^{y|h}h(t) + b^y)) \tag{55.4}$$

Define

$$W^h = [W^{h|x}, W^{h|h}, b^h] \ , \tag{55.5}$$

and

$$W^y = [W^{y|h}, b^y] \ . \tag{55.6}$$

The bnet of Fig.55.1 can be used for classification once its parameters W^h and W^y have been optimized. To optimize those parameters via gradient descent, one can use the bnet of Fig.55.2.

Let $\sigma = 0, 1, \ldots, nsam(\vec{x}) - 1$ be the labels for a minibatch of samples. Below, we will write $A^{[\sigma]} = A[\sigma] = [\vec{A}]^\sigma$ for the σ component of any vector $\vec{A}$. The node TPMs, printed in blue, for bnet Fig.55.2, are as follows.

$$P(x(\cdot)^{[\sigma]}) = \text{given} \tag{55.7}$$

$$P(x(t)^{[\sigma]}) = \delta(x(t)^{[\sigma]}, [x(\cdot)]_t^{[\sigma]}) \tag{55.8}$$

$$P(h(t)^{[\sigma]} \mid h(t-1)^{[\sigma]}, x(t)^{[\sigma]}) = \delta(h(t)^{[\sigma]}, \mathcal{A}(W^{h|x}x(t)^{[\sigma]} + W^{h|h}h(t-1)^{[\sigma]} + b^h)) \tag{55.9}$$

$$P(Y(t)^{[\sigma]} \mid h(t-1)^{[\sigma]}) = \delta(Y(t)^{[\sigma]}, \mathcal{A}(W^{y|h}h(t-1)^{[\sigma]} + b^y)) \tag{55.10}$$

Figure 55.2: RNN bnet used to optimize parameters W^h and W^y of RNN bnet Fig.55.1.

$$P(y(\cdot)^{[\sigma]} \mid x(\cdot)^{[\sigma]}) = \text{given} \tag{55.11}$$

$$P(\mathcal{E}(t) \mid \vec{y}(t), \vec{Y}(t)) = \frac{1}{nsam(\vec{x})} \sum_{\sigma} d(y(t)^{[\sigma]}, Y(t)^{[\sigma]}) \ , \tag{55.12}$$

where

$$d(y, Y) = |y - Y|^2 \ . \tag{55.13}$$

If $y, Y \in [0,1]$, one can use this instead

$$d(y,Y) = XE(y \to Y) = -y \ln Y - (1-y)\ln(1-Y) \ . \tag{55.14}$$

$$P(\mathcal{E} \mid \mathcal{E}(\cdot)) = \delta(\mathcal{E}, \sum_t \mathcal{E}(t)) \tag{55.15}$$

For $a = h, y$,

$$P(W^a) = \text{given} \ . \tag{55.16}$$

The first time it is used, W^a is arbitrary. Afterwards, it is determined by previous horizontal stage.

$$P((W^a)'|\mathcal{E}, W^a) = \delta((W^a)', W^a - \eta^a \partial_{W^a}\mathcal{E}) \ . \tag{55.17}$$

$\eta^a > 0$ is the learning rate for W^a.

55.1 Language Sequence Modeling

Estimate $P(x(\cdot))$ empirically. We can use this to:

- predict the probability of a sentence,

  example: Get $P(x(0), x(1), x(2))$.

- predict the most likely next word in a sentence,

  example: Get $P(x(2)|x(0), x(1))$.

- generate fake sentences.

  example:

  Get $x(0) \sim P(x(0))$.

  Next get $x(1) \sim P(x(1)|x(0))$.

  Next get $x(2) \sim P(x(2)|x(0), x(1))$.

55.2 Other types of RNN

Let $\mathcal{T} = \{0, 1, \ldots, T-1\}$, and $\mathcal{T}^x, \mathcal{T}^y \subset \mathcal{T}$. Above, we assumed that $\underline{x}(t)$ and $\underline{Y}(t)$ were both defined for all $t \in \mathcal{T}$. More generally, they might be defined only for subsets of $\mathcal{T}$: $\underline{x}(t)$ for $t \in \mathcal{T}^x$ and $\underline{Y}(t)$ for $t \in \mathcal{T}^y$. If $|\mathcal{T}^x| = 1$ and $|\mathcal{T}^y| > 1$, we say the RNN bnet is of the 1 to many kind. In general, can have **1 to 1, 1 to many, many to 1, many to many** RNN bnets.

Plain RNNs can suffer from the **vanishing or exploding gradients problem**. There are various ways to mitigate this (e.g., good choice of initial W^h and

$$
\begin{tikzcd}
\underline{x}(\cdot) \arrow[d] \arrow[dr] & & & \\
\underline{x}(0) \arrow[d] & \underline{x}(1) \arrow[d] & & \\
\underline{h}(0) \arrow[r] & \underline{h}(1) \arrow[r] & \underline{h}(2) \arrow[r] \arrow[d] & \underline{h}(3) \arrow[d] \\
& & \underline{Y}(2) & \underline{Y}(3)
\end{tikzcd}
$$

Figure 55.3: RNN bnet of the many to many kind. This one can be used for translation. $x(0)$ and $x(1)$ might denote two words of an English sentence, and $Y(2)$ and $Y(3)$ might be their Italian translation.

W^y, good choice of activation functions, regularization). Or by using GRU or LSTM (discussed below). **GRU and LSTM** were designed to mitigate the vanishing or exploding gradients problem. They are very popular in NLP (Natural Language Processing).

55.2.1 Long Short Term Memory (LSTM) unit (1997)

This section is based on Wikipedia article Ref.[102]. In this section, $\odot$ will denote the Hadamard matrix product (elementwise product).

Figure 55.4: bnet for a Long Short Term Memory (LSTM) unit.

Let
$\underline{x}(t) \in \mathbb{R}^{nx}$: **input state** vector to the LSTM unit
$\underline{f}(t) \in \mathbb{R}^{nh}$: **forget activation** vector
$\underline{i}(t) \in \mathbb{R}^{nh}$: **input activation** vector
$\underline{o}(t) \in \mathbb{R}^{nh}$: **output activation** vector
$\underline{h}(t) \in \mathbb{R}^{nh}$: **hidden state** vector
$\underline{\tilde{c}}(t) \in \mathbb{R}^{nh}$: **cell activation** vector
$\underline{c}(t) \in \mathbb{R}^{nh}$: **cell state** vector
$\underline{Y}(t) \in \mathbb{R}^{ny}$: **classification** of $x(t)$.
$W \in \mathbb{R}^{nh \times nx}$, $U \in \mathbb{R}^{nh \times nh}$ and $b \in \mathbb{R}^{nh}$: weight matrices and bias vectors, parameters learned by training.
$\mathcal{W}^{y|h} \in \mathbb{R}^{ny \times nh}$: weight matrix

Fig.55.4 is a bnet for a LSTM unit. The node TPMs, printed in blue, for this bnet, are as follows.

$$P(f(t)|x(t), h(t-1)) = \mathbb{1}(\quad f(t) = \text{smoid}(W^{f|x}x(t) + U^{f|h}h(t-1) + b^f) \quad) \,, \tag{55.18}$$

where $h(-1) = 0$.

$$P(i(t)|x(t), h(t-1)) = \mathbb{1}(\quad i(t) = \text{smoid}(W^{i|x}x(t) + U^{i|h}h(t-1) + b^i) \quad) \tag{55.19}$$

$$P(o(t)|x(t), h(t-1)) = \mathbb{1}(\quad o(t) = \text{smoid}(W^{o|x}x(t) + U^{o|h}h(t-1) + b^o) \quad) \tag{55.20}$$

$$P(\tilde{c}(t)|x(t), h(t-1)) = \mathbb{1}(\quad \tilde{c}(t) = \tanh(W^{c|x}x(t) + U^{c|h}h(t-1) + b^c) \quad) \tag{55.21}$$

$$P(c(t)|f(t), c(t-1), i(t), \tilde{c}(t)) = \mathbb{1}(\quad c(t) = f(t) \odot c(t-1) + i(t) \odot \tilde{c}(t) \quad) \tag{55.22}$$

$$P(h(t)|o(t), c(t)) = \mathbb{1}(\quad h(t) = o(t) \odot \tanh(c(t)) \quad) \tag{55.23}$$

$$P(Y(t)|h(t)) = \mathbb{1}(\quad Y(t) = \mathcal{A}(\mathcal{W}^{y|h}h(t) + b^y) \quad) \tag{55.24}$$

55.2.2 Gated Recurrence Unit (GRU) (2014)

This section is based on Wikipedia article Ref.[85]. In this section, $\odot$ will denote the Hadamard matrix product (elementwise product).

GRU is a more recent (17 years later) attempt at simplifying LSTM.

Figure 55.5: bnet for a Gated Recurrent Unit (GRU).

Let
$\underline{x}(t) \in \mathbb{R}^{nx}$: **input state** vector
$\underline{h}(t) \in \mathbb{R}^{nh}$: **hidden state** vector
$\underline{\hat{h}}(t) \in \mathbb{R}^{nh}$: **hidden activation** vector
$\underline{z}(t) \in \mathbb{R}^{nh}$: **update activation** vector
$\underline{r}(t) \in \mathbb{R}^{nh}$: **reset activation** vector
$\underline{Y}(t) \in \mathbb{R}^{ny}$: **classification** of $x(t)$.
$W \in \mathbb{R}^{nh \times nx}$, $U \in \mathbb{R}^{nh \times nh}$ and $b \in \mathbb{R}^{nh}$: weight matrices and bias vectors, parameters learned by training.
$\mathcal{W}^{y|h} \in \mathbb{R}^{ny \times nh}$: weight matrix

Fig.55.5 is a bnet for a GRU. The node TPMs, printed in blue, for this bnet, are as follows.

$$P(z(t)|x(t), h(t-1)) = \mathbb{1}(\quad z(t) = \text{smoid}(W^{z|x}x(t) + U^{z|h}h(t-1) + b^z) \quad) \ , \tag{55.25}$$

where $h(-1) = 0$.

$$P(r(t)|x(t), h(t-1)) = \mathbb{1}(\quad r(t) = \text{smoid}(W^{r|x}x(t) + U^{r|h}h(t-1) + b^r) \quad) \tag{55.26}$$

$$P(\hat{h}(t)|x(t), r(t), h(t-1)) = \mathbb{1}(\quad \hat{h}(t) = \tanh(W^{h|x}x(t) + U^{h|h}(r(t) \odot h(t-1)) + b^h) \quad) \tag{55.27}$$

$$P(h(t)|z(t), h(t-1), \hat{h}(t)) = \mathbb{1}(\quad h(t) = (1 - z(t)) \odot h(t-1) + z(t) \odot \hat{h}(t) \quad) \tag{55.28}$$

$$P(Y(t)|h(t)) = \mathbb{1}(\quad Y(t) = \mathcal{A}(\mathcal{W}^{y|h} h(t) + b^y) \quad) \tag{55.29}$$

Chapter 56

Regression Discontinuity Design

This chapter is based on Ref.[6].

This chapter assumes that the reader has read Chapter 52 on Potential Outcomes (PO).

In Regression Discontinuity Design (RDD), one switches the treatment dose $\underline{d}$ from 0 when $\underline{x} < \xi$ to 1 where $\underline{x} > \xi$, where $\underline{x}$ is an observed confounder (call it the **switch confounder**) and ξ is a threshold value for $\underline{x}$. One measures the jump δ in the treatment outcome $\underline{y}$ as $\underline{x}$ passes through $\underline{x} = \xi$. Then one makes the very reasonable assumption that δ equals[1] $\mathcal{Y}_{1|x=\xi} - \mathcal{Y}_{0|x=\xi} = ATE_{|x=\xi}$ for an imaginary experiment in which the confounder $\underline{x}$ acts as a normal confounder that doesn't switch the treatment dose $\underline{d}$.

For example, d^σ might be whether an individual is admitted to Harvard Univ., y^σ might be how much money the individual earns for the first 20 years after graduating from Harvard, and x^σ might be his SAT scores. We assume Harvard only admits students with an SAT score higher than ξ.

56.1 PO analysis

The TPMs, printed in blue, for the bnet G_{disc} shown in Fig.56.1, are as follows. Note that the TPMs for the bnet G_{disc} are defined in terms of the TPMs for the bnet G.

$$P(x^\sigma) = \delta(x^\sigma, x) \tag{56.1}$$

$$P(d^\sigma | x^\sigma = x) = \begin{cases} \delta(d^\sigma, 1) & \text{for } x > \xi \\ \delta(d^\sigma, 0) & \text{for } x < \xi \end{cases} \tag{56.2}$$

$$P(y^\sigma | y^\sigma(0), y^\sigma(1), d^\sigma) = \mathbb{1}(y^\sigma = y^\sigma(d^\sigma)) \tag{56.3}$$

[1] ATE, which stands for "average treatment effect", is defined in Chapter 52.

Figure 56.1: 2 bnets used in the PO analysis of RDD. The TPMs for G_{disc} are defined in terms of the TPMs for G. The TPM $P(d^\sigma|x^\sigma)$ for G_{disc} is discontinuous in x^σ.

$$P(y^\sigma(c)|x) = P(y^\sigma(c)|d^\sigma, x) = \text{given} \tag{56.4}$$

Define

$$E_{\sigma|x}[y^\sigma(c)] = E_{\underline{y}(c)|x}[\underline{y}(c)] = \mathcal{Y}_{c|x} \tag{56.5}$$

and

$$\xi\pm = \xi \pm \epsilon \tag{56.6}$$

for some infinitesimal $\epsilon > 0$.

See Fig.56.2. In RDD, we assume that if we define the following 2 δ's, one for bnet G and the other for bnet G_{disc}, then the two δ's are equal, and they equal a conditional ATE.

$$\delta_{G_{disc}} = \mathcal{Y}_{1|x=\xi+} - \mathcal{Y}_{0|x=\xi-} \tag{56.7}$$

$$\delta_G = \mathcal{Y}_{1|x=\xi} - \mathcal{Y}_{0|x=\xi} \tag{56.8}$$

$$\delta_G = \delta_{G_{disc}} = \delta \tag{56.9}$$

$$\delta = ATE_{|x=\xi} \tag{56.10}$$

Figure 56.2: The jump δ between $\mathcal{Y}_{1|x}$ and $\mathcal{Y}_{0|x}$ is the same for G and G_{disc}.

56.2 Linear Regression

In this section, we show how to apply linear regression (LR) to the PO analysis of RDD.

y^σ can be fitted as a function of $x \in \mathbb{R}$, for $c^\sigma \in \{0, 1\}$, as follows. Here ϵ^σ is the residual for individual σ and $b_0, m_0, b_1, m_1 \in \mathbb{R}$ are the fit parameters.

$$y^\sigma = [b_0 + m_0(x - \xi)](1 - c^\sigma) + [b_1 + m_1(x - \xi)]c^\sigma + \epsilon^\sigma \ . \tag{56.11}$$

Note that Eq.(56.11) yields a straight line in the $y^\sigma - x$ plane for $c^\sigma = 0$, and another straight line for $c^\sigma = 1$. These 2 lines are colored magenta in Fig.56.2. We are using the standard symbols b to denote the y-intercept, and m to denote the slope of a straight line.

Taking the expected value of Eq.(56.11), we get

$$\mathcal{Y}_{c|x} = [b_0 + m_0(x - \xi)](1 - c) + [b_1 + m_1(x - \xi)]c \ . \tag{56.12}$$

Hence,

$$\mathcal{Y}_{1|x=\xi+} = b_1 \ , \ \ \mathcal{Y}_{0|x=\xi-} = b_0 \tag{56.13}$$

$$\delta = b_1 - b_0 \tag{56.14}$$

Chapter 57

Regret Based Decision Making: COMING SOON

Chapter 58

Reinforcement Learning (RL)

Figure 58.1: Axes for episode time and episode number.

I based this chapter on the following references. Refs.[10][21]

In RL, we consider an "agent" or robot that is learning.

Let $T \in \mathbb{Z}_{>0}$ be the duration time of an **episode** of learning. If $T = \infty$, we say that the episode has an infinite time horizon. A learning episode will evolve towards the right, for times $t = 0, 1, \ldots, T-1$. We will consider multiple learning episodes. The episode number will evolve from top to bottom. This is illustrated in Fig.58.1.

Let $\underline{s}_t \in S_{\underline{s}}$ for $t \in \mathbb{Z}_{[0,T-1]}$ be random variables that record the **state** of the agent at various times t.

Let $\underline{a}_t \in S_{\underline{a}}$ for $t \in \mathbb{Z}_{[0,T-1]}$ be random variables that record the **action** of the agent at various times t.

Figure 58.2: State-Action-Reward dynamical bnet

Let $\underline{\theta}_t \in S_{\underline{\theta}}$ for $t \in \mathbb{Z}_{[0,T-1]}$ be random variables that record the **policy parameters** at various times t.

For $\underline{X} \in \{\underline{s}, \underline{a}, \underline{\theta}\}$, define $\underline{X}$ followed by a dot to be the vector

$$\underline{X}. = [\underline{X}_0, \underline{X}_1, \ldots, \underline{X}_{T-1}] \ . \tag{58.1}$$

Also let

$$\underline{X}_{\geq t} = [\underline{X}_t, \underline{X}_{t+1}, \ldots, \underline{X}_{T-1}] \ . \tag{58.2}$$

Fig.58.2 shows the basic State-Action-Reward bnet for an agent that is learning. The TPMs for the nodes of Fig.58.2 are given in blue below:

$$P(a_t|s_t, \theta_t) = \text{given.} \tag{58.3}$$

$P(a_t|s_t, \theta_t)$ is called a **policy with parameter θ_t.**

$$P(s_t|s_{t-1}, a_{t-1}) = \text{given.} \tag{58.4}$$

$P(s_t|s_{t-1}, a_{t-1})$ is called the **TPM of the model**. $P(s_t|s_{t-1}, a_{t-1})$ reduces to $P(s_0)$ when $t = 0$.

$$P(r_t|s_t, a_t) = \delta(r_t, r(s_t, a_t))) \ . \tag{58.5}$$

$r : S_{\underline{s}} \times S_{\underline{a}} \to \mathbb{R}$ is a given **one-time reward function**.

Note that

$$P(s., a.|\theta.) = \prod_{t=0}^{T-1} \{P(s_t|s_{t-1}, a_{t-1}) P(a_t|s_t, \theta_t)\} \ . \tag{58.6}$$

Define the **all times reward** Σ by

$$\Sigma(s., a.) = \sum_{t=0}^{T-1} \gamma^t r(s_t, a_t) \ . \tag{58.7}$$

Here $0 < \gamma < 1$. γ, called the **discount rate**, is included to assure convergence of Σ when $T \to \infty$. If $r(s_t, a_t) < K$ for all t, then $\Sigma < K\frac{1}{1-\gamma}$.

Define the **objective (i.e. goal) function** $E\Sigma(\theta.)$ by

$$E\Sigma(\theta.) = E_{\underline{s}.,\underline{a}.|\theta.}\Sigma(\underline{s}., \underline{a}.) = \sum_{s.,a.} P(s., a.|\theta.)\Sigma(s., a.) \tag{58.8}$$

The goal of RL is to maximize the objective function over its parameters $\theta.$. The parameters $\theta^*.$ that maximize the objective function are the optimum strategy:

$$\theta.^* = \underset{\theta.}{\operatorname{argmax}}\, E\Sigma(\theta.) \tag{58.9}$$

Define a **future reward** for times $\geq t$ as:

$$\Sigma_{\geq t}((s_{t'}, a_{t'})_{t' \geq t}) = \sum_{t'=t}^{T-1} \gamma^{t'-t} r(s_{t'}, a_{t'}) \tag{58.10}$$

Define the following **expected conditional future rewards** (rewards for times $\geq t$, conditioned on certain quantities having given values):

$$v_t = v(s_t, a_t; \theta.) = E_{\underline{s}.,\underline{a}.|s_t,a_t,\theta.}[\Sigma_{\geq t}] \tag{58.11}$$

$$V_t = V(s_t; \theta.) = E_{\underline{s}.,\underline{a}.|s_t,\theta.}[\Sigma_{\geq t}] = E_{\underline{a}_t|s_t,\theta.}[v(s_t, \underline{a}_t; \theta.)] \tag{58.12}$$

v is usually called Q in the literature. We will refer to Q as v in order to follow a convention wherein an $\underline{a}_t$-average changes a lower case letter to an upper case one.

We will sometimes write $v(s_t, a_t)$ instead of $v(s_t, a_t; \theta.)$.

Since $E\Sigma_{\geq t}$ only depends on $\theta_{\geq t}$, $v(s_t, a_t; \theta.) = v(s_t, a_t; \theta_{\geq t})$, and $V(s_t; \theta.) = V(s_t; \theta_{\geq t})$.

Note that the objective function $E\Sigma$ can be expressed in terms of v_0 by averaging over its unaveraged parameters:

$$E\Sigma(\theta.) = E_{\underline{s}_0,\underline{a}_0|\theta_0} v(\underline{s}_0, \underline{a}_0; \theta.) \tag{58.13}$$

Define a **one-time reward** and an **expected conditional one-time reward** as:

$$r_t = r(s_t, a_t) \tag{58.14}$$

$$R_t = R(s_t; \theta_t) = E_{\underline{a}_t|s_t,\theta_t}[r(s_t, \underline{a}_t)] \quad . \tag{58.15}$$

Note that

$$\Sigma_{\geq t} = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots + \gamma^{T-1-t} r_{t+(T-1-t)} \tag{58.16}$$

$$= r_t + \gamma \Sigma_{\geq t+1}; . \tag{58.17}$$

If we take $E_{\underline{s}_\cdot,\underline{a}_\cdot|s_t,a_t,\theta_\cdot}[\cdot]$ of both sides of Eq.(58.17), we get

$$v_t = r_t + \gamma E_{\underline{s}_{t+1},\underline{a}_{t+1}|\theta_\cdot}[v_{t+1}] \ . \tag{58.18}$$

If we take $E_{\underline{s}_\cdot,\underline{a}_\cdot|s_t,\theta_\cdot}[\cdot]$ of both sides of Eq.(58.17), we get

$$V_t = R_t + \gamma E_{\underline{s}_{t+1}|\theta_\cdot}[V_{t+1}] \ . \tag{58.19}$$

Note that

$$\begin{aligned} \Delta r_t &= r_t - R_t & (58.20)\\ &= r_t - (V_t - \gamma E_{\underline{s}_{t+1}|\theta_\cdot}[V_{t+1}]) & (58.21)\\ &= r_t + \gamma E_{\underline{s}_{t+1}|\theta_\cdot}[V_{t+1}] - V_t \ . & (58.22) \end{aligned}$$

Define

$$\Delta v_t = v_t - V_t \ . \tag{58.23}$$

Note that

$$\Delta v_t = \Delta r_t \ . \tag{58.24}$$

Next, we will discuss 3 RL bnets

- exact RL bnet (exact, assumes policy is known)
- Actor-Critic RL bnet (approximate, assumes policy is known)
- Q function learning RL bnet (approximate, assumes policy is NOT known)

58.1 Exact RL bnet

An exact RL bnet is given by Fig.58.3.

Fig.58.3 is the same as Fig.58.2 but with more nodes added in order to optimize the policy parameters. Here are the TPMs, printed in blue, for the nodes not already discussed in connection to Fig.58.2.

$$P(\theta_t|\theta_\cdot) = \delta(\theta_t, (\theta_\cdot)_t) \tag{58.25}$$

$$\forall(s_t,a_t): \quad P(v_t(s_t,a_t)|r_t, v_{t+1}(\cdot),\theta_\cdot) = \delta(v_t(s_t,a_t), r_t + \gamma E_{\underline{s}_{t+1},\underline{a}_{t+1}|\theta_\cdot}[v_{t+1}]) \tag{58.26}$$

$$P(\theta_\cdot'|\theta_\cdot, v_0(\cdot)) = \delta(\theta'_\cdot, \theta_\cdot + \alpha\partial_{\theta_\cdot} \underbrace{E_{\underline{s}_0,\underline{a}_0|\theta_0} v(\underline{s}_0,\underline{a}_0;\theta_\cdot)}_{E\Sigma(\theta_\cdot)}) \tag{58.27}$$

Figure 58.3: Exact RL bnet. $v_t(\cdot)$ means the array $[v_t(s_t, a_t)]_{\forall s_t, a_t}$ The following arrows are implicit: for all t, arrow from $\underline{\theta}. \to \underline{v}_t(\cdot)$. We did not draw those arrows so as not to clutter the diagram.

$\alpha > 0$ is called the **learning rate**. This method of improving $\theta.$ is called gradient ascent.

Concerning the gradient of the objective function, note that

$$\begin{aligned}
\partial_{\theta_t} E\Sigma(\theta.) &= \sum_{s.,a.} \partial_{\theta_t} P(s.,a.|\theta.)\Sigma(s.,a.) && (58.28)\\
&= \sum_{s.,a.} P(s.,a.|\theta.)\partial_{\theta_t} \ln P(s.,a.|\theta.)\Sigma(s.,a.) && (58.29)\\
&= E_{\underline{s}.,\underline{a}.|\theta.}\left\{\partial_{\theta_t} \ln P(a_t|s_t,\theta_t)\Sigma(s.,a.)\right\} \ . && (58.30)
\end{aligned}$$

If we run the agent $nsam(\vec{s}_t)$ times and obtain samples $s_t[i], a_t[i]$ for all t and for $i = 0, 1, \ldots, nsam(\vec{s}_t) - 1$, we can express this gradient as follows:

$$\partial_{\theta_t} E\Sigma(\theta.) \approx \frac{1}{nsam(\vec{s}_t)} \sum_i \sum_{t=0}^{T-1} \partial_{\theta_t} \ln P(a_t[i] \mid s_t[i], \theta_t) r(s_t[i], a_t[i]) \ . \qquad (58.31)$$

The exact RL bnet Fig.58.3 is difficult to use to calculate the optimum parameters θ^*.. The problem is that $\underline{s}_t$ propagates towards the future and the $\underline{v}_t(\cdot)$ propagates towards the past, so we don't have a Markov Chain with a chain link for each t (i.e., a dynamical bnet) in the episode time direction. Hence, people have come up with approximate RL bnets that are doubly dynamical (i.e., dynamical along the episode time and episode number axes.) We discuss some of those approximate RL bnets next.

58.2 Actor-Critic RL bnet

For the actor-critic RL bnet, we approximate Eq.(58.31) by

$$\partial_{\theta_t} E\Sigma(\theta.) \approx \frac{1}{nsam(\vec{s})} \sum_i \sum_{t=0}^{T-1} \underbrace{\partial_{\theta_t} \ln P(a_t[i] \mid s_t[i], \theta_t)}_{Actor} \underbrace{\Delta r_t(s_t[i], a_t[i])}_{Critic} \qquad (58.32)$$

The actor-critic RL bnet is given by Fig.58.4. This bnet is approximate and assumes that the policy is known. The TPMs for its nodes are given in blue below.

$$P(\theta_t) = \text{given} \qquad (58.33)$$

$$P(s_t[i] \mid s_{t-1}[i], a_{t-1}[i]) = \text{ given} \qquad (58.34)$$

Figure 58.4: Actor-Critic RL bnet.

$$P(a_t[i] \mid s_t[i], \theta_t) = \text{ given} \tag{58.35}$$

$$P(r_t[i] \mid s_t[i], a_t[i]) = \delta(r_t[i], r(s_t[i], a_t[i])) \tag{58.36}$$

$r : S_{\underline{s}} \times S_{\underline{a}} \to \mathbb{R}$ is given.

$$P(\Delta v_t[i] \mid s_t[i], a_t[i], s_{t+1}[i]) = \delta(\Delta v_t[i], r(s_t[i], a_t[i]) + \gamma \hat{V}(s_{t+1}[i]; \phi') - \hat{V}(s_t[i]); \phi) \ . \tag{58.37}$$

$$P(\theta'.) = \delta(\theta'., \theta_t + \alpha \partial_{\theta_t} \sum_i \ln P(a_t[i] \mid s_t[i], \theta_t) \Delta v_t[i]) \tag{58.38}$$

$\hat{V}(s_t[i]); \phi)$ is obtained by curve fitting (see Chapter 5) using samples $(s_t[i], a_t[i])$ $\forall t, i$ with

$$y[i] = \sum_{t'=t}^{T} r(s_{t'}[i], a_{t'}[i]) \tag{58.39}$$

Figure 58.5: Q function learning RL bnet.

and

$$\hat{y}[i] = \hat{V}(s_t[i]; \phi) \ . \tag{58.40}$$

Eq.(58.39) is an approximation because $(s_{t'}, a_{t'})_{t'>t}$ are averaged over in the exact expression for $V(s_t)$. $\hat{V}(s_{t+1}[i]); \phi')$ is obtained in the same way as $\hat{V}(s_t[i]); \phi)$ but with t replaced by $t+1$ and ϕ by ϕ'.

58.3 Q function learning RL bnet

The Q-function learning RL bnet is given by Fig.58.5. This bnet is approximate and assumes that the policy is NOT known. The TPMs for its nodes are given in blue below. (Remember that $Q = v$).

$$P(s_t|s_{t-1}, a_{t-1}) = \text{ given} \tag{58.41}$$

$$P(a_t|s_t, v_t(\cdot)) = \delta(a_t, \underset{a}{\text{argmax}}\, v_t(s_t, a)) \tag{58.42}$$

$$P(r_t|s_t, a_t) = \delta(r_t, r(s_t, a_t)) \tag{58.43}$$

$r : S_{\underline{s}} \times S_{\underline{a}} \to \mathbb{R}$ is given.

$$\begin{aligned} \forall (s_t, a_t): \quad & P(v_t(s_t, a_t)|v_{t-1}(\cdot)) = \\ & = \quad \delta(v_t(s_t, a_t), r(s_t, a_t) + \gamma \text{max}_a E_{\underline{s}_{t+1}|s_t, a_t} v_{t-1}(\underline{s}_{t+1}, a)) \end{aligned} \tag{58.44}$$

Figure 58.6: Q function learning RL bnet. Same as Fig.58.5 but with new arrow passing s_t to Q_{t-1}.

This value for $v_t(s_t, a_t)$ approximates $v_t = r_t + \gamma E_{\underline{s}_{t+1}, \underline{a}_{t+1}} v_{t+1}$.

Some people use the bnet of Fig.58.6) instead of Fig.58.5 and replace Eq.(58.44) by

$$\begin{aligned} \forall (s_t, a_t): \quad & P(v_t(s_t, a_t) | s_{t+1}, v_{t-1}(\cdot)) = \\ & = \quad \delta(v_t(s_t, a_t), r(s_t, a_t) + \gamma \max_a v_{t-1}(s_{t+1}, a)) \ . \end{aligned} \tag{58.45}$$

Chapter 59

Reliability Box Diagrams and Fault Tree Diagrams

This chapter is based on Refs.[43] and [58].

In this chapter, we assume that reader is familiar with Boolean Algebra. See Chapter Notational Conventions and Preliminaries for a quick review of what we recommend that you know about Boolean Algebra to fully appreciate this chapter.

Figure 59.1: Example of rbox diagram.

Figure 59.2: An ftree diagram equivalent to Fig.59.1. It represents $e = (\phi_1 \wedge \phi_3) \vee (\phi_2 \wedge \phi_3)$.

Complicated devices with a large number of components such as airplanes or rockets can fail in many ways. If their performance depends on some components

Figure 59.3: How to map an rbox diagram to a bnet.

Figure 59.4: bnet corresponding the rbox diagram Fig.59.1.

working in series and one of the components in the series fails, this may lead to catastrophic failure. To avert such disasters, engineers use equivalent components connected in parallel instead of in series, thus providing multiple backup systems. They analyze the device to find its weak points and add backup capabilities there. They also estimate the average time to failure for the device.

The two most popular diagrams for finding the failure modes and their rates for large complicated devices are

- rbox diagrams = Reliability Box diagrams. See Fig.59.1 for an example.
- ftree diagrams = Fault Tree Diagrams. See Fig.59.2 for an example.

In an ftree diagram, several nodes might stand for the same component of a physical

device. In an rbox diagram, on the other hand, each node represents a distinct component in a device. Hence, rbox diagrams resemble the device they are addressing whereas ftree diagrams don't. Henceforth, we will refer to this desirable property as **physical resemblance**.

As we will show below with an example, it is pretty straightforward to translate an rbox to an ftree diagram. Going the other way, translating an ftree to an rbox diagram is much more difficult.

Next we will define a new kind of bnet that we will call a failure bnet that has physical resemblance. Then we will describe a simple method of translating (i.e., mapping) any rbox diagram to a failure bnet. Then we will show how a failure bnet can be used to do all the calculations that are normally done with an rbox or an ftree diagram. In that sense, failure bnets seem to afford all the benefits of both ftree and rbox diagrams.

A **failure bnet** contains nodes of 5 types, labeled $\underline{b}$, $\underline{e}$, $\underline{x}_i$, $\underline{\phi}_i$, and $\underline{A}_i$. All nodes have only two possible states $S = Success = 0$, $F = Failure = 1$.

1. The bnet has a beginning node labeled $\underline{b}$ which is always set to success. The $\underline{b}$ node and the $\underline{\phi}_i$ nodes are the only root nodes of the bnet.

2. The bnet has a single leaf node, the end node, labeled $\underline{e}$. $\underline{e}$ is fixed. In rbox diagrams, $\underline{e} = S$ whereas in ftree diagrams, $\underline{e} = F$.

3. $\underline{x}^{nx} = (\underline{x}_0, \underline{x}_1, \ldots, \underline{x}_{nx-1})$. $\underline{x}_i \in \{S, F\}$ for all i.

   Suppose $\underline{x}_i$ has parents $\underline{\phi}_i$ and $\underline{a}^{na} = (\underline{a}_0, \underline{a}_1, \ldots \underline{a}_{na-1})$. Then the TPM of node $\underline{x}_i$ is defined to be

$$P(x_i|\phi_i, a^{na}) = \delta(x_i, \phi_i \vee \vee_{i=0}^{na-1} a_i) \tag{59.1}$$

4. For each node $\underline{x}_i$, the bnet has a "performance" root node $\underline{\phi}_i \in \{0, 1\}$ with an arrow pointing from it to $\underline{x}_i$ (i.e, $\underline{\phi}_i \rightarrow \underline{x}_i$). For all i,

$$P(\phi_i) = \epsilon_i \delta(\phi_i, F) + \overline{\epsilon}_i \delta(\phi_i, S) \ . \tag{59.2}$$

   ϵ_i is the failure probability and $\overline{\epsilon}_i = 1 - \epsilon_i$ is the success probability. We name the failure probability ϵ_i because it is normally very small. It is usually set to $1 - e^{-\lambda_i t} \approx \lambda_i t$ when $\lambda_i t << 1$, where λ_i is the failure rate for node $\underline{x}_i$ and t stands for time. The rblock literature usually calls $\overline{\epsilon_i} = R_i$ the **reliability** of node $\underline{x}_i$, and $\epsilon_i = (1 - R_i) = F_i$ its **unreliability**.

5. The nodes $\underline{A}_i \in \{0, 1\}$ are simply AND gates. If $\underline{A}_i$ has inputs $\underline{y}^{ny} = (\underline{y}_0, \underline{y}_1, \ldots, \underline{y}_{ny-1})$, then the TPM of $\underline{A}_i$ is

$$P(A_i|y^{ny}) = \delta(A_i, \wedge_{i=0}^{ny-1} y_i) \ . \tag{59.3}$$

An instance (instantiation) of a bnet is the bnet with all nodes set to a specific state. A **realizable instance (r-instance)** of a bnet is one which has non-zero probability.

Fig.59.3 shows how to translate any rbox diagram to a failure bnet. To illustrate this procedure, we translated the rbox diagram Fig.59.1 into the failure bnet Fig.59.4.

For the failure bnet Fig.59.4, one has:

$$\begin{array}{l} P(b) = \mathbb{1}(b = 0) \\ P(x_1|\phi_1, b) = \mathbb{1}(x_1 = \phi_1 \vee b) \\ P(x_2|\phi_2, x_1) = \mathbb{1}(x_2 = \phi_2 \vee x_1) \\ P(x_3|\phi_3, b) = \mathbb{1}(x_3 = \phi_3 \vee b) \\ P(A|x_2, x_3)e = \mathbb{1}(x_2 \wedge x_3) \\ P(e|A) = \mathbb{1}(e = A) \end{array} . \tag{59.4}$$

Therefore, all r-instances of this bnet must satisfy

$$e = (\phi_1 \vee \phi_2) \wedge \phi_3 \tag{59.5}$$

$$= (\phi_1 \wedge \phi_3) \vee (\phi_2 \wedge \phi_3) . \tag{59.6}$$

Eq.(59.6) proves that Fig.59.2 is indeed a representation of Fig.59.1.

Next, we consider r-instances of this bnet for two cases: $e = S$ and $e = F$.

- **rblock analysis:** $e = S = 0$**.**
  Table 59.1 shows the probability of all possible r-instances that end in success for the failure bnet Fig.59.4. (These r-instances are the main focus of rblock analysis). The first 4 of those probabilities (those with $\phi_3 = 0$) sum to $\overline{\epsilon}_3$ so the sum $P(e = S)$ of all 5 is

  $$P(e = S) = \overline{\epsilon}_3 + \overline{\epsilon}_1\overline{\epsilon}_2\epsilon_3 , \tag{59.7}$$

  or, expressing it in reliability language in which $\overline{\epsilon} = R$,

  $$P(e = S) = R_3 + R_1R_2\overline{R}_3 . \tag{59.8}$$

- **ftree analysis:** $e = F = 1$**.**
  Table 59.2 shows the probability of all possible r-instances that end in failure for the failure bnet Fig.59.4. (These r-instances are the main focus of ftree analysis). If we set $\epsilon_i = \epsilon$ and $\overline{\epsilon}_i \approx 1$ for $i = 1, 2, 3$, then the first two of those r-instances have probabilities of $order(\epsilon^2)$ and the third has probability of $order(\epsilon^3)$. The two lowest order ($order(\epsilon^2)$) r-instances are called the "minimal

instance	probability
	$\overline{\epsilon}_1\epsilon_2\overline{\epsilon}_3$
	$\epsilon_1\overline{\epsilon}_2\overline{\epsilon}_3$
	$\epsilon_1\epsilon_2\overline{\epsilon}_3$
	$\overline{\epsilon}_1\overline{\epsilon}_2\overline{\epsilon}_3$
	$\overline{\epsilon}_1\overline{\epsilon}_2\epsilon_3$

Table 59.1: Probabilities of all possible r-instances with $e = S = 0$ for failure bnet Fig.59.4.

instance	probability
	$\bar{\epsilon}_1\epsilon_2\epsilon_3$
	$\epsilon_1\bar{\epsilon}_2\epsilon_3$
	$\epsilon_1\epsilon_2\epsilon_3$

Table 59.2: Probabilities of all possible r-instances with $e = F = 1$ for the failure bnet Fig.59.4.

cut sets" of the ftree. We will have more to say about minimal cut sets later on. For now, just note from Eq.(59.6) that the ftree Fig.59.2 is just the result of joining together with ORs two expressions, one for each of the two minimal cut sets.

More general $\underline{x}_i$.

Failure bnets can actually accommodate $\underline{x}_i$ nodes of a more general kind than what we first stipulated. Here are some possibilities:

For any $a^n \in \{0,1\}^n$, let

$$len(a^n) = \sum_i a_i \tag{59.9}$$

- **OR gate**

$$\begin{aligned} P(x_i|\phi_i, a^{na}) &= \delta(x_i, \phi_i \vee \vee_j a_j) && (59.10)\\ &= \delta(x_i, \phi_i \vee \mathbb{1}(len(a^{na}) > 0)) && (59.11) \end{aligned}$$

- **AND gate**

$$P(x_i|\phi_i, a^{na}) = \delta(x_i, \phi_i \vee \wedge_j a_j) \tag{59.12}$$
$$= \delta(x_i, \phi_i \vee \mathbb{1}(len(a^{na}) = na)) \tag{59.13}$$

- **Fail if least K failures (less than K successes)**

$$P(x_i|\phi_i, a^{na}) = \delta(x_i, \phi_i \vee \mathbb{1}(len(a^{na}) \geq K)) \tag{59.14}$$

- **Fail if less than K failures (at least K successes)**

$$P(x_i|\phi_i, a^{na}) = \delta(x_i, \phi_i \vee \mathbb{1}(len(a^{na}) < K)) \tag{59.15}$$

- **Fail if exactly one failure**

$$P(x_i|\phi_i, a^{na}) = \delta(x_i, \phi_i \vee \mathbb{1}(len(a^{na}) = 1)) \tag{59.16}$$

  This equals an XOR (exclusive OR) gate when $na = 2$.

- **General gate**
  $f : \{0,1\}^{na} \to \{0,1\}$

$$P(x_i|\phi_i, a^{na}) = \delta(x_i, \phi_i \vee f(a^{na})) \tag{59.17}$$

59.1 Minimal Cut Sets

Suppose $x \in \{0,1\}$ and $f : \{0,1\} \to \{0,1\}$. Then by direct evaluation, we see that

$$f(x) = [\overline{x} f(0)] \vee [x f(1)] \,. \tag{59.18}$$

Let

$$\begin{aligned} !x &= 1 - x, \\ !^0 x &= x, \\ !^1 x &= !x \end{aligned} \tag{59.19}$$

Then Eq.(59.18) can be rewritten as

$$f(x) = \vee_{a\in\{0,1\}}[(!^{\overline{a}}x)f(a)] \ . \tag{59.20}$$

Now suppose $x^n \in \{0,1\}^n$ and $f : \{0,1\}^n \to \{0,1\}$. Eq.(59.20) generalizes to

$$f(x^n) = \vee_{a^n\in\{0,1\}^n}[\prod_i (!^{\overline{a}_i}x_i)f(a^n)] \ . \tag{59.21}$$

Eq.(59.21) is called an **ors-of-ands** normal form expansion. There is also an **ands-of-ors** normal form expansion obtained by swapping multiplication and $\vee$ in Eq.(59.21), but we won't need it here.

A **cut set** is a set of ϕ_i's such that if they are all equal to F, then $e = F$ for all the r-instances. A **minimal cut set** is a cut set such that there are no larger cut sets that contain it. From the failure bnet, we can always find a function $f : \{0,1\}^{nx} \to \{0,1\}$ such that $e = f(\phi^{nx})$ for all the r-instances. We did that for our example failure bnet and obtained Eq.(59.6). We can then express $f(\phi^{nx})$ as an ors-of-ands expansion to find all the minimal cut sets. The ands terms in that ors-of-ands expansion each gives a different minimal cut set, after some simplification. The ors-of-ands expression is not unique and it may be necessary to simplify (using the Boolean Algebra identities given in Chapter Notational Conventions and Preliminaries) to remove those redundancies.

Chapter 60

Restricted Boltzmann Machines

In what follows, we will abbreviate "restricted Boltzmann machine' by rebo.

Let
$v \in \{0,1\}^{nv}$
$h \in \{0,1\}^{nh}$
$b \in \mathbb{R}^{nv}$ (mnemonic, v and b sound the same)
$a \in \mathbb{R}^{nh}$
$W^{v|h} \in \mathbb{R}^{nv \times nh}$
Energy:

$$E(v,h) = -(b^T v + a^T h + v^T W^{v|h} h) \tag{60.1}$$

Boltzmann distribution:

$$P(v,h) = \frac{e^{-E(v,h)}}{Z} \tag{60.2}$$

Partition function:

$$Z = \sum_{v,h} e^{-E(v,h)} = Z(a,b,W^{v|h}) \tag{60.3}$$

$$\begin{aligned} P(v|h) &= \frac{e^{b^T v + a^T h + v^T W^{v|h} h}}{\sum_v e^{b^T v + a^T h + v^T W^{v|h} h}} && (60.4) \\ &= \frac{e^{b^T v + v^T W^{v|h} h}}{\sum_v e^{b^T v + v^T W^{v|h} h}} && (60.5) \\ &= \prod_i \frac{e^{v_i (b_i + \sum_j W_{i,j}^{v|h} h_j)}}{\sum_{v_i=0,1} e^{v_i (b_i + \sum_j W_{i,j}^{v|h} h_j)}} && (60.6) \\ &= \prod_i P(v_i|h) && (60.7) \end{aligned}$$

$$P(v_i|h) = \frac{e^{v_i (b_i + \sum_j W_{i,j}^{v|h} h_j)}}{Z_i(h)} \tag{60.8}$$

Figure 60.1: bnet for a Restricted Boltzmann Machine (rebo) with $nv = 3$

Eq.(60.8) implies that a rebo can be represented by the bnet Fig.60.1.
Let

$$x_i = b_i + \sum_j W_{ij}^{v|h} h_j \ . \tag{60.9}$$

Then

$$\begin{aligned} P(v_i = 1|h) &= \frac{e^{x_i}}{1 + e^{x_i}} && (60.10) \\ &= \frac{1}{1 + e^{-x_i}} && (60.11) \\ &= \text{smoid}(x_i) \ . && (60.12) \end{aligned}$$

One could also expand the node $\underline{h}$ in Fig.60.1 into nh nodes. But note that $P(h) \neq \prod_j P(h_j)$ so there would be arrows among the h_j nodes.

Note that the rebo bnet is a special case of Naive Bayes (See Chapter 47) with $v_i, h_i \in \{0, 1\}$ and specific $P(h)$ and $P(v_i|h)$ node matrices.

Chapter 61

ROC curves

This chapter is based on Ref.[116].

ROC stands for **Receiver Operating Characteristic**. ROC curves are used in binary classification (BC).

To do BC, we are given the value $x \in \mathbb{R}$ for an individual. From this, we want to decide whether that individual has $a = 0$ or $a = 1$. The decision will depend on the value of a **threshold parameter** $\tau \in \mathbb{R}$.

$$\underline{x} \longleftarrow \underline{a}$$

Figure 61.1: bnet for BC.

Fig.61.1 shows the bnet used for BC.

Figure 61.2: x-distribution for two hypotheses $a = 0, 1$.

Fig.61.2 is a plot of $P(x|a)$, i.e., the TPM for node $\underline{x}$ of the bnet in Fig.61.1.

Whereas a is binary, x is continuous. But we can replace x by a binary variable

$$b = \mathbb{1}(x > \tau) \ . \tag{61.1}$$

$P(b|a)$ for $b, a \in \{0, 1\}$ is called the **confusion matrix** or **contingency table** for BC. The confusion matrix can be calculated from the TPM $P(x|a)$. Fig.61.3 illustrates

		Actual Value (a)	
		0	1
Predicted Value (x)	0	TNR	FNR
	1	FPR	TPR

Figure 61.3: The confusion matrix $P(b|a)$ for BC.

the confusion matrix $P(b|a)$ for BC. In that figure, the rates R are defined as follows.[1]

- **True Negative Rate (TNR)**

$$R_{0|0}(\tau) = P(x < \tau | a = 0) = \int_{x<\tau} dx\ P(x|a = 0) \tag{61.2}$$

- **False Positive Rate (FPR)**

$$R_{1|0}(\tau) = 1 - R_{0|0}(\tau) \tag{61.3}$$

$$= P(x > \tau | a = 0) = \int_{x>\tau} dx\ P(x|a = 0) \tag{61.4}$$

In Hypothesis Testing, $R_{1|0}$ is called the **p-value that** $\underline{x} > \tau$ **assuming curve** 0 **is the null hypothesis**.

- **False Negative Rate (FNR)**

$$R_{0|1}(\tau) = P(x < \tau | a = 1) = \int_{x<\tau} dx\ P(x|a = 1) \tag{61.5}$$

In Hypothesis Testing, $R_{0|1}$ is called the **p-value that** $\underline{x} < \tau$ **assuming curve** 1 **is the null hypothesis**.

- **True Positive Rate (TPR)**

$$R_{1|1}(\tau) = 1 - R_{0|1}(\tau) \tag{61.6}$$

$$= P(x > \tau | a = 1) = \int_{x>\tau} dx\ P(x|a = 1) \tag{61.7}$$

The **Receiver Operating Characteristic (ROC)** is a parametric plot with $X = R_{1|0}(\tau)$ and $Y = R_{1|1}(\tau)$, where $\tau \in \mathbb{R}$. The **Area Under the Curve (AUC)** is the area under the ROC. Fig.61.4 shows an example of a ROC and its AUC.

[1] I find the notation $x|a$ where $x, a \in \{0, 1\}$ much clearer than $\alpha\beta$ where $\alpha = T, F$ and $\beta = N, P$. Note that $\alpha = \mathbb{1}(x = a)$ and $\beta = x$, if we identify $0 = F = N$ and $1 = T = P$.

Figure 61.4: Example of ROC. Green shaded area is the AUC of the ROC.

Figure 61.5: ROC curves for 3 different separations between the 0 and 1 x-distributions.

Fig.61.5 shows situations that give AUC=.5 (random classifier), AUC=.85, and AUC=1 (perfect classifier). It's also possible to get an $AUC \in [0, 0.5]$, but we will ignore those models because they are useless for BC.

Note that

$$
\begin{aligned}
AUC &= \int_{x=0}^{1} d\tau \;\; R_{1|1}(\tau) \frac{dR_{1|0}(\tau)}{d\tau} && (61.8)\\
&= \int_{\infty}^{-\infty} d\tau \;\; \left\{ \int_{-\infty}^{\infty} dx \;\; \mathbb{1}(x > \tau) P(x|a=1) \right\} (-1) P(x = \tau | a = 0) && (61.9)\\
&= \int_{-\infty}^{\infty} dx' \;\; \int_{-\infty}^{\infty} dx \;\; \mathbb{1}(x > x') P(x|a=1) P(x'|a=0) \; . && (61.10)
\end{aligned}
$$

61.1 Terminology Table Adapted from Wikipedia Ref.[116]

Let $N_{x|a}$ be numbers (counts) so that

$$P(x|a) = \frac{N_{x|a}}{\sum_{x'} N_{x'|a}} \tag{61.11}$$

for all $x, a \in \{0, 1\}$.

condition positive (P): $N_{|1} = \sum_x N_{x|1}$, the number of real positive cases in the data

condition negative (N): $N_{|0} = \sum_x N_{x|0}$, the number of real negative cases in the data

true positive (TP): $N_{1|1}$, hit

true negative (TN): $N_{0|0}$, correct rejection

false positive (FP): $N_{1|0}$, false alarm, type I error or underestimation

false negative (FN): $N_{0|1}$, miss, type II error or overestimation

sensitivity, recall, hit rate, or true positive rate (TPR):

$$TPR = R_{1|1} = \frac{N_{1|1}}{N_{|1}} = \frac{N_{1|1}}{N_{1|1} + N_{0|1}} = 1 - R_{0|1} \tag{61.12}$$

specificity, selectivity or true negative rate (TNR):

$$TNR = R_{0|0} = \frac{N_{0|0}}{N_{|0}} = \frac{N_{0|0}}{N_{0|0} + N_{1|0}} = 1 - R_{1|0} \tag{61.13}$$

precision or positive predictive value (PPV):

$$PPV = \frac{N_{1|1}}{N_{1|1} + N_{1|0}} = 1 - FDR \tag{61.14}$$

negative predictive value (NPV):

$$NPV = \frac{N_{0|0}}{N_{0|0} + N_{0|1}} = 1 - FOR \tag{61.15}$$

miss rate or false negative rate (FNR):

$$FNR = R_{0|1} = \frac{N_{0|1}}{N_{|1}} = \frac{N_{0|1}}{N_{0|1} + N_{1|1}} = 1 - R_{1|1} \tag{61.16}$$

fall-out or false positive rate (FPR):

$$FPR = R_{1|0} = \frac{N_{1|0}}{N_{|0}} = \frac{N_{1|0}}{N_{1|0} + N_{0|0}} = 1 - R_{0|0} \tag{61.17}$$

false discovery rate (FDR):

$$FDR = \frac{N_{1|0}}{N_{1|0} + N_{1|1}} = 1 - PPV \tag{61.18}$$

false omission rate (FOR):

$$FOR = \frac{N_{0|1}}{N_{0|1} + N_{0|0}} = 1 - NPV \tag{61.19}$$

accuracy (ACC):

$$ACC = \frac{N_{1|1} + N_{0|0}}{N_{|1} + N_{|0}} = \frac{N_{1|1} + N_{0|0}}{N_{1|1} + N_{0|0} + N_{1|0} + N_{0|1}} \tag{61.20}$$

balanced accuracy (BA):

$$BA = \frac{R_{1|1} + R_{0|0}}{2} \tag{61.21}$$

F1 score is the harmonic mean of precision and sensitivity:

$$F_1 = 2 \times \frac{PPV \times R_{1|1}}{PPV + R_{1|1}} = \frac{2N_{1|1}}{2N_{1|1} + N_{1|0} + N_{0|1}} \tag{61.22}$$

Chapter 62

Scoring the Nodes of a Learned Bnet

Chapter 66 discusses how to learn a bnet from data. Many algorithms for doing this require scoring how well a particular bnet fits the data. This chapter is an introduction to such scoring.

Normally, each node of a bnet is scored separately, and then those node scores are summed to get the bnet score.

In this chapter, scores are defined so that a higher score means a better fit. By taking the negative of such a score, one can always get a score such that a lower score means a better fit.

There are 2 main types of bnet scores: Maximum Likelihood (ML) scores, and Shannon Information Theory (SIT) scores. ML scores consist of the log of a maximum likelihood function $P(\vec{x}|\theta)$ for i.i.d. samples $\vec{x} = (x[\sigma])_{\sigma=0,1,\ldots,nsam-1}$, where $x[\sigma] \sim P_{\underline{x}|\theta}(x|\theta)$:

$$\begin{aligned}
\text{ML-score} &= \ln(P(\vec{x}|\theta)) && (62.1)\\
&= \ln \prod_{\sigma} P(x[\sigma] \mid \theta) && (62.2)\\
&= \sum_{\sigma} \ln P(x[\sigma] \mid \theta) && (62.3)\\
&\approx nsam \sum_{x} P(x|\theta) \ln P(x|\theta) && (62.4)\\
&= -nsamH(P_{\underline{x}|\theta}) \ , && (62.5)
\end{aligned}$$

and SIT scores consist of a negative entropy:

$$\text{Info-score} = -H(P_{\underline{x}|\theta}) \ . \qquad (62.6)$$

Thus, up to a factor of $nsam$, they are the same thing. Maximizing a log likelihood function for i.i.d. samples or minimizing the corresponding entropy, are the same

thing, and they both yield a good estimate of the hidden parameters θ.

62.1 Probability Distributions and Special Functions

While writing this chapter, I briefly consulted the following Wikipedia articles about the definitions and properties of certain probability distributions and special functions.

- Categorical Distribution, Ref.[72]
- Multinomial Distribution, Ref.[108]
- Dirichlet Distribution, Ref.[79]
- Multivariate Normal Distribution, Ref.[110]
- Beta function, Ref.[68]
- Multinomial Coefficients, Ref.[109]
- Gamma Function Ref.[84]

Here are a few results from those Wikipedia articles that we will use later on in this chapter.

Below, we will abbreviate $q_+ = \sum_i q_i$, and $q. = (q_0, q_1, \ldots, q_{nq-1})$ for various quantities q

Gamma function. If $n > 0$ is an integer,

$$\Gamma(n+1) = n! \tag{62.7}$$

The **multivariate Beta function** is defined by

$$B(\alpha.) = \frac{\prod_k \Gamma(\alpha_k)}{\Gamma(\alpha_+)} \tag{62.8}$$

where $\alpha_k > 0$ for all k.

The **multinomial coefficient** is defined by

$$C(N.) = \frac{N_+!}{\prod_k N_k!} \tag{62.9}$$

where N_k are non-negative integers.

The **inverse of the multinomial coefficient** will be denoted by

$$CI(N.) = \frac{1}{C(N.)} = \frac{\prod_k N_k!}{N_+!} \tag{62.10}$$

The **Categorical Distribution** is defined by

$$Cat(x;\pi.) = \pi_x = \prod_k \pi_k^{\mathbb{1}(k=x)} \tag{62.11}$$

for $k, x \in S_{\underline{x}}$, where $\pi.$ is a probability dist.(i.e., $\pi_k \geq 0$ for all k, and $\pi_+ = 1$).

The **Multinomial Distribution** is defined by

$$Mul(N.;\pi., N) = C(N.) \ , \tag{62.12}$$

where N_k is a non-negative integer for all k, $N_+ = N$, and $\pi.$ is a probability dist. Mul() satisfies:

$$E[\underline{N_k}] = N\pi_k \ . \tag{62.13}$$

The **Dirichlet Distribution** is defined by

$$Dir(\pi.;\alpha.) = \frac{1}{B(\alpha.)} \prod_k \pi_k^{\alpha_k - 1} \tag{62.14}$$

where $\alpha_k > 0$ for all k, and $\pi.$ is a probability dist. The $\alpha.$ are called **concentration parameters** or **hyperparameters**. Dir() satisfies:

$$E[\underline{\pi_k}] = \frac{N_k}{N_+} \ . \tag{62.15}$$

Dir() is **conjugate prior** of Mul()

Note that

$$Mul(N.;\pi., N)Dir(\pi.;\alpha.) = \mathcal{K}(N., \alpha.)Dir(\pi.; N. + \alpha.) \ , \tag{62.16}$$

where

$$\mathcal{K}(N., \alpha.) = \frac{B(N. + \alpha.)}{CI(N.)B(\alpha.)} \ . \tag{62.17}$$

Dir() is replaceable by a Mul() for large concentration parameters

Note that if N_k is a positive integer and $\alpha_k = N_k + 1$ for all k, then

$$\begin{aligned} Dir(\pi.; \alpha_k = N_k + 1) &= C(N.) \prod_k \pi_k^{N_k} & (62.18) \\ &= Mul(N.;\pi., N_+) \ . & (62.19) \end{aligned}$$

62.2 Single node with no parents

In this section, we consider a learned bnet consisting of a single node with no parents. We will consider arbitrary learned bnets in the next section. But we start with this simplified case so as to reduce the number of indices in most quantities from 3 to 1. All the results that we derive in this section will be used in the next section after adding the extra indices. This way, we will avoid carrying the extra indices throughout the intermediate steps of many derivations.

For state $k \in \{0, 1, \ldots, nk-1\}$ of a single node $\underline{x}$, let
$\underline{N}_k$= current count number (an integer, data)
$\underline{\pi}.$= a probability dist, the TPM for the node
$\underline{\alpha}_k$= prior count number

$$\underline{N}. \longleftarrow \underline{\pi.} \longleftarrow \underline{\alpha.}$$

Figure 62.1: For a bnet consisting of a single node with no parents, this is a Markov chain of current counts ($\underline{N}.$), TPM ($\underline{\pi.}$), and prior counts ($\underline{\alpha.}$) .

Consider the Markov chain bnet of Fig.62.1, with the following TPMs, given in blue.

$$P(N.|\pi.) = Mul(N.; \pi., N_+) \tag{62.20}$$

$$P(\pi.|\alpha.) = Dir(\pi.; \alpha.) \tag{62.21}$$

It follows that

$$\begin{aligned} P(N., \pi.|\alpha.) &= P(N.|\pi.)P(\pi.|\alpha.) && (62.22)\\ &= Mul(N.; \pi., N_+)Dir(\pi.; \alpha.) && (62.23)\\ &= \mathcal{K}(N., \alpha.)Dir(\pi.; N. + \alpha.) \ . && (62.24) \end{aligned}$$

From Eq.(62.15) for the expected value of Dir(), we get

$$\hat{\pi}. = E[\underline{\pi}.] = \frac{N. + \alpha.}{N_+ + \alpha_+} \ . \tag{62.25}$$

Integrating both sides of Eq.(62.24) over $\pi.$, we find that

$$P(N.|\alpha.) = \mathcal{K}(N., \alpha.) \ . \tag{62.26}$$

If $N_k >> 1$ for all k, then the Dir() in Eq.(62.24) can be replaced by a Mul()

$$P(N., \pi.|\alpha.) \approx \mathcal{K}(N., \alpha.)Mul(N. + \alpha.; \pi., N_+ + \alpha_+) \ . \tag{62.27}$$

Therefore,

$$P(N.|\pi., \alpha.) = \frac{P(N., \pi.|\alpha.)}{P(N.|\alpha.)} \tag{62.28}$$

$$= Mul(N. + \alpha.; \pi., N_+ + \alpha_+) \ . \tag{62.29}$$

Claim 63

$$\ln P(N.|\hat{\pi}., \alpha.) = -(N_+ + \alpha_+) H\left(\frac{N. + \alpha.}{N_+ + \alpha_+}\right) + \ln C(N. + \alpha.) \tag{62.30}$$

$$> -(N_+ + \alpha_+) H\left(\frac{N. + \alpha.}{N_+ + \alpha_+}\right) - \frac{1}{2}(nk - 1) \ln N_+ \tag{62.31}$$

proof:

$$\ln P(N.|\hat{\pi}., \alpha.) = \sum_k (N_k + \alpha_k) \ln \hat{\pi}_k + \ln C(N. + \alpha.) \tag{62.32}$$

$$= \sum_k (N_k + \alpha_k) \ln \frac{N_k + \alpha_k}{N_+ + \alpha_+} + \ln C(N. + \alpha.) \tag{62.33}$$

$$= -(N_+ + \alpha_+) H\left(\frac{N. + \alpha.}{N_+ + \alpha_+}\right) + \ln C(N. + \alpha.) \tag{62.34}$$

Recall Stirling's approximation of a factorial, valid for large integers n:

$$\ln n! \approx (n + \frac{1}{2}) \ln n - n \ . \tag{62.35}$$

Assume $N_k >> 1$ for all k. Applying Stirling's approximation to all factorials in $C(N)$, we get

$$\ln C(N.) \approx (N_+ + \frac{1}{2}) \ln N_+ - N_+ - \sum_k \left[(N_k + \frac{1}{2}) \ln N_k - N_k\right] \tag{62.36}$$

$$= (N_+ + \frac{1}{2}) \ln N_+ - \sum_k (N_k + \frac{1}{2}) \ln N_k \ . \tag{62.37}$$

Next assume that

$$N_k \approx \frac{N_+}{nk} \ . \tag{62.38}$$

Then

$$\ln C(N.) = (N_+ + \frac{1}{2})\ln N_+ - nk(\frac{N_+}{nk} + \frac{1}{2})[\ln N_+ - \ln nk] \tag{62.39}$$

$$= -\frac{1}{2}(nk-1)\ln N_+ + (N_+ + \frac{nk}{2})\ln nk \tag{62.40}$$

$$> -\frac{1}{2}(nk-1)\ln N_+ \ . \tag{62.41}$$

QED

62.3 Multiple nodes with any number of parents

In the previous section, we considered a bnet consisting of a single node with no parents, so we only needed a single index k for the states of the single node. In this section, we consider an arbitrary bnet with multiple nodes each of which may have multiple parents. Most of the results in the previous section are valid for the general case if we make the following replacements: $\pi. \to \pi^i_{\cdot|\mu}$ $N. \to N^i_{\cdot,\mu}$ $\alpha. \to \alpha^i_{\cdot,\mu}$. Upon this replacement, Fig.62.1 becomes Fig.62.2. The TPMs, printed in blue, of the new Markov chain, are as follows:

$$\underline{N^i_{\cdot,\mu}} \longleftarrow \underline{\pi^i_{\cdot|\mu}} \longleftarrow \underline{\alpha^i_{\cdot,\mu}}$$

Figure 62.2: Generalization of Fig.62.1. For a bnet with multiple nodes each of which may have multiple parents, this is a Markov chain of current counts ($\underline{N^i_{\cdot,\mu}}$), TPM ($\underline{\pi^i_{\cdot|\mu}}$), and prior counts ($\underline{\alpha^i_{\cdot,\mu}}$) .

$$P(N^i_{\cdot,\mu}|\pi^i_{\cdot|\mu}) = Mul(N^i_{\cdot,\mu};\pi^i_{\cdot|\mu}, N^i_{+,\mu}) \tag{62.42}$$

$$P(\pi^i_{\cdot|\mu}|\alpha^i_{\cdot,\mu}) = Dir(\pi^i_{\cdot|\mu};\alpha^i_{\cdot,\mu}) \tag{62.43}$$

In these TPMs,
$i \in S_{\underline{i}} = \{0, 1, \ldots, ni-1\}$= node index
$\underline{x}. = (\underline{x}_i)_{i\in S_{\underline{i}}}$= the nodes of the learned bnet.
$k \in S_{\underline{k}^i} = \{0, 1, \ldots, nk^i - 1\}$= states of node $\underline{x}_i$
$\mu \in S_{\underline{\mu}^i} = \{0, 1, \ldots, n\mu^i - 1\}$= states of parents of node $\underline{x}_i$.

In the previous section, we assumed a single node ($ni = 1$) with no parents ($n\mu^0 = 1$) so that we could drop the i, μ indices. In this section, we eliminate that restriction.

It is convenient to define the magnitude of a bnet B to equal the sum over nodes of the number of free parameters in each TPM:

$$|B| = \sum_i (nk^i - 1) n\mu^i \ . \tag{62.44}$$

Suppose that we are given $nsam$ samples $\vec{\underline{x}}_i = (\underline{x}_i[\sigma])_{\sigma=0,1,\ldots,nsam-1}$ of our learned bnet. The count numbers $N^i_{k,\mu}$ are defined in terms of those samples as follows:

$$N^i_{k,\mu} = \sum_\sigma \mathbb{1}(\underline{x}_i[\sigma] = k, pa(\underline{x}_i[\sigma]) = \mu) \ . \tag{62.45}$$

It is also convenient to defined count number ratios

$$N^i_{k|\mu} = \frac{N^i_{k,\mu}}{N^i_{+,\mu}} \ . \tag{62.46}$$

Note that $N^i_{k,\mu}$ is a positive integer whereas $N^i_{k|\mu} \in [0,1]$.

Let's denote the components of the TPMs by $\pi^i_{k|\mu}$:

$$\pi^i_{k|\mu} = P(\underline{x}_i = k \mid pa(\underline{x}_i) = \mu) \approx N^i_{k|\mu} \ . \tag{62.47}$$

The rest of this section lists equations that we obtained from the previous section, by adding the new indices i, μ:

$$\mathcal{K}(N^i_{\cdot,\mu}, \alpha^i_{\cdot,\mu}) = \frac{B(N^i_{\cdot,\mu} + \alpha^i_{\cdot,\mu})}{CI(N^i_{\cdot,\mu}) B(\alpha^i_{\cdot,\mu})} \tag{62.48}$$

$$\hat{\pi}^i_{k|\mu} = \frac{N^i_{k,\mu} + \alpha^i_{k,\mu}}{N^i_{+,\mu} + \alpha^i_{+,\mu}} \tag{62.49}$$

$$P(N^i_{\cdot,\mu} | \alpha^i_{\cdot,\mu}) = \mathcal{K}(N^i_{\cdot,\mu}, \alpha^i_{\cdot,\mu}) \tag{62.50}$$

$$P(N^i_{\cdot,\mu} | \pi^i_{\cdot|\mu}, \alpha^i_{\cdot,\mu}) \approx Mul(N^i_{\cdot,\mu} + \alpha^i_{\cdot,\mu}; \pi^i_{\cdot|\mu}, N^i_{+,\mu} + \alpha^i_{+,\mu}) \tag{62.51}$$

Claim 64

$$\ln P(N^i_{\cdot,\mu} | \hat{\pi}^i_{\cdot|\mu}, \alpha^i_{\cdot,\mu}) = \sum_k (N^i_{k,\mu} + \alpha^i_{k,\mu}) \ln \left(\frac{N^i_{k,\mu} + \alpha^i_{k,\mu}}{N^i_{+,\mu} + \alpha^i_{+,\mu}} \right) + \ln C(N^i_{\cdot,\mu} + \alpha^i_{\cdot,\mu}) \tag{62.52}$$

$$> \sum_k (N^i_{k,\mu} + \alpha^i_{k,\mu}) \ln \left(\frac{N^i_{k,\mu} + \alpha^i_{k,\mu}}{N^i_{+,\mu} + \alpha^i_{+,\mu}} \right) - \frac{1}{2}(nk^i - 1) \ln N^i_{+,\mu} \tag{62.53}$$

62.4 Bayesian Scores

- Bayesian Information Criterion (BIC)

$$\text{BIC-score} = -\sum_i \sum_{k,\mu} N^i_{k,\mu} \ln \left(\frac{N^i_{k,\mu}}{N^i_{+,\mu}} \right) + \underbrace{\left[-\frac{|B|}{2} \ln N^+_{+,+} \right]}_{\sum_i \sum_\mu \ln C(N^i_{\cdot,\mu}) \text{ would be more accurate}} \tag{62.54}$$

$$\approx \sum_i \sum_\mu \ln P(N^i_{\cdot,\mu} | \hat{\pi}^i_{\cdot|\mu}, \alpha^i_{\cdot,\mu} = 0) \tag{62.55}$$

- Bayesian Dirichlet (BD)

$$\text{BD-score} = \sum_i \sum_\mu \ln \frac{B(N^i_{\cdot,\mu} + \alpha^i_{\cdot,\mu})}{B(N^i_{\cdot,\mu})} \tag{62.56}$$

$$= \sum_i \sum_\mu \ln \left[CI(N^i_{\cdot,\mu}) P(N^i_{\cdot,\mu} | \alpha^i_{\cdot,\mu}) \right] \tag{62.57}$$

- BD equivalent (BDe)

$$\text{BDe-score} = \text{BD-score} \left(\alpha^i_{k,\mu} = \alpha' N^i_{k,\mu} \right) , \tag{62.58}$$

where α' is a free parameter.

- BD equivalent unified (BDeu)

$$\text{BDeu-score} = \text{BD-score} \left(\alpha^i_{k,\mu} = \frac{\alpha'}{nk^i n\mu^i} \right) , \tag{62.59}$$

where α' is a free parameter. The BDeu score satisfies **score equivalence**; i.e., it is the same for all DAGs in an equivalence class of observational equivalent DAGs. See Chapter 51 for more information about observational equivalence.

62.5 Information Theoretic scores

- Maximum likelihood

$$\text{ML-score} = \sum_i \sum_{k,\mu} N^i_{k,\mu} \ln N^i_{k|\mu} \tag{62.60}$$

$$= -\sum_i H(\underline{k}^i | \underline{\mu}^i) , \tag{62.61}$$

where $P_{\underline{k}^i|\underline{\mu}^i}(k|\mu) = N^i_{k|\mu}$ and $P_{\underline{k}^i,\underline{\mu}^i}(k,\mu) = N^i_{k,\mu}$.

- Bayesian Information Criterion (BIC), aka Minimum Description Length (MDL)

$$\text{BIC-score} = \text{ML-score} - \frac{|B|}{2} \ln N^{+}_{+,+} \tag{62.62}$$

$$\approx \sum_{i} \sum_{k,\mu} N^{i}_{k,\mu} \ln \frac{N^{i}_{k|\mu}}{\sqrt{N^{+}_{+,+}}} \tag{62.63}$$

- Akaike Information Criterion (AIC)

$$\text{AIC-score} = \text{ML-score} - |B| \tag{62.64}$$

$$\approx \sum_{i} \sum_{k,\mu} N^{i}_{k,\mu} \left[\ln N^{i}_{k|\mu} - 1 \right] \tag{62.65}$$

Chapter 63

Selection Bias Removal: COMING SOON

Ref.[1]

Chapter 64

Shapley Explainability

This chapter is based on Refs.[24] and [25], which I highly recommended.

"AI" is an ill-defined term. So is the term "explainability". So the term "Explainable AI (XAI)" is doubly ill-defined. In 2018, the European Union codified the need for XAI — because of an individuals "right to explanation" — into a law called the General Data Protection Right (GDPR). This EU law was a strong motivation for Neural Net and boosted decision tree practitioners to come up with a way to enhance their machine learning algorithms so that these comply with that law. Shapley explainability (SX) is one of the most popular methods for doing XAI.

So what does SX do? It ranks, for each individual of a population, the features (for example, race) of a dataset, in the order of how influential those features were in arriving at the decision the classifier made for that individual.

In my opinion, the goal of XAI is accomplished better with bnets than with SX enhanced NNs. SX "explains", a posteriori, the outcome of a model, whereas bnets reveal the a priori process whereby that outcome was reached. Thus, SX can tell you that a model is racist, but it can't suggest how to fix it. On the other hand, if a bnet is acting racist, you don't have to throw it away. It can be fixed. Another weakness of SX is that it is quite expensive computationally. Bnets have explainability built into them. For bnets, explainability is not an additional, posterior and quite onerous, calculation. That is why I like to call bnets the gold standard of XAI.

Let

F be the feature set. For example, $F = \{age, gender, job\}$.

$\mathcal{P}(A) = \{S : S \subset A\}$ be the power set of the set A (i.e., the set of all subsets of A, including the empty set $\emptyset$).

$|\mathcal{P}(A)| = \sum_{k=0}^{|A|} \binom{|A|}{k} = \sum_{k=0}^{|A|} \binom{|A|}{k} 1^k 1^{|A|-k} = (1+1)^{|A|} = 2^{|A|}$

$\mathcal{P}_f(F) = \{S \in \mathcal{P}(F) : f \in S\}$ for $f \in F$, be all sets in $\mathcal{P}(F)$ containing feature f. Note that $\mathcal{P}_f(F) = \mathcal{P}(F) - \mathcal{P}(F - \{f\})$

$\mathcal{P}_{!f}(F) = \{S \in \mathcal{P}(F) : f \notin S\}$ for $f \in F$, be all sets in $\mathcal{P}(F)$ not containing feature f. Note that $\mathcal{P}_{!f}(F) = \mathcal{P}(F - \{f\})$

Fig.64.1 shows a graph of $\mathcal{P}(F)$ for $F = \{age, gender, job\}$. Henceforth, we will refer to the generalization of Fig.64.1 to an arbitrary finite set F, as the **power**

Figure 64.1: Graph of $\mathcal{P}(F)$ for $F = \{age, gender, job\}$. An arrow $H \leftarrow T$, where $H \in \mathcal{P}(F)$ is the head of the arrow and $T \in \mathcal{P}(F)$ is the tail of the arrow, means $H \supset T$ and $|H| = |T| + 1$.

set graph of F.[1]

For any finite set F, consider its power set graph. Let $S \in \mathcal{P}(F)$ be a node of that graph. Let

$N_{arr/nd}(S) = |S|$ = number of arrows entering node S.

$N_{nds}(|S|) = \binom{|F|}{|S|}$ = number of nodes in level $|S|$ (i.e., row $|S|$).

$N_{arr}(S) = N_{arr/nd}(S) N_{nds}(|S|) = |S| \binom{|F|}{|S|}$ = number of arrows going from row $|S| - 1$ to row $|S|$.

$$P(S) = \frac{1}{N_{arr}(S)} = \frac{1}{|S| \binom{|F|}{|S|}} = \frac{1}{|F| \binom{|F|-1}{|S|-1}} \ . \tag{64.1}$$

Claim 65

$$\sum_{S \in \mathcal{P}_f(F)} P(S) = 1 \tag{64.2}$$

proof:

[1]Note that a power set graph is a DAG. We won't define TPMs for its nodes in this chapter, so it's a DAG but not a bnet, in this chapter at least.

$$\sum_{S\in\mathcal{P}_f(F)} P(S) = \sum_{S\in\mathcal{P}_f(F)} \frac{1}{|F|\binom{|F|-1}{|S|-1}} \tag{64.3}$$

$$= \sum_{k=1}^{|F|} \sum_{S\in\mathcal{P}_f(F)} \mathbb{1}(|S|=k) \frac{1}{|F|\binom{|F|-1}{k-1}} \tag{64.4}$$

$$= \sum_{k=1}^{|F|} \frac{1}{|F|\binom{|F|-1}{k-1}} \underbrace{\sum_{S\in\mathcal{P}_f(F)} \mathbb{1}(|S|=k)}_{\binom{|F|-1}{k-1}} \tag{64.5}$$

$$= 1 \tag{64.6}$$

QED

Consider any $S \in \mathcal{P}(F)$. Henceforth, we will represent a Machine Learning (ML) model ML_S as follows. ML_S can be a Linear Regression (LR_S) model, a Neural Net model (NN_S), or any other type of ML model. We will list a dataset; i.e., a set of tuples indexed by the individuals σ of a population Σ such that $|\Sigma| = nsam$. The independent variables of ML_S (i.e., $x_S^\sigma = [x_f^\sigma : f \in S]$) will be shown unboxed and the dependent variable (aka target feature) (i.e., y^σ) will be shown inside a box. Then we will show an arrow with the superscript "ML-fit", followed by the fit function obtained by performing ML_S.

$$ML_S: \quad \{(\sigma, x_S^\sigma, \boxed{y^\sigma}) : \sigma \in \Sigma\} \xrightarrow{\text{ML-fit}} \hat{y}(x_S) \tag{64.7}$$

For any feature $f \in F$ and individual $\sigma \in \Sigma$, define the **Shapley Value** (SHAP) by

$$SHAP_f^\sigma = \sum_{S\in\mathcal{P}_f(F)} P(S)\left[\hat{y}(x_S^\sigma) - \hat{y}(x_{S-\{f\}}^\sigma)\right] \tag{64.8}$$

$$= E_S[\hat{y}(x_S^\sigma) - \hat{y}(x_{S-\{f\}}^\sigma)] \tag{64.9}$$

Hence,

- $SHAP_f^\sigma$ is an average over the ensemble $\{ML_S : S \in \mathcal{P}_f(F)\}$ for each individual $\sigma \in \Sigma$.
- $SHAP_f^\sigma$ averages the change in output $\hat{y}$ when we change the model from one without feature f to one with feature f.
- $SHAP_f^\sigma$ can be negative or positive. Zero $SHAP_f^\sigma$ for individual σ means feature f does not influence how the decision $\hat{y}$ was arrived at for individual σ.

- An exact calculation of $SHAP_f^\sigma$, for all f and for a single σ, requires training a different ML model for each node of the power set graph of F, so it requires training $2^{|F|}$ models. Yikes! As $|F|$ grows, this quickly becomes unfeasible. For large $|F|$, one must resort to using sampling and approximations to get an approximation of the SHAP.

64.0.1 Numerical examples of SHAP

Next we present 2 numerical examples of SHAP. The figures and numerical values in this section were taken directly from Refs. [24] and [25].

1. **Predicting Income from $F = \{age, gender, job\}$.**

Figure 64.2: Same as Fig.64.1, but with added information for a specific individual σ. This figure contains enough information to evaluate $SHAP_{age}^\sigma$.

Consider the problem of predicting the income of a person based on the feature set $F = \{age, gender, job\}$. Suppose we are given a dataset for this problem. We can train models ML_S for each $S \in \mathcal{P}_{age}(F)$ where $\hat{y}$ is the income. Then we can calculate the matrix $SHAP_{age}^\sigma$. Fig.64.2 gives all the information necessary

to calculate $SHAP^{\sigma}_{age}$ for a single individual σ.

$$P(S) = \frac{1}{3\binom{2}{|S|-1}} = \begin{cases} \frac{1}{3\binom{2}{0}} = \frac{1}{3} & \text{if } |S| = 1 \\ \frac{1}{3\binom{2}{1}} = \frac{1}{6} & \text{if } |S| = 2 \\ \frac{1}{3\binom{2}{2}} = \frac{1}{3} & \text{if } |S| = 3 \end{cases} \tag{64.10}$$

$$\begin{aligned} SHAP^{\sigma}_{age} &= \underbrace{P(age)}_{1/3} \underbrace{[\hat{y}(x^{\sigma}_{age}) - \hat{y}(x^{\sigma}_{\emptyset})]}_{40K-50K} && (64.11) \\ &+ \underbrace{P(age, job)}_{1/6} \underbrace{[\hat{y}(x^{\sigma}_{age,job}) - \hat{y}(x^{\sigma}_{job})]}_{85K-100K} && (64.12) \\ &+ \underbrace{P(age, gender)}_{1/6} \underbrace{[\hat{y}(x^{\sigma}_{age,gender}) - \hat{y}(x^{\sigma}_{gender})]}_{39K-48K} && (64.13) \\ &+ \underbrace{P(age, gender, job)}_{1/3} \underbrace{[\hat{y}(x^{\sigma}_{age,gender,job}) - \hat{y}(x^{\sigma}_{gender,job})]}_{83K-95K} && (64.14) \\ &= -\$11.33K && (64.15) \end{aligned}$$

2. **Predicting passenger survival in the Titanic disaster.**

PassengerId	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
1	3	male	22.0	1	0	7.2500	S
2	1	female	38.0	1	0	71.2833	C
3	3	female	26.0	0	0	7.9250	S
4	1	female	35.0	1	0	53.1000	S
5	3	male	35.0	0	0	8.0500	S

Figure 64.3: First five rows (passengers) of an abridged version of the Titanic Dataset available at kaggle.com. This figure shows (σ, x^{σ}_{F}) for $\sigma = 1, 2, \ldots, 5$. It doesn't show the column $y^{\sigma} \in \{died, survived\}$.

Consider the problem of predicting whether an individual will survive or not based on a Titanic Dataset. We can train models ML_S for each $S \in \mathcal{P}(F)$ where $\hat{y} \in \{died, survived\}$. Then we can calculate the matrix $SHAP^{\sigma}_{f}$ for all individuals σ and features f. Fig.64.0.1 shows the first 5 rows of an abridged[2] version of the Titanic Dataset available at kaggle.com. Fig.64.4 displays $SHAP^{\sigma}_{f}$ in tabular form and Fig.64.5 displays $SHAP^{\sigma}_{f}$ in graphical form (in what is called a beeswarm plot).

[2]The Titanic Dataset available at kaggle.com has 891 rows and 15 columns, including columns for passenger ID and for $y^{\sigma} = survived? \in \{0, 1\}$. This abridged version has 8 columns.

PassengerId	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
1	-0.36	-0.76	0.11	0.05	-0.03	-0.3	-0.08
2	1.22	2.18	0.1	0.06	0.1	0.78	0.42
3	-0.75	1.62	0.07	0.1	-0.02	-0.01	-0.13
4	1.15	2.07	0.11	0.05	0.08	0.89	-0.13
5	-0.41	-0.73	-0.06	0.08	-0.03	-0.15	-0.08

Figure 64.4: For the Titanic dataset, this is a table of $SHAP_f^\sigma$, where $\sigma \in \{1, 2, \ldots 5\}$ and $f \in F$. Cells with positive SHAP are colored green, and those with negative SHAP are colored red. The colors are not an indication of whether the passenger died or survived. Note that the table of a dataset and the matrix $SHAP_f^\sigma$ have the same shape $(|\Sigma|, |F|)$.

Figure 64.5: For the Titanic Dataset, this is a so called "beeswarm" plot of $SHAP_f^\sigma$, where $\sigma \in \{1, 2, \ldots, 891\}$ and $f \in F$. In a beeswarm plot, the thickness of each row is proportional to how many individuals of the population have that value of the x coordinate. This plot comes from Ref.[24], where it was generated using the Titanic Dataset from kaggle.com and the wonderful Python library "SHAP". The SHAP library can plot Shapley Values in many other styles besides this one.

Chapter 65

Simpson's Paradox

This chapter is based on Chapter 6 of "The Book of Why", Ref.[41]. See also Ref.[120] and references therein.

Simpson's paradox is a recurring nightmare for all statisticians overseeing a clinical trial for a medicine. It is possible that if they leave out a certain "confounding" variable from a study, the study's conclusion on whether a medicine is effective or not, might be, without measuring that confounding variable, the opposite of what it would have been had that variable been measured.

Simpson's Paradox is greatly clarified by Judea Pearl's theory of causality. At the end of this chapter, we explain how.

Here is a simple example of Simpson's Paradox.

An equal number of patients of male and female genders are given a heart medicine or a placebo in a double blind study. Some subsequently have a heart attack. Let

$\underline{a}$ = heart attack? No=0, Yes=1
$\underline{t}$ = took medicine? No=0, Yes=1
$\underline{g}$ = gender? Female=0, Male=1

Figure 65.1: bnet for a simple example of Simpson's paradox. Here node $\underline{g}$ is a chain junction and a mediator.

This situation can be modeled by either bnet Fig.65.1. or bnet Fig.65.2. The two bnets are probabilistically equivalent (i.e., they both represent the same proba-

Figure 65.2: bnet that is probabilistically but not physically equivalent to bnet Fig.65.1. Here node $\underline{g}$ is a fork junction and a confounder.

bility distribution $P(a,t,g)$) because

$$P(g|t)P(t) = P(g,t) = P(t|g)P(g) \ . \tag{65.1}$$

For the bnet Fig.65.1, one has

$$P(a,g,t) = P(a|g,t)P(g|t)P(t) \ . \tag{65.2}$$

Therefore,

$$P(a=1|t) = \sum_g P(a=1|t,g)P(g|t) = E_{\underline{g}|t}P(a=1|t,\underline{g}) \ , \tag{65.3}$$

where $E_{\underline{g}|t}$ is a conditional expected value (a kind of weighted average).

Suppose q_0, q_1 are non-negative real numbers. For the vector $\vec{q} = (q_0, q_1)$:

Define a negative outcome (or failure or q_t increasing with t) if $q_0 \leq q_1$.

Define a positive outcome (or success or q_t decreasing with t) if $q_0 \geq q_1$.

Let

$$\vec{q}^{\,g} = [P(\underline{a}=1|t,g)]_{t=0,1} \tag{65.4}$$

for $g = 0, 1$, and

$$\vec{q}^{\,*} = [P(\underline{a}=1|t)]_{t=0,1} \ . \tag{65.5}$$

It is possible (see Fig.65.3 for a graphical explanation of how) to find perverse cases in which $P(a=1|t,g=0)$ and $P(a=1|t,g=1)$ increase with t but $P(a=1|t)$ decreases with t. So it is possible to conclude that the medicine is a failure for each of the two g populations considered separately, yet the medicine is a success when both populations are "amalgamated". The lesson is that a "trend reversal" is possible upon amalgamation. Trends are not necessarily preserved when we do a weighted average of type $E_{\underline{g}|t}$. $E_{\underline{g}|t}$ is an expected value on the random variable $\underline{g}$ conditioned on the root random variable $\underline{t}$.

So far, we have proven that probabilistically, the drug can be a failure for the populations of both sexes considered separately, but a success for the aggregate population.

Figure 65.3: $\vec{q}^{\,0}$, $\vec{q}^{\,1}$ vectors and bounding box for vector $\vec{q}^{\,*}$.

65.1 Pearl Causality

Pearl Causality would add the following two important insights to this problem:

1. bnets Fig.65.1 and Fig.65.2, although they are probabilistically equivalent, do not represent the same physical situation. In fact, only Fig.65.2 occurs in this case.

2. To decide whether the medicine is effective, we must apply a $do()$ operator to the $\underline{t}$ variable in Fig.65.2. The effect of that $do()$ operator is to erase the arrow going from $\underline{g}$ to $\underline{t}$. This in turn means that the average $E_{\underline{g}|t}$ in our equation for $P(a = 1|t)$ becomes a simpler average $E_{\underline{g}}$ which is independent of $\underline{t}$. But for such an average, the bounding box in Fig.65.3 degenerates to its diagonal line that connects the tips of the two vectors $\vec{q}^{\,0}$ and $\vec{q}^{\,1}$. The vector $\vec{q}^{\,*}$ must now fall on that diagonal line and must therefore also fall in the success region.

In conclusion, as Judea Pearl would say, if we ask the right question to Nature, i.e., what is $P[a = 1|do(\underline{t} = t)]$ for $t = 0, 1$, we get as an answer that the aggregate population preserves rather than reverses the unanimous trend of the two gendered populations.

65.2 Numerical Example

(a,t,g)	number of patients segregated by gender	number of patients of either gender
0,0,0	19	47
0,0,1	28	
0,1,0	37	49
0,1,1	12	
1,0,0	1	13
1,0,1	12	
1,1,0	3	11
1,1,1	8	

Table 65.1: Data for numerical example of Simpson's Paradox. This fictitious data was taken directly from Table 6.4, page 210 of "The Book of Why", Ref.[41].

$$P(a|t,g) = \begin{array}{c|cccc} & 0,0 & 0,1 & 1,0 & 1,1 \\ \hline 0 & 19/20 & 28/40 & 37/40 & 12/20 \\ 1 & 1/20 & 12/40 & 3/40 & 8/20 \end{array} \tag{65.6}$$

$$P(a|t) = \begin{array}{c|cc} & 0 & 1 \\ \hline 0 & 47/60 & 49/60 \\ 1 & 13/60 & 11/60 \end{array} \tag{65.7}$$

$$\begin{aligned} \frac{P(a=1,t=1,g=0)}{\sum_a P(a,t=1,g=0)} &= P(a=1|t=1,g=0) &= \frac{3}{40} \\ \frac{P(a=1,t=0,g=0)}{\sum_a P(a,t=0,g=0)} &= P(a=1|t=0,g=0) &= \frac{1}{20} = \frac{2}{40} \end{aligned} \tag{65.8}$$

$$\begin{aligned} \frac{P(a=1,t=1,g=1)}{\sum_a P(a,t=1,g=1)} &= P(a=1|t=1,g=1) &= \frac{8}{20} = \frac{16}{40} \\ \frac{P(a=1,t=0,g=1)}{\sum_a P(a,t=0,g=1)} &= P(a=1|t=0,g=1) &= \frac{12}{40} \end{aligned} \tag{65.9}$$

$$\begin{aligned} \frac{\sum_g P(a=1,t=1,g)}{\sum_g \sum_a P(a,t=1,g)} &= P(a=1|t=1) &= \frac{11}{60} \\ \frac{\sum_g P(a=1,t=0,g)}{\sum_g \sum_a P(a,t=0,g)} &= P(a=1|t=0) &= \frac{13}{60} \end{aligned} \tag{65.10}$$

Note that the right hand side of Eq.(65.8) is higher for $t = 1$ than for $t = 0$. Same trend occurs in Eqs.65.9 but is reversed in Eqs.65.10.

Chapter 66

Structure and Parameter Learning for Bnets

Learning a bnet from data is a computationally intensive NP-complete problem. Therefore, the best one can hope for is for heuristic algorithms that solve this problem approximately. A huge number of such algorithms have been tried and continue to be tried. Luckily, there exists a free open source software library called `bnlearn` that covers many of them. The goal of this chapter is to give a brief overview of the subject of bnet learning, after which we recommend to those readers who want to pursue this subject further, to learn `bnlearn` .

This chapter is based on the `bnlearn` website Ref.[46], and on a 2019 survey paper [47] by Scutari et al. I highly recommend looking at both. Refs. [3] and [22] were also helpful to me in understanding this subject.

`bnlearn` (Ref.[46]) (free, open source) is very comprehensive and well maintained. It is written mostly in C with an R front-end. It was developed by Marco Scutari and collaborators over a time period of more than 10 years, and is still under active development. How things stand in the field of bnet learning software reminds me of how things stand in the field of linear algebra (LA) software. Perfecting and optimizing LA software takes many years so I would not advise you to write your own LA software library starting from scratch. There is no need to do so. Instead, you can use LAPACK (free, open source), which has been perfected and expanded for decades by world experts. I view `bnlearn` as the LAPACK of bnet learning.

66.1 Overview

To give the reader an overview of the subject and of `bnlearn` itself, here is a highly simplified tree, compiled from the `bnlearn` website and documentation, of some of the subjects covered by `bnlearn` .

```
│ Parameter Learning
│  └─ missing data
└─ Structure Learning
```

```
│   ├── tree-like structures given a priori
│   │   ├── Naive Bayes
│   │   ├── Chow-Liu tree
│   │   ├── Tree Augmented Naive Bayes (TAN)
│   │   └── ARACNE
│   ├── score based
│   │   ├── algorithms
│   │   │   ├── hill climbing (HC)
│   │   │   ├── HC with random restarts
│   │   │   ├── HC with Tabu list (Tabu)
│   │   │   ├── simulated annealing
│   │   │   └── genetic algorithms
│   │   └── scoring functions
│   │       ├── Information Theoretic scores
│   │       ├── Bayesian Information Criterion (BIC)
│   │       └── Bayesian Dirichlet (BD) family
│   ├── constraint based
│   │   ├── algorithms
│   │   │   ├── PC family
│   │   │   ├── Grow-Shrink (GS)
│   │   │   └── Incremental Association Markov Blanket (IAMB) family
│   │   └── conditional independence tests
│   │       ├── mutual information (parametric, semiparametric and permutation
│   │       │   tests)
│   │       └── shrinkage-estimator for the mutual information
│   └── hybrid
│       ├── Max-Min Hill Climbing (MMHC)
│       ├── Hybrid HPC (H2PC)
│       └── General 2-Phase Restricted Maximization (RSMAX2)
├── parallel mode structure learning
├── node types
│   ├── all-discrete
│   ├── all-continuous
│   └── mixed
└── utility functions
    ├── model comparison and manipulation
    ├── random data generation
    ├── arc orientation testing
    ├── simple and advanced plots
    ├── parameter estimation (maximum likelihood and Bayesian)
    ├── inference, conditional probability queries
    └── cross-validation
```

```
└─ bootstrap
└─ model averaging
```

Let

- PL=parameters learning (i.e, learning the TPMs)
- SL= structure learning (i.e., learning the DAG)
- ML= model (or bnet) learning, SL+PL

PL is easy, once the structure is known. PL assuming no missing data goes as follows. Using the notation of Chapter 62, define

$$\pi^i_{k|\mu} = P(\underline{x}_i = k \mid pa(\underline{x}_i) = \mu) \ . \tag{66.1}$$

Then $\pi^i_{k|\mu}$ can be estimated from the data $N^i_{k,\mu}$ using:

$$\pi^i_{k|\mu} \approx N^i_{k|\mu} = \frac{N^i_{k,\mu}}{N^i_{+,\mu}} \ . \tag{66.2}$$

PL described by Eq.(66.2) is only for discrete nodes with no missing data. `bnlearn` can also do PL with missing data and continuous (Gaussian linear only) nodes. See Chapter 44 on missing data and Chapter 24 on Gaussian linear nodes. SL actually does PL and SL at the same time.

There are 3 main types of SL: score based, constraint based, and hybrid. `bnlearn` can perform many algorithms of each of these 3 types of SL. It can perform most of them with either all-discrete, or all-continuous or mixed nodes. It can perform many of them in parallel mode. The 2019 survey paper Ref.[47] by Scutari et al compares the performance of many different bnet learning algorithms.

66.2 Score based SL algorithms

Score based SL algorithms require scoring bnets (with either all-discrete, all-continuous or mixed nodes). See Chapter 62 for an introduction to scoring bnets. The BIC score explained in that chapter is very popular and works for all-discrete, all-continuous or mixed nodes.

Score-based SL algorithms apply standard optimisation techniques. In the Hill Climbing algorithm, the current best bnet is changed slightly and then given a score that measures how well it fits the data. The bnet with the highest (=best) score so far, as well as that highest score, are stored. (Hence, this is called a greedy search). The process continues until the latest highest score stops changing. The problem with being greedy all the time is that the answer might converge to a local maximum. To mitigate this problem and allow some probability of visiting more than one local maximum, one uses a Tabu Table, random restarts, simulated annealing, genetic algorithms, etc.

66.3 Constraint based SL algorithms

To fully understand constraint based SL algorithms, the reader is advised to read Chapters 17 and 51 first.

Constraint based SL algorithms require estimating from the data the conditional independence $\underline{x}. \perp_P \underline{y}.|\underline{a}.$ for any 3 disjoint multinodes $\underline{x}., \underline{y}., \underline{a}.$. This can be done by estimating the conditional mutual information (CMI) $H(\underline{x}. : \underline{y}.|\underline{a}.)$. `bnlearn` can calculate CMI and other metrics of $\underline{x}. \perp_P \underline{y}.|\underline{a}.$. All these metrics are very similar; they all measure how close $P(x.|y., a.)$ and $P(x.|a.)$ are.

The first constraint-based SL algorithm was the Inductive Causation (IC) algorithm proposed by Pearl and Verma in 1991. Incremental improvements have been proposed since then, such as the PC family of algorithms, Grow-Shrink and the Incremental Association Markov Blanket (IAMB) family of algorithms.

66.4 Pseudo-code for some bnet learning algorithms

Algorithm 2: Pseudo-code for Hill Climbing algorithm

```
Input  : Data D, Vertices V
Output: a bnet B = (G, T), where G = (V, E) is a DAG, where V are its
        vertices (nodes) and E are its edges (arrows). T are all its
        Transition Probability Matrices (TPMs) T = TPMs(G, D).
E ← ∅
T ← ∅
B ← (V, E, T)
maxscore ← −∞
// DE= all possible directed edges
DE = {x → y ∈ V × V : x ≠ y}
again ← True
while again do
    for all x → y ∈ DE do
        // add arrow
        E+ ← E ∪ {x → y}
        // delete arrow
        E− ← E − {x → y}
        // reverse arrow
        ER ← E− ∪ {y → x}
        for E' = E+, E−, ER do
            if E' ≠ E and G' = (V, E') is a legal DAG then
                T' ← TPMs(G', D)
                B' ← (G', T')
                newscore = BIC-score(B')
                if newscore > maxscore then
                    B ← B'
                    maxscore ← newscore
                else
                    again ← False
return B
```

Algorithm 3: Pseudo-code for PC-Stable algorithm

Input : Data D, Vertices (nodes) V, tolerance in CMI $\epsilon > 0$

Output: partially oriented acyclic graph $G = (V, E, UE)$, where V are the vertices (nodes), E are the oriented edges (arrows) and UE are the unoriented edges.

```
E ← ∅
// initialize UE to fully-connected undirected graph
UE ← {x − y ∈ V × V : x − y = y − x, x ≠ y}
// Shrink phase.  Deletes edges from E.
for λ = 0, 1, 2, ..., |V| − 2 do
    for all x − y ∈ UE do
        for all S = {a ∈ V : x − a ∈ UE, a ≠ x, y} ∋ |S| = λ do
            if H(x : y|S) < ε then
                /* If there were an arrow between x and y, then
                   conditioning on S would not be enough to interrupt
                   info transmission H(x : y|S) between x and y        */
                UE ← UE − {x − y}
                S(x − y) ← S
// Growth phase.  Adds v structures to E.
for all x, y, a such that x − a ∈ UE, a − y ∈ UE, x − y ∉ UE, a ∉ S(x − y) do
    /* If there were no collider at a, then there would be info
       transmission between x and y                                    */
    UE ← UE − {x − a, a − y}
    E ← E ∪ {x → a, y → a}
// Orienting edges.
again ← True
size ← |UE|
while again do
    for all x − y ∈ UE do
        if x → y ∈ E, y − z ∈ UE, x − z ∉ UE, ∄w ∋ w → y ∈ E then
            // to avoid introducing new v structure
            UE ← UE − {y − z}
            E ← E ∪ {y → z}
        if x → y ∈ E and there is directed path from x to y in E then
            // to avoid introducing cycles
            UE ← UE − {x − y}
            E ← E ∪ {x → y}
    newsize ← |UE|
    if size == newsize then
        again ← False
    else
        size ← newsize
return G = (V, E, UE)
```

Chapter 67

Support Vector Machines And Kernel Method

This chapter is based on Refs.[97]. [123] and [98].

The Support Vector Machines (SVM) method was first invented with a linear kernel, but was later generalized to arbitrary kernels. We will use the terms SVM method and Kernel Method indistinguishably.

The SVM method is a fairly general method for calculating, via supervised learning, a *binary* classifier. The SVM method finds a continuous surface that separates a space into two disjoint parts.

Let $\Sigma = [0, 1, 2, \ldots, nsam-1]$ be a list of individuals (samples) in a population. In this chapter, we will use the notation $A^\sigma = A[\sigma]$ and $\vec{A} = [A^\sigma : \sigma \in \Sigma]$ for a list (vector, 1-D array) indexed by Σ. We will refer to $DS = (\vec{x}, \vec{y})$ where $x^\sigma \in S_{\underline{x}}$, $y^\sigma \in \{-1, 1\}$, as a dataset. Let $x^\sigma = (x_0^\sigma, x_1^\sigma, \ldots, x_{nf-1}^\sigma) \in S_{\underline{x}_0} \times S_{\underline{x}_1} \times \ldots \times S_{\underline{x}_{nf-1}} = S_{\underline{x}}$. When $x_j^\sigma \in \mathbb{R}$ for all j, we will take $x^\sigma \in \mathbb{R}^{nf}$ to be a column vector. x^σ is the feature vector for individual σ, and its components x_i^σ for $i = 0, 1, \ldots, nf-1$ are the features. $y^\sigma \in \{-1, 1\}$ is the binary class to which x^σ belongs.

Let $\hat{y}(x^{\sigma_0}) \in \{-1, 1\}$ be an estimate of $y^{\sigma_0} \in \{-1, 1\}$. The **SVM classifier** is defined as

$$\hat{y}(x^{\sigma_0}) = \text{sign}(Y(x^{\sigma_0})) \tag{67.1}$$

where[1]

$$Y(x^{\sigma_0}) = \sum_\sigma \alpha^\sigma y^\sigma K(x^\sigma, x^{\sigma_0}) \ . \tag{67.2}$$

The **binary weight coefficients** $\alpha^\sigma \in \{0, 1\}$ for all $\sigma \in \Sigma$ are found by training, via an algorithm to be described below.

The function $K : S_{\underline{x}} \times S_{\underline{x}} \to \mathbb{R}$ is called the **Kernel or Similarity function**. We assume that $K(x^\sigma, x^{\sigma_0})$ grows bigger when its two arguments x^σ and x^{σ_0} become

[1] Define sign(0) = 1.

more "similar". We also assume that $K(x^\sigma, x^{\sigma_0})$ is symmetric in its two arguments.

67.1 Learning Algorithm for SVM Classifier

Figure 67.1: Time slice σ_0 of dynamic bnet for learning binary weights $\vec{\alpha}$ of SVM classifier.

Given a kernel function K and a dataset $(\vec{x}, \vec{y})$, the SVM classifier is fully specified except for its binary weights $\vec{\alpha}$. Those weights can be learned via the algorithm represented as a causal diagram in Fig.67.1. That figure shows one time slice of a dynamic bnet. The TPMs, printed in blue, of the nodes of the bnet Fig.67.1, are as follows:

$$P(\hat{y}|\vec{\alpha}, (\vec{x}, \vec{y}), x^{\sigma_0}) = \mathbb{1}(\quad \hat{y} = \text{ given by Eq.(67.1).} \quad) \tag{67.3}$$

$$P(\mathcal{E}|\hat{y}, y^{\sigma_0}) = \mathbb{1}(\quad \mathcal{E} = \mathbb{1}(\hat{y} \neq y^{\sigma_0}) \quad) \tag{67.4}$$

The first (but not the second, third , etc.) $\vec{\alpha}$ node of Fig.67.1 is a root node. The TPM for that root node should set all components of $\vec{\alpha}$ to zero:

$$P(\vec{\alpha}) = \prod_\sigma \mathbb{1}(\quad \alpha^\sigma = 0 \quad) \,. \tag{67.5}$$

After that initialization,

$$P(\vec{\alpha'}|\vec{\alpha}, \mathcal{E}) = \mathbb{1}(\quad (\alpha')^{\sigma_0} = \alpha^{\sigma_0} + \mathcal{E} \quad) \prod_{\sigma \neq \sigma_0} \mathbb{1}(\quad (\alpha')^\sigma = \alpha^\sigma \quad) \tag{67.6}$$

Why this learning algorithm works.

Figure 67.2: Define the neighborhood of x^{σ_0} by $\mathcal{N}(x^{\sigma_0}) = \{x^\sigma : |K(x^\sigma, x^{\sigma_0})| < \epsilon\}$ for some $\epsilon > 0$.

$K(x^\sigma, x^{\sigma_0})$ sets to zero any contribution to $Y(x^{\sigma_0})$ from points x^σ outside the neighborhood $\mathcal{N}(x^{\sigma_0})$ of x^{σ_0}. (See Fig.67.2). If $\hat{y}(x^{\sigma_0}) = y^{\sigma_0}$, keep $\alpha^{\sigma_0} = 0$ because the neighbors of x^{σ_0} are giving the correct $\hat{y}(x^{\sigma_0})$ when they are polled and the majority wins. If, on the other hand, $\hat{y}(x^{\sigma_0}) \neq y^{\sigma_0}$, then switch α^{σ_0} from 0 to 1, which means x^{σ_0} gets to vote by adding y^{σ_0} to $Y(x^{\sigma_0})$. So we start off with all $\alpha^\sigma = 0$ and we end with most of them still zero except for a select few. If we were to set all α^σ equal to one, we would get overfitting and a very jagged separation between the two classes. The fact that we end with only a select few α^σ equal to 1, and the rest equal to 0, helps make the demarcation between the two classes less jagged.

67.2 Linear (dot-product) Kernel

$$Y(x^\sigma) = w^T x^\sigma + b$$

Figure 67.3: Graph of line $Y(x^\sigma) = 0$ splits plane into regions with $Y < 0$, $Y = 0$ and $Y > 0$.

So far, we have discussed the SVM method for an arbitrary kernel. This section is devoted to the **Linear (aka dot-product) Kernel**. Said kernel is defined as

$$K(x^\sigma, x^{\sigma_0}) = (x^\sigma)^T x^{\sigma_0} \ . \tag{67.7}$$

For this kernel, Eq.(67.2) specializes to

$$Y(x^{\sigma_0}) = \sum_{\sigma} \alpha^{\sigma} y^{\sigma} K(x^{\sigma}, x^{\sigma_0}) + b \tag{67.8}$$

$$= w^T x^{\sigma_0} + b \tag{67.9}$$

where

$$w = \sum_{\sigma} \alpha^{\sigma} y^{\sigma} x^{\sigma} \ . \tag{67.10}$$

Figure 67.4: We refer to the gray shaded region with $-1 < Y < 1$, where $Y = w^T x + b$, as the DMZ.

We started this chapter by pulling the SVM classifier out of a hat. We did give reasons why it works, but we did not derive it from a more general minimization principle. Such a derivation is possible, at least in the linear kernel case, and we give it next.

Consider the following 3 straight lines:

$$w^T x^{\sigma} + b = +A \tag{67.11}$$

$$w^T x^{\sigma} + b = 0 \tag{67.12}$$

$$w^T x^{\sigma} + b = -A \tag{67.13}$$

where $w, x^{\sigma} \in \mathbb{R}^{nf}$, and $b, A \in \mathbb{R}$. We can re-scale the vector w and scalar b so as to get rid of the A. (i.e., replace $w \to wA$ and $b \to bA$ and divide common factor A out of equations). This rescaling does not affect the graphs (i.e., x loci) of these 3 lines. Now we have:

$$w^T x^\sigma + b = +1 \tag{67.14}$$
$$w^T x^\sigma + b = 0 \tag{67.15}$$
$$w^T x^\sigma + b = -1 \tag{67.16}$$

If Y stands for

$$Y = w^T x^\sigma + b \ , \tag{67.17}$$

then we define the **DMZ (demilitarized zone)** to be the region

$$DMZ = \{x^\sigma : |Y(x^\sigma)| < 1\} \ . \tag{67.18}$$

The lines $Y = \pm 1$ will be called the **borders (aka margins)** of the DMZ, and line $Y = 0$ will be called the **line of demarcation** of the DMZ. The DMZ is illustrated in Fig.67.4.

Let D_{DMZ} be the **DMZ width** (i.e., the distance from one border of the DMZ to the other.) Position vectors pointing from the origin to either of the two DMZ borders are called **support vectors**. Suppose $X^+, X^- \in \mathbb{R}^{nf}$ are two support vectors on opposite DMZ borders with $|X^+ - X^-| = D_{DMZ}$. Then

$$w^T X^+ + b = 1 \tag{67.19}$$
$$w^T X^- + b = -1 \tag{67.20}$$

so

$$D_{DMZ} = \frac{2}{|w|} \ . \tag{67.21}$$

For any $a \in \mathbb{R}$, let the **positive** a_+ **and negative** a_- **parts of** a be given by

$$a = \underbrace{a_+}_{a\mathbb{1}(a>0)} + \underbrace{a_-}_{a\mathbb{1}(a\leq 0)} \ . \tag{67.22}$$

When $Y = \pm 1$, an error in $Y(x^\sigma)$ occurs iff $y^\sigma Y(x^\sigma) = -1$. But how should we define errors when Y is a real number? Define the **Cost of erring for sample** σ to be

$$CE^\sigma(x^\sigma, y^\sigma) \ = \ (1 - y^\sigma Y(x^\sigma))_+ \ . \tag{67.23}$$

CE^σ is shown in Fig.67.5. As you can see, there is a penalty for living on the incorrect side, and even a penalty for living on the correct side but too close to the DMZ.

Figure 67.5: Plot of CE^σ versus Y.

Note that the line of demarcation should have the lowest CE^σ for all possible w, b. So to find that line, we want to minimize CE^σ with respect to w, b. But note that $CE^\sigma \geq 0$, and it can be zero for an appropriately chosen w. So we need to add another cost in order to get a non-zero total cost. Define the **DMZ cost** as

$$CZ = \frac{1}{2}|w|^2 = \frac{2}{D_{DMZ}^2} \tag{67.24}$$

Note that $CZ \to \infty$ as $D_{DMZ} \to 0$, so CZ penalizes DMZ's that are too narrow.

Now define a Lagrangian $\mathcal{L}$ to be the sum of these 2 contributions.

$$\mathcal{L} = CZ + \sum_\sigma CE^\sigma \tag{67.25}$$

$$= \frac{1}{2}|w|^2 + \sum_\sigma (1 - y^\sigma Y(x^\sigma))_+ \tag{67.26}$$

This particular choice of CZ is not unique, but it isn't totally arbitrary either. We want it to be independent of the sample σ, and to depend on a geometrical aspect of the DMZ, like its width $D_{DMZ} = 2/|w|$. Note that CE^σ behaves, when $|w| \to \infty$, linearly in $|w|$. We are going to differentiate $\mathcal{L}$ with respect to $|w|$ to find an optimum. But straight lines have no optima, so we need CZ to behave, when $|w| \to \infty$, as $|w|^p$ for some integer $p > 1$.

Setting the variation $\delta\mathcal{L}$ to zero, we get

$$0 = \delta\mathcal{L} = \delta w_i \left\{ w_i - \sum_\sigma \mathbb{1}(y^\sigma Y(x^\sigma) < 1) y^\sigma x_i^\sigma \right\} \tag{67.27}$$

so

$$w = \sum_\sigma \underbrace{\mathbb{1}(y^\sigma Y(x^\sigma) < 1)}_{\alpha^\sigma} y^\sigma x^\sigma \ . \tag{67.28}$$

67.3 Alternatives to Linear Kernel

Sometimes it is convenient to replace the dot-product kernel given above by a different kernel. Other popular kernels are:

- **Radial Basis Function (RBF) Kernel.** In this case, K is a radial function; i.e., a function that depends only on the magnitude (radius, Euclidean distance, L^2 norm) of a vector. An example of an RBF kernel is the **Gaussian Kernel**

$$K(x^\sigma, x^{\sigma_0}) = e^{-\gamma|x^\sigma - x^{\sigma_0}|^2} \tag{67.29}$$

  for some free parameter $\gamma > 0$.

- **Inhomogeneous Polynomial Kernel**

$$K(x^\sigma, x^{\sigma_0}) = [(x^\sigma)^T x^{\sigma_0} + 1]^d \tag{67.30}$$

  for some positive integer d.

- **Homogeneous Polynomial Kernel**

$$K(x^\sigma, x^{\sigma_0}) = [(x^\sigma)^T x^{\sigma_0}]^d \tag{67.31}$$

  for some positive integer d.

- **"Kernel trick" Kernel.** Consider a map $\Phi : \mathbb{R}^{nf} \to \mathbb{R}^N$. Usually $N > nf$. Φ can be a rectangular matrix $A \in \mathbb{R}^{N \times nf}$ so that $\Phi(x^\sigma) = Ax^\sigma \in \mathbb{R}^N$. Let

$$K(x^\sigma, x^{\sigma_0}) = [\Phi(x^\sigma)]^T \Phi(x^{\sigma_0}) \tag{67.32}$$

  Although the constant surfaces of this kernel are hyperplanes in $\mathbb{R}^N$, its constant surfaces in $\mathbb{R}^{nf}$ can be curved and even closed compact surfaces (e.g. spheres).

67.4 Random Forest and Kernel Method

Chapter 68

Synthetic Controls

This chapter is based on Refs.[8] and [6].

This chapter assumes that the reader has read Chapter 13 on the Difference-in-Differences (DID) method.

The Synthetic Controls (SC) method is a simple enhancement of the DID method. SC enhances DID in two simple yet powerful ways:

1. **Better time resolution.** DID considers just 2 time-snapshots (i.e., a time-series with only 2 times) whereas SC considers arbitrarily many time-snapshots (i.e., a time-series with more than 2 times).

2. **Weighted average of controls.** DID divides the population of individuals into just 2 kinds: the treated and the untreated (aka controls). SC divides the total population into treated and controls just like DID does, but it goes further and divides the control population into multiple subpopulations, and calculates a weighted average, called a "synthetic control", of those subpopulations. The weights of the synthetic control are chosen so that it mimics as closely as possible the behavior of the treated population for all times measured before the treatment was applied.

Let us describe these two enhancements more precisely.

- **timing**: Let t_k for $k = 0, 1, \ldots, npre - 1$ be the pre-treatment times at which a measurement occurs. Let t_k for $k = npre, npre + 1, \ldots, nt - 1$ be the post-treatment times at which a measurement occurs. Note that $npre + npost = nt$. Note that $t_* = t_{npre+1}$ is the first measurement time after the treatment is applied, t_0 is the first measurement time, and $t_{fin} = t_{nt-1}$ is the last one.

- **subpopulations**: Let $S_1 = \{\sigma_1\}$ be the set of treated units (just one). Let $S_0 = \{\sigma : \sigma \neq \sigma_1\}$ be the set of untreated units (i.e., controls). Let $nsam =$ number of all units σ, $n_1 = |S_1| = 1$, and $n_0 = |S_0| = nsam - 1$.

- **weights**:

Figure 68.1: Pictorial representation of the Synthetic Controls (SC) method. The outcome y of the synthetic control unit is colored red and that of the treated unit is colored blue. They roughly agree for $t < t_*$.

We want to define a time-independent weight w^σ for each unit σ in such a way that the output y_t^σ for the synthetic control unit behaves like the output for the treated unit σ_1 for $t < t_*$.

Let

$$w^{\sigma_1} = 0 \tag{68.1}$$

and

$$w^{n_0} = \{w^\sigma\}_{\sigma \neq \sigma_1} \ . \tag{68.2}$$

Define a cost function $\mathcal{C}$:

$$\mathcal{C}(w^{n_0}) = \sum_{t<t_*} \left(y_t^{\sigma_1} - \sum_{\sigma \neq \sigma_1} w^\sigma y_t^\sigma \right)^2 \tag{68.3}$$

Then calculate w^{n_0} by minimizing the cost function, subject to the constraint that w^{n_0} be a probability distribution:

$$w^{n_0} = \underset{W^{n_0}}{\operatorname{argmin}} \left\{ \mathcal{C}(W^{n_0}) : W^\sigma \geq 0, \sum_{\sigma \neq \sigma_1} W^\sigma = 1 \right\} \ . \tag{68.4}$$

Now that we have defined a weight w^σ for every unit σ, we can define for $c \in \{0, 1\}$,

$$y_t^{\sigma_1}(c) = \begin{cases} y_t^{\sigma_1} & \text{if } c = 1 \\ \sum_{\sigma \neq \sigma_1} w^\sigma y_t^\sigma & \text{if } c = 0 \end{cases} \tag{68.5}$$

$$\mathcal{Y}_c(t) = E_\sigma[y_t^\sigma(c)] \tag{68.6}$$

and

$$\delta_t = \mathcal{Y}_1(t) - \mathcal{Y}_0(t) \tag{68.7}$$

δ_t is illustrated in Fig.68.1. It approximates $ATE(t)$.

68.1 PO analysis

In this section, we show how to analyze the SC method using the formalism of PO theory.

Figure 68.2: $t \in \{t_0, t_1, \ldots, t_{fin}\}$. Bnet $G_{t,+}$ is obtained by adding two new nodes $\underline{y}_t(0)$ and $\underline{y}_t(1)$ to bnet G_t.

As usual for PO theory, we will consider expected values of y_t^σ:

$$E_{\sigma|d,x}[y_t^\sigma(c)] = E_{\underline{y}_t(c)|d,x}[\underline{y}_t(c)] = \mathcal{Y}_{c|d,x}(t) \tag{68.8}$$

To calculate these expected values, we need a "model" with probability distributions. In this case, the needed model and probability distributions are provided by the bnets depicted in Fig.68.2. The TPMs, printed in blue, for the bnet $G_{t,+}$ in Fig.68.2, are as follows. Note that the TPMs for the bnet $G_{t,+}$ are defined in terms of the TPMs for the bnet G_t.

$$P(x) = P_{\underline{x}}(x) \tag{68.9}$$

$$P(d|x) = P_{\underline{d}|\underline{x}}(d|x) \tag{68.10}$$

$$P(y_t|y_t(0), y_t(1), d) = \mathbb{1}(y_t = y_t(d)) \tag{68.11}$$

$$P(y_t(c)|x) = P(y_t(c)|d, x) = \text{given} \tag{68.12}$$

Figure 68.3: Four different time-dependent expected values $\mathcal{Y}_{c|d}(t)$ of y_t^σ for bnet $G_{t,+}$ The $2 * nt$ magenta stars represents the $2 * nt$ SC measurements.

Fig.68.3 depicts the four functions $\mathcal{Y}_{c|d}(t)$ for t in the interval $[t_0, t_{fin}]$ and for $c, d \in \{0, 1\}$. The $\mathcal{Y}$ coordinates of the $2 * nt$ magenta stars in Fig.68.3 can be calculated using bnet G_t. Note that in Fig.68.3, we display a large gap between the curves $\mathcal{Y}_{0|d}(t)$ for $d \in \{0, 1\}$. In reality, $P(y_t(0)|d)$ has been constructed so as to make that gap as small as possible. Thus, to a good(?) approximation,

$$\delta_t \approx ATE_t \tag{68.13}$$

Unlike in the DID method, in the SC method, to a good(?) approximation, we don't have to worry about parallel trends.

Chapter 69

Time Series Analysis: ARMA and VAR

This chapter is based mostly on the book Ref.[14] on time series analysis by Hamilton, and on the lectures Ref. [19] by Chung-Ming Kuan. In writing this chapter, we also profited greatly from numerous Wikipedia entries on time series analysis, such as the entries on time series (Ref.[124]), ARMA time series (Ref.[62]), AR time series (Ref.[61]), MA time series (Ref.[107]), and VAR time series (Ref.[129]).

We cover only a small fraction of the treasures covered in those sources, and only cover stationary time-series. Non-stationary time series we don't even touch. But we hope to have covered enough to pique our readers's interest in time series analysis, and make him/her appreciate how bnets make time series much more intuitive and fun. The time-series considered in this chapter can be represented by one of the simplest types of bnets, namely, the LDEN bnets introduced in Chapter 36.

As usual, for $t, t_a, t_b \in \mathbb{Z}$, let $\mathbb{Z}_{<t} = \{t-1, t-2, t-3, \ldots\}$, $\mathbb{Z}_{[t_a,t_b]} = \{t_a, t_a + 1, \ldots, t_b\}$, etc.

Let $\underline{x}_t \in \mathbb{R}$. A **time series (t-series)**, denoted variously by $\{\underline{x}_1, \underline{x}_2, \ldots, \underline{x}_{nt}\} = \{\underline{x}_t\}_{t=1}^{nt} = \{\underline{x}_t\}_{\forall t}$, is a set of real numbers index by a discrete set of times $\mathbb{Z}_{[0,nt]}$.

For $t_a < t_b$, let $\underline{x}_{[t_a,t_b]} = (\underline{x}_{t_a}, \underline{x}_{t_a+1}, \ldots, \underline{x}_{t_b})$. Let $x_{<t} = (\ldots, x_{t-2}, \underline{x}_{t-1})$.

69.1 White noise

By **white noise** $\{\underline{n}_t\}_{\forall t} \sim WN(0, \sigma^2)$ we mean a t-series $\{\underline{n}_t\}_{\forall t}$ that satisfies

$$E[\underline{n}_t] = 0 \tag{69.1}$$

and

$$\langle \underline{n}_t, \underline{n}_{t'} \rangle = \sigma^2 \delta(t, t') \ . \tag{69.2}$$

Gaussian white noise $\{\underline{n}_t\}_{\forall t} \sim WN(0, \sigma^2)_{\mathcal{N}}$ is white noise such that also $\underline{n}_t \sim \mathcal{N}(0, \sigma^2)$.

69.2 Backshift operator

$\mathcal{B}$ is the backshift (a.k.a. lag) operator. For any t-series $\{x_t\}_{\forall t}$, $\{y_t\}_{\forall t}$ and scalars $a, b \in \mathbb{R}$,

$$\mathcal{B}\underline{x}_t = \underline{x}_{t-1} \tag{69.3}$$

$$\mathcal{B}(a\underline{x}_t + b\underline{y}_t) = a\mathcal{B}(\underline{x}_t) + b\mathcal{B}(\underline{y}_t) \tag{69.4}$$

A $\mathcal{B}$ Polynomial with coefficients $\alpha_{[0,p]}$:

$$\alpha(\mathcal{B}) = \alpha_0 + \alpha_1\mathcal{B} + \alpha_2\mathcal{B}^2 + \ldots \alpha_p\mathcal{B}^p \tag{69.5}$$

$\mathcal{B}^{-1}$ is the inverse of the backshift operator (a.k.a. frontshift operator)

$$\mathcal{B}^{-1}\underline{x}_t = \underline{x}_{t+1} \tag{69.6}$$

The following two Taylor expansions prove useful in finding the inverse of backshift operator polynomials:

- $$\frac{1}{1-z} = 1 + z + z^2 + \ldots \tag{69.7}$$

  converges for $z \in \mathbb{C}$ with $|z| < 1$. We will use this expansion with z replaced by $\alpha\mathcal{B}$, where $\alpha \in \mathbb{R}$.

- $$\frac{1}{1-z} = (-z^{-1})\left[\frac{1}{1-z^{-1}}\right] = (-z^{-1})\left[1 + z^{-1} + +z^{-2} + \ldots\right] \tag{69.8}$$

  converges for $z \in \mathbb{C}$ with $|z| > 1$. We will use this expansion with z^{-1} replaced by $(\alpha\mathcal{B})^{-1}$, where $\alpha \in \mathbb{R}$.

69.3 Metrics

Consider a t-series $\{\underline{x}_t\}_{\forall t}$.

In general, if we have a metric like Auto-covariance (ACov) that is defined for $\tau = 1, 2, 3, \ldots$, it is conventional in time series analysis to refer to the plot of that metric for all values of τ as the Auto-covariance *Function* (ACovF).

- **Expected value and Variance**

$$E[\underline{x}_t] \tag{69.9}$$

$$Var[\underline{x}_t] = \langle \underline{x}_t, \underline{x}_t \rangle \tag{69.10}$$

- **Auto-covariance (ACov)**

$$\gamma_{t,t+\tau} = \left\langle \underline{x}_t, \underline{x}_{t+\tau} \right\rangle \tag{69.11}$$

- **Auto-correlation (ACorr)** (assumes w-stationarity)

$$\rho(\tau) = \frac{\gamma(\tau)}{\gamma(0)} \tag{69.12}$$

- **Generating function of auto-covariance** (assumes w-stationarity)

$$\tilde{\gamma}(z) = \sum_{\tau=-\infty}^{\infty} \gamma(\tau) z^{\tau} \tag{69.13}$$

  Note that this transform is double sided. Fourier Transform if $z = e^{-i\omega\tau}$.

- **Expected value and variance conditioned on all past information**

  For $\tau = 1, 2, 3, \ldots,$

$$E_{|x_{\leq t}}[\underline{x}_{t+\tau}] \tag{69.14}$$

$$Var_{|x_{\leq t}}[\underline{x}_{t+\tau}] = \left\langle \underline{x}_{t+\tau}, \underline{x}_{t+\tau} \right\rangle_{|x_{\leq t}} \tag{69.15}$$

- **Partial auto-covariance (PACov)**

  Assume w-stationarity. For $\tau = 1, 2, 3, \ldots$

$$\gamma^{part}(\tau) = \left\langle \underline{x}_t, \underline{x}_{t+\tau} \right\rangle_{|x_{\leq t}, \underline{x}_{t+\tau}} \tag{69.16}$$

  The idea is that we set to zero the nodes $\underline{x}_{(t,t+\tau)} = \{\underline{x}_{t+1}, \underline{x}_{t+2}, \ldots, \underline{x}_{t+\tau-1}\}$ that lie between (but not including) $\underline{x}_t$ and $\underline{x}_{t+\tau}$.

- **Partial auto-correlation (PACorr)**

$$\rho^{part}(\tau) = \frac{\gamma^{part}(\tau)}{\gamma^{part}(0)} \tag{69.17}$$

---

weak stationarity (w-stationarity) means that $E[\underline{x}_t] = \mu$ and $\gamma_{t,t+\tau} = \gamma(\tau)$ are both independent of t. If we have w-stationarity, then

$$\begin{aligned} \gamma(-\tau) &= \left\langle \underline{x}_t, \underline{x}_{t-\tau} \right\rangle && (69.18) \\ &= \left\langle \underline{x}_{t-\tau}, \underline{x}_t \right\rangle && (69.19) \\ &= \gamma(\tau) && (69.20) \end{aligned}$$

We will often abbreviate $\gamma(\tau)$ by γ_τ.

Example of various metrics. If $\{\underline{n}_t\}_{\forall t} \sim WN(0, \sigma^2)$ then

$$E[\underline{n}_t] = 0 \tag{69.21a}$$

$$\gamma(\tau) = \sigma^2 \delta(\tau, 0) \tag{69.21b}$$

$$\gamma(0) = \sigma^2 \tag{69.21c}$$

$$\tilde{\gamma}(z) = \gamma(0) \tag{69.21d}$$

For $\tau > 0$,

$$E_{|n_{\leq t}}[\underline{n}_{t+\tau}] = E[\underline{n}_{t+\tau}] = 0 \tag{69.21e}$$

$$\left\langle \underline{n}_{t+\tau}, \underline{n}_{t+\tau} \right\rangle_{|n_{\leq t}} = E[\underline{n}_{t+\tau}^2] = \sigma^2 \tag{69.21f}$$

69.4 Definition of $ARMA(p, q)$, $AR(p)$ and $MA(q)$.

Suppose $\{\underline{y}_t\}_{\forall t}$ is a zero mean t-series. Hence $\underline{y}_t = \underline{Y}_t - \mu$, $E[\underline{Y}_t] = \mu$, $E[\underline{y}_t] = 0$. $\underline{y}_t$ is said to be the **demeaned** version of Y_t.

Suppose also that $\{\underline{n}_t\}_{\forall t} \sim WN(0, \sigma^2)$.

Then we define the **Auto-Regressive Moving-Average t-series** $ARMA(p, q)$ by

$$\underline{y}_t = \underbrace{\sum_{j=1}^{p} \alpha_j \underline{y}_{t-j}}_{\mathcal{Y}_t^{AR(p)}} + \underline{n}_t + \underbrace{\sum_{j=1}^{q} \nu_j \underline{n}_{t-j}}_{\mathcal{Y}_t^{MA(q)}} \tag{69.22}$$

(α stands for the first letter of "auto-regressive". ν stands for first letter of "noise".)

Special cases

1. **Auto-Regressive t-series** $AR(p)$

$$\underline{y}_t = \mathcal{Y}_t^{AR(p)} + \underline{n}_t \tag{69.23}$$

2. **Moving-Average t-series** $MA(q)$

$$\underline{y}_t = \underline{n}_t + \mathcal{Y}_t^{MA(q)} \tag{69.24}$$

Figure 69.1: $ARMA(2,3)$, $AR(2)$ and $MA(3)$ bnets. For clarity, we show only the arrows entering node $\underline{y}_t$. The full bnet has the same structural pattern of incoming arrows (including the weights α_j, ν_j) for each node $\underline{y}_{t'}$ for all t'.

Fig.69.1 shows the bnets for $ARMA(2,3)$, $AR(2)$ and $MA(3)$. The TPM, printed in blue, for node $\underline{y}_t$ in those bnets, is as follows:

For $ARMA(p,q)$,

$$P(y_t|y_{[t-p,t-1]}, n_{[t-q,t]}) = \mathbb{1}(y_t = \text{see Eq.69.22})) \tag{69.25}$$

For $AR(p)$,

$$P(y_t|y_{[t-p,t-1]}, n_t) = \mathbb{1}(y_t = \text{see Eq.69.23})) \tag{69.26}$$

For $MA(q)$,

$$P(y_t|n_{[t-q,t]}) = \mathbb{1}(y_t = \text{see Eq.69.24})) \tag{69.27}$$

The $\underline{n}_t$ variable is variously referred to as the **external noise, impulse, shock, innovation** at time t.

69.5 Solving $AR(p)$

Suppose $\{\underline{y}_t\}_{\forall t}$ is an $AR(p)$ t-series. Hence

$$\underline{y}_t = \sum_{j=1}^{p} \alpha_j y_{t-j} + \underline{n}_t \tag{69.28}$$

$AR(0)$ satisfies:

$$\underline{y}_t = \underline{n}_t \tag{69.29}$$

See Fig.69.2. This is just white noise.

$AR(1)$ satisfies:

$$\underline{y}_t = \alpha_1 \underline{y}_{t-1} + \underline{n}_t \tag{69.30}$$

See Fig69.2. This is a Markov chain with external i.i.d. noise injected to each node. $AR(1)$ is the discrete form of the so called **Ornstein-Uhlenbeck t-series** (aka as the **Langevin Equation**). When $\alpha_1 = 1$, it is called a **random walk**.

$$\begin{tikzcd}
AR(0) & \cdots \underline{n}_{t-2} \arrow[d, "1"'] & \underline{n}_{t-1} \arrow[d, "1"'] & \underline{n}_t \arrow[d, "1"'] & \underline{n}_{t+1} \cdots \arrow[d, "1"'] \\
 & \cdots \underline{y}_{t-2} & \underline{y}_{t-1} & \underline{y}_t & \underline{y}_{t+1} \cdots \\
AR(1) & \cdots \underline{n}_{t-2} \arrow[d, "1"'] & \underline{n}_{t-1} \arrow[d, "1"'] & \underline{n}_t \arrow[d, "1"'] & \underline{n}_{t+1} \cdots \arrow[d, "1"'] \\
 & \cdots \underline{y}_{t-2} \arrow[r, "\alpha_1"'] & \underline{y}_{t-1} \arrow[r, "\alpha_1"'] & \underline{y}_t \arrow[r, "\alpha_1"'] & \underline{y}_{t+1} \cdots
\end{tikzcd}$$

Figure 69.2: Bnets for $AR(0)$ and $AR(1)$.

Note that

$$\alpha^-(\mathcal{B})\underline{y}_t = \underline{n}_t \tag{69.31}$$

where

$$\alpha^-(\mathcal{B}) = 1 - \sum_{j=1}^{p} \alpha_j \mathcal{B}^j \tag{69.32}$$

Note that we can get $AR(p)$ from $AR(\infty)$ by setting $\alpha_{>p} = 0$.

If $\alpha^-(\beta)$ is invertible, then, using the Taylor expansion Eq.(69.7), we get

$$\underline{y}_t = \alpha'(\mathcal{B})\underline{n}_t \tag{69.33}$$

where

$$\alpha'(\mathcal{B}) = \frac{1}{\alpha^{-}(\mathcal{B})} = 1 + \sum_{k=1}^{\infty} \left[\sum_{j=1}^{p} \alpha_j \mathcal{B}^j \right]^k = \sum_{j=0}^{\infty} \alpha'_j \mathcal{B}^j \tag{69.34}$$

where $\alpha'_0 = 1$.

69.6 Solving $MA(q)$

Suppose $\{\underline{y}_t\}_{\forall t}$ is an $MA(q)$ t-series. Hence,

$$\underline{y}_t = \underline{n}_t + \sum_{j=1}^{q} \nu_j \underline{n}_{t-j} = \sum_{j=0}^{q} \nu_j \underline{n}_{t-j} \tag{69.35}$$

where $\nu_0 = 1$. Thus,

$$\underline{y}_t = \nu(\mathcal{B}) \underline{n}_t \tag{69.36}$$

where

$$\nu(\mathcal{B}) = 1 + \sum_{j=1}^{q} \nu_j \mathcal{B}^j = \sum_{j=0}^{q} \nu_j \mathcal{B}^j \ . \tag{69.37}$$

Note that we can get $MA(q)$ from $MA(\infty)$ by setting $\nu_{>q} = 0$.

Claim 66 *If $\{\underline{y}_t\}_{\forall t}$ is an $MA(q)$ t-series, then*

$$E[\underline{y}_t] = 0 \tag{69.38a}$$

For $\tau \geq 0$,

$$\gamma(\tau) = \mathbb{1}(\tau \leq q) \sigma^2 \sum_{j=0}^{q-\tau} \nu_j \nu_{\tau+j} \tag{69.38b}$$

$$\gamma(0) = \sigma^2 \sum_{j=0}^{q} \nu_j^2 \tag{69.38c}$$

proof:

$$\gamma(\tau) = \left\langle \underline{y}_t, \underline{y}_{t+\tau} \right\rangle \tag{69.39}$$

$$= \left\langle \sum_{j=0}^{q} \nu_j \underline{n}_{t-j}, \sum_{k=0}^{q} \nu_k \underline{n}_{t+\tau-k} \right\rangle \tag{69.40}$$

$$= \sigma^2 \sum_{j=0}^{q} \sum_{k=0}^{q} \nu_j \nu_k \delta(t-j, t+\tau-k) \tag{69.41}$$

$$= \sigma^2 \sum_{j=0}^{q} \sum_{k=0}^{q} \nu_j \nu_k \delta(k, \tau+j) \tag{69.42}$$

$$= \mathbb{1}(\tau \leq q)\sigma^2 \sum_{j=0}^{q-\tau} \nu_j \nu_{\tau+j} \tag{69.43}$$

QED

69.7 Solving $ARMA(p,q)$

Suppose $\{\underline{y}_t\}_{\forall t}$ is an $ARMA(p,q)$ t-series. Hence, using Eqs.(69.31) and (69.36), $\underline{y}_t$ satisfies

$$\alpha^-(\mathcal{B})\underline{y}_t = \nu(\mathcal{B})\underline{n}_t \tag{69.44}$$

If $\alpha^-(\beta)$ is invertible, then, using the Taylor expansion Eq.(69.7), we get

$$\underline{y}_t = \frac{\nu(\mathcal{B})}{\alpha^-(\mathcal{B})}\underline{n}_t = \nu(\mathcal{B})\alpha'(\mathcal{B})\underline{n}_t \tag{69.45}$$

We see that, if the $\alpha^-(\mathcal{B})$ operator is invertible, an $AR(p)$ or an $ARMA(p,q)$ t-series can be represented as an $MA(\infty)$ t-series. $MA(\infty)$ is often called **Wold's Decomposition**. Furthermore, if the $\nu(\mathcal{B})$ operator is invertible, an $MA(q)$ or an $ARMA(p,q)$ t-series can be represented as an $AR(\infty)$ t-series.

The polynomials $\alpha^-(z)$ and $\nu(z)$ can be expressed in factored form $\alpha^-(z) = \prod_{j=1}^{p}(z - z_j^\alpha)$ and $\nu(z) = \prod_{j=1}^{q}(z - z_j^\nu)$. If these two polynomials have a root z_0 in common, both polynomials should be divided by $(z - z_0)$. This reduces an $ARMA(p,q)$ t-series to an $ARMA(p-1,q-1)$ t-series. The bnet for $ARMA(p-1,q-1)$ has one less α_j arrow and one less ν_j arrow than the bnet for $AR(p,q)$.

69.8 Auto-correlation and partial auto-correlation

Note from Eq.(69.38b) that if $\{\underline{y}_t\}_{\forall t}$ is a $MA(q)$ t-series, then $\gamma(\tau) = 0$ for all $\tau > q$. Fig.69.3 gives a graphical proof, using bnets and the d-separation theorem, that for

a $MA(2)$ t-series, $\gamma(\tau) = 0$ for $\tau > 2$. As a consequence of this, a plot of $\gamma(\tau)$ versus τ for a typical $MA(2)$ t-series looks like Fig.69.4.

Figure 69.3: $MA(2)$ bnet. For clarity, we show only arrows entering nodes $\underline{y}_t$ and $\underline{y}_{t+\tau}$ for $\tau = 1, 2, 3$. For $\tau = 1, 2$, there is a path through which information can flow from node $\underline{y}_t$ to node $\underline{y}_{t+\tau}$. For $\tau = 2$, there is no such path.

Figure 69.4: Plot of auto-correlation function (ACorrF) $\rho(\tau)$ and partial auto-correlation function (PACorrF) $\rho^{part}(\tau)$ for an instance of $MA(2)$. Note that $\rho(\tau)$ vanishes for $\tau > 2$.

Claim 67 *If $\{\underline{y}_t\}_{\forall t}$ is an $AR(p)$ t-series, then $\gamma^{part}(\tau) = 0$ for all $\tau > p$.*

proof:

Let

$$\underline{\xi} = \underline{y}_{\leq t}, \underline{y}_{t+\tau} \tag{69.46}$$

Recall that

$$\begin{aligned}
\gamma^{part}(\tau) &= \left\langle \underline{y}_t, \underline{y}_{t+\tau} \right\rangle_{|\xi} && (69.47)\\
&= \left\langle \underline{y}_t \; \underline{y}_{t+\tau} \right\rangle_{|\xi} - \left\langle \underline{y}_t \right\rangle_{|\xi} \left\langle \underline{y}_{t+\tau} \right\rangle_{|\xi} && (69.48)\\
&= \left\langle \underline{y}_t \; \underline{y}_{t+\tau} \right\rangle_{|\xi} && (69.49)
\end{aligned}$$

Define

$$Z_{(t,t+\tau)} = \mathbb{1}(\underline{y}_{(t,t+\tau)} = 0) \tag{69.50}$$

Hence, the operator $Z_{(t,t+\tau)}$ sets all $\underline{y}_{t'}$ with $t < t' < t + \tau$ equal to zero. Note that we can express the PACov as

$$\gamma^{part}(\tau) = \left\langle Z_{(t,t+\tau)} \; (\underline{y}_t \; \underline{y}_{t+\tau}) \right\rangle \; . \tag{69.51}$$

Since

$$\begin{aligned}
\underline{y}_{t+\tau} &= \sum_{j=1}^{p} \alpha_j \underline{y}_{t+\tau-j} + \underline{n}_{t+\tau} && (69.52)\\
&= \alpha_p \underline{y}_{t+\tau-p} + \ldots + \alpha_2 \underline{y}_{t+\tau-2} + \alpha_1 \underline{y}_{t+\tau-1} + \underline{n}_{t+\tau} \; , && (69.53)
\end{aligned}$$

we get

$$Z_{(t,t+\tau)} \underline{y}_{t+\tau} = \underline{n}_{t+\tau} \quad \text{if } \tau > p \; . \tag{69.54}$$

Hence, for $\tau > p$,

$$\gamma^{part}(\tau) = \left\langle \underline{y}_t \; \underline{n}_{t+\tau} \right\rangle = 0 \tag{69.55}$$

QED

Fig.69.5 gives a graphical proof, using bnets and the d-separation theorem, that for an $AR(2)$ t-series, $\gamma^{part}(\tau) = 0$ for $\tau > 2$. As a consequence of this, a plot of $\gamma^{part}(\tau)$ versus τ for a typical $AR(2)$ t-series looks like Fig.69.6.

Figure 69.5: $AR(2)$ bnet. For clarity, we show only arrows entering nodes $\underline{y}_t$ and $\underline{y}_{t+\tau}$ for $\tau = 1, 2, 3$. Yellow nodes are in set $\underline{y}_{(t,t+\tau)}$. They are conditioned on, so information can't flow through them to node $\underline{y}_{t+\tau}$ by the d-separation theorem.

Figure 69.6: Plot of auto-correlation function (ACorrF) $\rho(\tau)$ and partial auto-correlation function (PACorrF) $\rho^{part}(\tau)$ for an instance of $AR(2)$. Note that $\rho^{part}(\tau)$ vanishes for $\tau > 2$.

	ACorr	PACorr
$AR(p)$	tapers off	jumps to zero for $\tau > p$
$MA(q)$	jumps to zero for $\tau > q$	tapers off

Table 69.1: Detecting $AR(p)$ and $MA(q)$ using auto-correlation and partial auto-correlation.

69.9 Generating function of auto-correlation

Claim 68 *If*

$$\alpha(z) = \sum_{\tau=-\infty}^{\infty} \alpha_\tau z^\tau \tag{69.56}$$

then

$$\alpha(z)\alpha(z^{-1}) = \sum_{\tau=-\infty}^{\infty} z^\tau \sum_{j=-\infty}^{\infty} \alpha_j \alpha_{j+\tau} \tag{69.57}$$

proof:

$$\begin{aligned}
\alpha(z)\alpha(z^{-1}) &= \sum_{j'=-\infty}^{\infty} z^{j'} \alpha_{j'} \sum_{j=-\infty}^{\infty} z^{-j} \alpha_j && (69.58)\\
&= \sum_{\tau=-\infty}^{\infty} \sum_{j'=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} z^\tau \alpha_{j'} \alpha_j \delta(j'-j,\tau) && (69.59)\\
&= \sum_{\tau=-\infty}^{\infty} z^\tau \sum_{j=-\infty}^{\infty} \alpha_j \alpha_{j+\tau} && (69.60)
\end{aligned}$$

QED

As an example of this claim, note that

$$(\alpha_0 + \alpha_1 z)(\alpha_0 + \alpha_1 z^{-1}) = \alpha_0\alpha_1 z^{-1} + (\alpha_0^2 + \alpha_1^2) + \alpha_0\alpha_1 z \tag{69.61}$$

Claim 69 *If $\{\underline{y}_t\}_{\forall t}$ is an $AR(p)$ t-series, then*

$$\tilde{\gamma}(z) = \alpha'(z)\sigma^2\alpha'(z^{-1}) \tag{69.62}$$

proof: Left to reader. See Claim 68.
QED

Claim 70 *If $\{\underline{y}_t\}_{\forall t}$ is an $MA(q)$ t-series, then*

$$\tilde{\gamma}(z) = \nu(z)\sigma^2\nu(z^{-1}) \tag{69.63}$$

proof:

$$\underline{y}_t = \sum_{j=-\infty}^{\infty} \nu_j \underline{n}_{t-j} \tag{69.64}$$

$$\begin{aligned}
\tilde{\gamma}(z) &= \sum_{\tau=-\infty}^{\infty} \left\langle \underline{y}_0, \underline{y}_\tau \right\rangle z^\tau && (69.65)\\
&= \sum_{\tau=-\infty}^{\infty} z^\tau \sum_{j=-\infty}^{\infty} \sum_{j'=-\infty}^{\infty} \nu_j \nu_{j'} \underbrace{\left\langle \underline{n}_{-j}, \underline{n}_{\tau-j'} \right\rangle}_{\sigma^2 \delta(j', \tau+j)} && (69.66)\\
&= \sigma^2 \sum_{\tau=-\infty}^{\infty} z^\tau \sum_{j=-\infty}^{\infty} \nu_j \nu_{j+\tau} && (69.67)
\end{aligned}$$

Now use Claim 68.
QED

Claim 71 *If $\{\underline{y}_t\}_{\forall t}$ is an $ARMA(p,q)$ t-series, then*

$$\tilde{\gamma}(z) = \alpha'(z)\nu(z)\sigma^2\nu(z^{-1})\alpha'(z^{-1}) \tag{69.68}$$

proof: Left to reader. See Claim 68.
QED

69.10 Impulse Response

The derivatives

$$IR_\tau = \frac{\partial y_{t+\tau}}{\partial n_t} \tag{69.69}$$

are called **impulse responses** or **dynamic multipliers**. Note that this derivative depends on τ but not t by w-stationarity. A plot of IR_τ versus τ is called the **Impulse Response Function (IRF)**. Examples:

- **Claim 72** *For $MA(q)$,*

$$\frac{\partial y_{t+\tau}}{\partial n_t} = \nu_\tau \mathbb{1}(0 \le \tau \le q) \tag{69.70}$$

where $\nu_0 = 1$. (See Fig.69.7)

proof:

$$\underline{y}_{t+\tau} = \sum_{j=0}^{q} \nu_j \underline{n}_{t+\tau-j} \tag{69.71}$$

so Eq.(69.70) follows.
QED

- **Claim 73** *For $AR(1)$,*

$$\frac{\partial y_{t+\tau}}{\partial n_t} = (\alpha_1)^\tau \mathbb{1}(\tau \geq 0). \tag{69.72}$$

(See Fig.69.8)

proof:

$$(1 - \alpha_1 \mathcal{B}) \underline{y}_{t+\tau} = \underline{n}_{t+\tau} \tag{69.73}$$

Therefore, using the Taylor expansion Eq.(69.7),

$$\underline{y}_{t+\tau} = 1 + \sum_{j=1}^{\infty} \alpha_1^j \mathcal{B}^j \underline{n}_{t+\tau} \tag{69.74}$$

$$= 1 + \sum_{j=1}^{\infty} \alpha_1^j \underline{n}_{t+\tau-j} \tag{69.75}$$

so Eq.(69.72) follows.
QED

In general, IR_τ equals a sum over paths from $\underline{n}_t$ to $\underline{y}_{t+\tau}$. Each path contributes the product of the weights of all the arrows in the path.

$$\begin{tikzcd}
\cdots \underline{n}_t \arrow[drr, "\nu_2"'] & \underline{n}_{t+1} & \underline{n}_{t+2} \cdots \\
\cdots \underline{y}_t & \underline{y}_{t+1} & \underline{y}_{t+2} \cdots
\end{tikzcd}$$

Figure 69.7: Pictorial representation of impulse response $IR_2 = \nu_2$ for a $MA(q)$ t-series with $q \geq 2$.

$$\begin{tikzcd}
\cdots \underline{n}_t \arrow[d, "1"'] & \underline{n}_{t+1} & \underline{n}_{t+2} \cdots \\
\cdots \underline{y}_t \arrow[r, "\alpha_1"'] & \underline{y}_{t+1} \arrow[r, "\alpha_1"'] & \underline{y}_{t+2} \cdots
\end{tikzcd}$$

Figure 69.8: Pictorial representation of impulse response $IR_2 = \alpha_1^2$ for an $AR(1)$ t-series.

69.11 $AR(p)$ and Yule-Walker equations

Claim 74 *(Yule-Walker equations) If $\{\underline{y}_t\}_{\forall t}$ is an $AR(p)$ t-series,*

$$\gamma_\tau = \sum_{j=1}^{p} \gamma_{\tau-j}\alpha_j + \sigma^2\delta(\tau,0) \tag{69.76}$$

for all $\tau \in \mathbb{Z}$, where we are abbreviating $\gamma(j) = \gamma_j$ for all j.

proof:

Recall that

$$\underline{y}_\tau = \sum_{j=1}^{p} \underline{y}_{\tau-j}\alpha_j + \underline{n}_\tau \tag{69.77}$$

Therefore

$$\underbrace{\left\langle \underline{y}_0, \underline{y}_\tau \right\rangle}_{\gamma_\tau} = \sum_{j=1}^{p} \underbrace{\left\langle \underline{y}_0, \underline{y}_{\tau-j} \right\rangle}_{\gamma_{\tau-j}} \alpha_j + \underbrace{\left\langle \underline{y}_0, \underline{n}_\tau \right\rangle}_{\sigma^2\delta(\tau,0)} \tag{69.78}$$

QED

Figure 69.9: Bnet representing Yule-Walker (Christmas Walker) Eqs.(69.76) with $p = 3$. For clarity, we show only arrows entering node $\underline{\gamma}_\tau$. The full bnet has the same structural pattern of incoming arrows (including the α_j) for each node $\gamma_{\tau'}$ for all τ'.

Fig.69.9 shows a bnet for the Yule-Walker Eqs.(69.76. The TPMs, printed in blue, for the τ time slice of that bnet, are as follows:

$$P(\gamma_\tau) = \mathbb{1}\left(\gamma_\tau = \text{ See Eq.(69.76)}\right) \tag{69.79}$$

$$P(c_\tau) = \mathbb{1}(c_\tau = \sigma^2\delta(\tau,0)) \tag{69.80}$$

The Yule-Walker Eqs.(69.76) imply that

$$\underbrace{\begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_p \end{bmatrix}}_{\gamma^p} = \underbrace{\begin{bmatrix} \gamma_0 & \gamma_{-1} & \gamma_{-2} & \cdots & \gamma_{1-p} \\ \gamma_1 & \gamma_0 & \gamma_{-1} & \cdots & \gamma_{2-p} \\ \vdots & & & & \\ \gamma_{p-1} & \gamma_{p-2} & \gamma_{p-3} & \cdots & \gamma_0 \end{bmatrix}}_{\Gamma} \underbrace{\begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_p \end{bmatrix}}_{\alpha^p} \tag{69.81}$$

Hence

$$\gamma^p = \Gamma \alpha^p \ . \tag{69.82}$$

If Γ is invertible,

$$\alpha^p = \Gamma^{-1} \gamma^p \ . \tag{69.83}$$

Once we know α^p, we can solve for σ^2

$$\gamma_0 = \sum_{j=1}^{p} \gamma_{-j} \alpha_j + \sigma^2 \ , \tag{69.84}$$

i.e., Eq.(69.76) for $\tau = 0$.

69.12 Forecasting

Before submerging the reader in the messy details of t-series forecasting, and running the risk of quickly losing him/her, let me explain to the reader that what we are about to do in this section, is, in the final analysis, quite trivial, and boils down to solving simple systems of linear equations. That is all there is to it!! How hard can that be? In reading this section, don't lose sight of that fact and you'll be okay.

Suppose $\{\underline{y}_t\}_{\forall t}$ is an $AR(\infty)$ t-series. For $\tau > 0$, the "orthogonal projection" $\underline{y}_{t+\tau}|\underline{y}_{\leq t}$ is defined as the linear combination of $\underline{y}_{\leq t}$ obtained by doing Linear Regression with x-variables $\underline{y}_{\leq t}$ and y-variable $\underline{y}_{t+\tau}$. Thus

$$E\left[\underline{y}_{t+\tau}|\underline{y}_{\leq t}\right] = E_{|\underline{y}_{\leq t}}[\underline{y}_{t+\tau}] \tag{69.85}$$

and

$$\begin{aligned} Var\left[\underline{y}_{t+\tau}|\underline{y}_{\leq t}\right] &= E\left[\left(\underline{y}_{t+\tau}|\underline{y}_{\leq t} - E[\underline{y}_{t+\tau}|\underline{y}_{\leq t}]\right)^2\right] && (69.86) \\ &= Var_{|\underline{y}_{\leq t}}[\underline{y}_{t+\tau}] \ . && (69.87) \end{aligned}$$

For example, for $\tau = 1$,

$$\underline{y}_{t+1} = \sum_{j=1}^{p} \alpha_j \underline{y}_{t+1-j} + \underline{n}_{t+1} \tag{69.88}$$

$$\underline{y}_{t+1}|\underline{y}_{\leq t} = \sum_{j=1}^{p} \alpha_j \underline{y}_{t+1-j} \tag{69.89}$$

$$\underline{y}_{t+1} - \underline{y}_{t+1|\underline{y}\leq t} = \underline{n}_{t+1} \tag{69.90}$$

$$\begin{aligned} E\left[\underline{y}_{t+1}|\underline{y}_{\leq t}\right] &= E_{|\underline{y}_{\leq t}}[\underline{y}_{t+1}] && (69.91) \\ &= 0 && (69.92) \end{aligned}$$

$$\begin{aligned} Var\left[\underline{y}_{t+1}|\underline{y}_{\leq t}\right] &= Var_{|\underline{y}_{\leq t}}[\underline{y}_{t+1}] && (69.93) \\ &= \sigma^2 && (69.94) \end{aligned}$$

If $\{\underline{y}_t\}_{\forall t}$ is an $AR(\infty)$ t-series and $\tau > 0$, by **forecasting** $\underline{y}_{t+\tau}|\underline{y}_{\leq t}$, we will mean finding $\underline{y}_{t+\tau}|\underline{y}_{\leq t}$, given $y_{\leq t}$ as input data or some other input data from which $y_{\leq t}$ can be derived. Fig.69.10 illustrates 3 types of input data that we will consider next.

Note that if $\{\underline{y}_t\}_{\forall t}$ is an $AR(\infty)$ t-series, then $\alpha^-(\mathcal{B})\underline{y}_t = n_t$. Thus, $\underline{n}_{\leq t}$ can be expressed in terms of $\underline{y}_{\leq t}$, yielding a function $\underline{n}_{\leq t}(\underline{y}_{\leq t})$. If the operator $\alpha^-(\mathcal{B})$ is invertible, the opposite is also true: $\underline{y}_{\leq t}$ can be expressed in terms of $\underline{n}_{\leq t}$, yielding a function $\underline{y}_{\leq t}(\underline{n}_{\leq t})$. Thus,

$$\underline{y}_{t+\tau}|\underline{y}_{\leq t} = \underline{y}_{t+\tau}|\underline{y}_{\leq t}(\underline{n}_{\leq t}) = \underline{y}_{t+\tau}|\underline{n}_{\leq t} \tag{69.95}$$

We've seen that if we assume invertibility of $\alpha^-(\mathcal{B})$, then, in order to forecast $\underline{y}_{t+\tau}|\underline{y}_{\leq t}$, we have the option of considering $\{\underline{y}_t\}_{\forall t}$ to be either an $AR(\infty)$ or an $MA(\infty)$ t-series. We will assume the latter, because this simplifies the algebra necessary to calculate both $\underline{y}_{t+\tau}|\underline{y}_{\leq t}$ and its variance.

(a) **find** $\underline{y}_{t+\tau}|y_{\leq t}$ **given** $\underline{y}_{\leq t}$

Suppose $\{\underline{y}_t\}_{\forall t}$ is an $MA(\infty)$ t-series. Then

Figure 69.10: Bnet for $MA(\infty)$ for forecasting $\underline{y}_{t+\tau}|\underline{y}_{\leq t}$. Input data in red. For clarity, we show only the arrows entering node $\underline{y}_t$.

$$\underline{y}_{t+\tau} = \sum_{j=0}^{\infty} \nu_j \underline{n}_{t+\tau-j} \tag{69.96}$$

$$= \underbrace{\sum_{j=0}^{\tau-1} \nu_j \underline{n}_{t+\tau-j}}_{\underline{y}_{t+\tau} - \underline{y}_{t+\tau}|y_{\leq t}} + \underbrace{\sum_{j=\tau}^{\infty} \nu_j \underline{n}_{t+\tau-j}}_{\underline{y}_{t+\tau}|y_{\leq t}} \tag{69.97}$$

$$\underline{y}_{t+\tau}|y_{\leq t} = \sum_{j=\tau}^{\infty} \nu_j \underline{n}_{t+\tau-j} \tag{69.98}$$

$$E_{|y_{\leq t}}[\underline{y}_{t+\tau}] = E[\underline{y}_{t+\tau}|y_{\leq t}] = 0^1 \tag{69.99}$$

$$\left\langle \underline{y}_{t+\tau}\underline{y}_{t+\tau} \right\rangle_{|y_{\leq t}} = \left\langle \underline{y}_{t+\tau} - \underline{y}_{t+\tau}|y_{\leq t},\ \underline{y}_{t+\tau} - \underline{y}_{t+\tau}|y_{\leq t} \right\rangle \tag{69.100}$$

$$= \sum_{j=0}^{\tau-1} \nu_j^2 \tag{69.101}$$

Figure 69.11: $MA(2)$ bnet. For clarity, we show only arrows entering node $\underline{y}_{t+\tau}$ for $\tau = 1, 2, 3$. Green arrows originate at a time after time t.

For $MA(q)$, because $\nu_j = 0$ for $j > q$, this reduces to

$$\underline{y}_{t+\tau}|y_{\leq t} = \mathbb{1}(\tau \leq q) \sum_{j=\tau}^{q} \nu_j \underline{n}_{t+\tau-j} \tag{69.102}$$

$$E_{|y_{\leq t}}[\underline{y}_{t+\tau}] = E[\underline{y}_{t+\tau}|y_{\leq t}] = 0 \tag{69.103}$$

$$\left\langle \underline{y}_{t+\tau}, \underline{y}_{t+\tau} \right\rangle_{|y_{\leq t}} = \sum_{j=0}^{\min(\tau-1,q)} \nu_j^2 \tag{69.104}$$

[1]Remember that $\underline{y}_t$ has zero mean by definition. $Y_t = y_t + \mu$ for some time independent constant μ.

Eq.(69.104) is motivated by Fig.69.11). Only the green arrows entering $\underline{y}_{t+\tau}$ and originating in a node $\underline{n}_{t'}$ with $t' > t$ contribute to the right hand side of Eq.(69.104).

Thus, the mean of $\underline{y}_{t+\tau}$ can be predicted to remain zero for all $\tau > 0$ (forever). The error in that prediction increases with τ until τ reaches q. After that, the error remains constant.

Note that

$$\begin{aligned}
\underline{y}_{t+\tau}|y_{\leq t} &= \sum_{j=\tau}^{\infty} \nu_j \underline{n}_{t+\tau-j} && (69.105)\\
&= \sum_{j=\tau}^{\infty} \nu_j \mathcal{B}^{j-\tau} \underline{n}_t && (69.106)\\
&= \left[\sum_{j=0}^{\infty} \nu_j \mathcal{B}^{j-\tau}\right]_{\mathcal{B}^{\geq 0}} \underline{n}_t && (69.107)\\
&= \left[\mathcal{B}^{-\tau}\nu(\mathcal{B})\right]_{\mathcal{B}^{\geq 0}} \underline{n}_t && (69.108)
\end{aligned}$$

$\mathcal{B}^{\geq 0}$ means only the non-negative powers of $\mathcal{B}$ are kept. Eq.(69.108) is just Eq.(69.98) in fancy notation.

In $MA(\infty)$, if $\nu(\mathcal{B})$ is invertible, then

$$\underline{n}_t = \nu(\mathcal{B})^{-1}\underline{y}_t \tag{69.109}$$

so

$$\boxed{\underline{y}_{t+\tau}|y_{\leq t} = \mathcal{L}_{WK}\underline{y}_t} \tag{69.110}$$

where

$$\mathcal{L}_{WK} = \left[\mathcal{B}^{-\tau}\nu(\mathcal{B})\right]_{\mathcal{B}^{\geq 0}} \nu(\mathcal{B})^{-1} \tag{69.111}$$

Eq.(69.110) is called the **Wiener-Kolgomorov prediction formula (WKPF)**. Next, we apply WKPF to $AR(1)$ and $MA(1)$ as examples of its usage.

- $AR(1)$

$$\underbrace{(1-\alpha_1\mathcal{B})}_{\nu(\mathcal{B})^{-1}}\underline{y}_t = \underline{n}_t \tag{69.112}$$

$$\underline{y}_t = \underbrace{\left[\sum_{j=0}^{\infty}(\alpha_1\mathcal{B})^j\right]}_{\nu(\mathcal{B})}\underline{n}_t \tag{69.113}$$

$$\nu_j = (\alpha_1)^j \tag{69.114}$$

$$\left[\mathcal{B}^{-\tau}\nu(\mathcal{B})\right]_{\mathcal{B}^{\geq 0}} = (\alpha_1)^\tau \nu(\mathcal{B}) \tag{69.115}$$

$$\underline{y}_{t+\tau}|y_{\leq t} = (\alpha_1)^\tau \underline{y}_t \tag{69.116}$$

Hence, $\underline{y}_{t+\tau}|y_{\leq t}$ decreases geometrically as τ grows.

- $MA(1)$

$$\underline{y}_t = \underbrace{(1+\nu_1\mathcal{B})}_{\nu(\mathcal{B})}\underline{n}_t \tag{69.117}$$

$$\left[\mathcal{B}^{-\tau}\nu(\mathcal{B})\right]_{\mathcal{B}^{\geq 0}} = \delta(\tau,1)\nu_1 \tag{69.118}$$

$$\begin{aligned}\underline{y}_{t+\tau}|y_{\leq t} &= \delta(\tau,1)\nu_1 \underbrace{\left[\sum_{j=0}^{\infty}(-\nu_1\mathcal{B})^j\right]}_{\nu(\mathcal{B})^{-1}}\underline{y}_t && (69.119)\\ &= \delta(\tau,1)\nu_1\underline{n}_t && (69.120)\end{aligned}$$

(b) **find** $\underline{y}_{t+\tau}|y_{\leq t}$ **given** $\underline{y}_{\leq t}$**, but, for some** $m > 0$**,** $\underline{y}_{\leq t-m} \approx 0$ **(approximation)**

$$\underline{y}_{t+\tau}|\underline{y}_{\leq \tau} \approx [\mathcal{L}_{WK}]_{\mathcal{B}^{<m}}\,\underline{y}_t \tag{69.121}$$

(c) **find** $\underline{y}_{t+\tau}|y_{\leq t}$ **given** $\gamma_{[0,m-1]}$ **and** $\gamma^{part}_{[\tau,\tau+m-1]}$

$$\underline{y}_{t+\tau}|\underline{y}_{\leq t} = \sum_{j=0}^{m-1}\underline{y}_{t-j}\beta_j \tag{69.122}$$

Suppose that $k \in \mathbb{Z}_{[0,m-1]}$.

Note that

$$\underline{y}_{t-k}|\underline{y}_{\leq t} = \underline{y}_{t-k}\ . \tag{69.123}$$

$$\underbrace{\left\langle \underline{y}_{t-k}|\underline{y}_{\leq t} \ , \ \underline{y}_{t+\tau}|\underline{y}_{\leq t} \right\rangle}_{\gamma^{part}_{\tau+k}} = \sum_{j=0}^{m-1} \underbrace{\left\langle \underline{y}_{t-k} \ , \ \underline{y}_{t-j} \right\rangle}_{\gamma_{k-j}} \beta_j \tag{69.124}$$

$$\begin{bmatrix} \gamma^{part}_{\tau} \\ \gamma^{part}_{\tau+1} \\ \vdots \\ \gamma^{part}_{\tau+m-1} \end{bmatrix} = \underbrace{\begin{bmatrix} \gamma_0 & \gamma_{-1} & \gamma_{-2} & \cdots & \gamma_{1-m} \\ \gamma_1 & \gamma_0 & \gamma_{-1} & \cdots & \gamma_{2-m} \\ \vdots & & & & \\ \gamma_{m-1} & \gamma_{m-2} & \gamma_{m-3} & \cdots & \gamma_0 \end{bmatrix}}_{\Gamma} \underbrace{\begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{m-1} \end{bmatrix}}_{\beta^m} \tag{69.125}$$

Solve Eq.(69.125) for β^m and plug β^m into Eq.(69.122).

69.13 Model Learning

Box-Jenkins Method

1. **Filtering Data (FD)**

   Removing trends and periodicity (seasonality) via differencing, so as to achieve a w-stationary t-series. We deal with FD in Section 69.14.

2. **Model Selection**

   Finding best possible p, q for $ARMA(p, q)$ using various statistical tests and metrics.

3. **Parameter Learning (PL)**

   Finding best possible coefficients α_j and ν_j. We deal with PL in Section 69.15.

4. **Testing**

   Measuring the goodness of fit.

69.14 Differencing and $ARIMA(p, d, q)$

Let $\alpha \in (0, 1)$. We say that $\{\underline{s}_t\}_{\forall t}$ is a **Simple Exponential Smoothing (SES) t-series** if it satisfies

$$s_t = (1 - \alpha)s_{t-1} + \alpha x_t \tag{69.126}$$

Figure 69.12: Bnet for Simple Exponential Smoothing t-series. $\alpha \in (0, 1)$

A SES t-series can be represented by the bnet of Fig.69.12. The TPM, printed in blue, for node $\underline{s}_t$, is as follows

$$P(s_t) = \mathbb{1}(s_t = \text{ see Eq.(69.126)}) \tag{69.127}$$

One has

$$(1 - (1 - \alpha)\mathcal{B})\underline{s}_t = \alpha \underline{x}_t \tag{69.128}$$

so

$$\underline{s}_t = \frac{\alpha}{1 - (1 - \alpha)\mathcal{B}} \underline{x}_t \tag{69.129}$$

$$= \alpha \sum_{j=0}^{\infty} (1 - \alpha)^j \mathcal{B}^j \underline{x}_t \quad \text{(by Eq.(69.7))} \tag{69.130}$$

Note from Fig.69.12 and Eq.(69.130) that each shock $\underline{x}_{t-j}$ with $j > 0$ contributes to $\underline{s}_t$ proportionally to the product $\alpha(1 - \alpha)^j$ of arrow weights in the path from $\underline{x}_{t-j}$ to $\underline{s}_t$. For example, $\underline{x}_{t-2}$ contributes to $\underline{s}_t$ in the proportion $\alpha(1 - \alpha)^2$. The contributions of $\underline{x}_{t-j}$ to $\underline{s}_t$ decrease geometrically as the lag j increases.

Henceforth, we will abbreviate "**differencing**" by " **diff**".

- **First order diff operator**

$$\Delta x_t = x_t - x_{t-1} = (1 - \mathcal{B})x_t \tag{69.131}$$

Seasonal first order diff:

$$\Delta_{sea} x_t = x_t - x_{t-ts} \tag{69.132}$$

where ts is length of season.

- **Second order diff operator**

$$\Delta^2 x_t = \Delta x_t - \Delta x_{t-1} \tag{69.133}$$

$$= x_t - 2x_{t-1} + x_{t-2} \tag{69.134}$$

$$= (1 - 2\mathcal{B} + \mathcal{B}^2)x_t \tag{69.135}$$

$$= (1 - \mathcal{B})^2 x_t \tag{69.136}$$

- **d-th order diff operator**

$$\Delta^d \underline{x}_t = (1-\mathcal{B})^d \underline{x}_t \tag{69.137}$$

Recall that if $\{\underline{y}_t\}_{\forall t}$ is an $ARMA(p+d,q)$ t-series,

$$\left(1-\sum_{j=1}^{p+d}\alpha_j\mathcal{B}^j\right)\underline{y}_t = \left(1+\sum_{k=1}^{q}\nu_k\mathcal{B}^k\right)\underline{n}_t \tag{69.138}$$

We will say that $\{\underline{y}_t\}_{\forall t}$ is an **Autoregressive Integrated Moving Average** $ARIMA(p,d,q)$ **t-series** if

$$\left(1-\sum_{j=1}^{p}\alpha_j\mathcal{B}^j\right)\underbrace{\Delta^d\underline{y}_t}_{(1-\mathcal{B})^d\underline{y}_t} = \left(1+\sum_{k=1}^{q}\nu_k\mathcal{B}^k\right)\underline{n}_t \tag{69.139}$$

So an $ARIMA(p,d,q)$ t-series is a special case of an $ARMA(p+d,q)$ t-series.

Define $\mathcal{O}_{BEF}$ and $\mathcal{O}_{AFT}$ by

$$\mathcal{O}_{BEF} = 1-\alpha_1\mathcal{B}-\alpha_2\mathcal{B}^2-\alpha_3\mathcal{B}^3 \tag{69.140}$$

$$\begin{aligned}
\mathcal{O}_{AFT} &= \mathcal{O}_{BEF}(1-\mathcal{B}) && (69.141)\\
&= \left\{\begin{array}{lllll} 1 & -\alpha_1\mathcal{B} & -\alpha_2\mathcal{B}^2 & -\alpha_3\mathcal{B}^3 & \\ & -\mathcal{B} & +\alpha_1\mathcal{B}^2 & +\alpha_2\mathcal{B}^3 & +\alpha_3\mathcal{B}^4 \end{array}\right. && (69.142)\\
&= 1-\underbrace{(\alpha_1+1)}_{\alpha_1'}\mathcal{B}-\underbrace{(\alpha_2-\alpha_1)}_{\alpha_2'}\mathcal{B}^2-\underbrace{(\alpha_3-\alpha_2)}_{\alpha_3'}\mathcal{B}^3-\underbrace{(0-\alpha_3)}_{\alpha_4'}\mathcal{B}^4 && (69.143)
\end{aligned}$$

$\mathcal{O}_{BEF}\underline{y}_t$ and $\mathcal{O}_{AFT}\underline{y}_t$ are illustrated in Fig.69.13. Note that in going from BEF to AFT, a new arrow is introduced with weight $\alpha_4' = -\alpha_3$. The intermediate length arrows $\underline{y}_{t-3} \to \underline{y}_t$ and $\underline{y}_{t-2} \to \underline{y}_t$ have weights $\alpha_3' = \alpha_3 - \alpha_2$ and $\alpha_2' = \alpha_2 - \alpha_1$ so if $\alpha_j \approx \alpha_{j-1}$, then those 2 arrows have negligible weights. Only the the longest arrow $\underline{y}_{t-4} \to \underline{y}_t$ and the shortest arrow $\underline{y}_{t-1} \to \underline{y}_t$ have non-negligible weights.

Suppose

$$\underline{y}_t \approx a+bt+ct^2+dt^3 \tag{69.144}$$

This smooth, low-order polynomial-fit to the t-series is called its **trend**.

$$\Delta\underline{y}_t \approx \Delta t\frac{d}{dt}\ \underline{y}_t = b+2ct+3dt^2 \tag{69.145}$$

$$\Delta^2\underline{y}_t \approx (\Delta t)^2\frac{d^2}{dt^2}\ \underline{y}_t = 2c+6dt \tag{69.146}$$

Figure 69.13: Effect on $\underline{y}_t$ of going from before (BEF) with a t-series $ARIMA(3,0,0)$ to after (AFT) with a t-series $ARIMA(3,1,0)$.

$$\Delta^3 \underline{y}_t \approx (\Delta t)^3 \frac{d^2}{dt^3} \ \underline{y}_t = 6d \tag{69.147}$$

Note that the mean $6d$ in $\Delta^3 \underline{y}_t$ can be adjusted to zero (demeaning). From this example, we see that every time we apply a diff operator to a time series, we reduce the order of its polynomial trend by one.

Note that if $\{\underline{y}_t\}_{\forall t}$ is an $ARIMA(0,1,1)$ t-series,

$$\Delta \underline{y}_t = (1 + \nu_1 \mathcal{B}) \underline{n}_t \tag{69.148}$$

$$\underline{y}_t = \underline{y}_{t-1} + \underline{n}_t + \nu_1 \underline{n}_{t-1} \tag{69.149}$$

Claim 75 *$ARIMA(0,1,1)$ is equivalent to a simple exponential smoothing t-series.*

proof: Setting

$$\underline{n}_t = \underline{y}_t - \hat{\underline{y}}_t \tag{69.150}$$

in Eq.(69.148), we get

$$(1 - \mathcal{B}) \underline{y}_t = (1 - \nu_1 \mathcal{B})(\underline{y}_t - \hat{\underline{y}}_t) \tag{69.151}$$

$$(1 - \nu_1 \mathcal{B}) \hat{\underline{y}}_t = (1 - \nu_1) \mathcal{B} \underline{y}_t \tag{69.152}$$

$$\hat{\underline{y}}_t = \nu_1 \hat{\underline{y}}_{t-1} + (1 - \nu_1) \underline{y}_{t-1} \tag{69.153}$$

Now replace $\hat{\underline{y}}_t \to \underline{s}_t$, $\underline{y}_{t-1} \to \underline{n}_t$ and $\nu_1 \to 1 - \alpha$.

Note that this proof has established the equivalence of two different bnets. Those two bnets are pictured in Fig.69.14.

QED

Figure 69.14: Bnets for two t-series that were proven equivalent in Claim 75. Note that the 2 bnets have NO arrows in common!

69.15 Parameter Learning

In this section, we will show how to find an estimate $\hat{\theta}$ for the parameters $\theta = (\{\alpha_j\}_{\forall j}, \{\nu_j\}_{\forall j}, \sigma^2)$ used in $ARMA(p,q)$. We do this **parameter learning (PL)** by postulating a "quasi" (log-) likelihood function $\mathcal{L}(\theta)$, and maximizing that over θ. The estimate $\hat{\theta}$ that we obtain is called the **Quasi-Maximum Likelihood Estimate (QMLE)**. The reason for using the prefix "quasi" is that the likelihood probability involved doesn't really satisfy the i.i.d assumption.

69.15.1 PL of $AR(p)$

Figure 69.15: Bnet for PL of $AR(3)$. For clarity, we show only the arrows entering nodes $\underline{y}_t$ and $\underline{n}_t$.

Recall that if $\{\underline{y}_t\}_{\forall t}$ is an $AR(p)$ t-series,

$$y_t = \sum_{j=1}^{p} \alpha_j y_{t-j} + n_t \tag{69.154}$$

Let $\theta = (\sigma^2, \alpha_{[1,p]})$. We want to learn the parameters θ of the $AR(p)$ bnet Fig.69.15, assuming the TPMs, printed in blue, are as follows.

For $t \geq p+1$

$$P(y_t|y_{<t}, n_t, \theta) = \mathbb{1}\left(y_t = \text{see Eq.(69.154)}\right) \tag{69.155}$$

$$P(n_t|\theta) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[\frac{-(n_t)^2}{2\sigma^2}\right] \tag{69.156}$$

Note that

$$P(y_{\leq t}|\theta) = \sum_{n_{[p+1,t]}} \left\{ \prod_{\tau\in[p+1,t]} P(y_\tau|y_{[\tau-p,\tau-1]}, n_\tau) P(n_\tau|\theta) \right\} P(y_{\leq p}|\theta) \tag{69.157}$$

$$= \prod_{\tau\in[p+1,t]} \left\{ \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[\frac{-\left(y_\tau - \sum_{j=1}^{p} \alpha_j y_{\tau-j}\right)^2}{2\sigma^2}\right] \right\} P(y_{\leq p}|\theta) \tag{69.158}$$

The log likelihood function $\mathcal{L}_t(\theta)$ for this bnet is defined by

$$\mathcal{L}_t(\theta) = \frac{1}{t} \ln P(y_{\leq t}|\theta) \ . \tag{69.159}$$

Hence,

$$\mathcal{L}_t(\theta) \quad = \quad \underbrace{\frac{1}{t} \ln P(y_{\leq p}|\theta)}_{\mathcal{L}_t^{prior}(\theta)} + \mathcal{L}'_t(\theta) \tag{69.160a}$$

where

$$\mathcal{L}'_t(\theta) = \frac{-(t-p)}{t} \ln\sqrt{2\pi\sigma^2} - \frac{1}{2t\sigma^2} \sum_{\tau=p+1}^{t} \left(y_\tau - \sum_{j=1}^{p} \alpha_j y_{\tau-j}\right)^2 \ . \tag{69.160b}$$

Henceforth, we wll assume that $y_{[1,t]}$ is known.

We start by assuming that the prior log-likelihood $\mathcal{L}_t^{prior}(\theta)$ is zero. This means that we have uniform (uninformative) priors. Later on, we will consider instead a Gaussian prior that uses the info that $y_{\leq p}$ is known a priori.

If we assume that $y_{[1,t]}$ is known, then we can plug this info into $\mathcal{L}_t'(\theta)$ and use a non-linear optimization method to maximize $\mathcal{L}_t'(\theta)$ and find the optimum θ.

Alternatively, one can use LR. Define the strange looking dataset

$$\mathcal{D} = \{(t, y_{[t-p,t-1]}, \boxed{y_t}) : t\} \tag{69.161}$$

This dataset is strange because it replaces sampling from a population with sampling in time. (i.e., a "stochastic" average by an "ergodic" average.) The datasets that we use to do LR usually satisfy the i.i.d. assumption, but that assumption does not necessarily hold for this one. Luckily, that assumption is not necessary for doing LR with x-variables $y_{[t-p,t-1]}$, y-variable y_t, and regression coefficients $\alpha_{[1,p]}$. The LR software gives estimates $\hat{\alpha}_{[1,p]}$ that we can use to calculate residuals

$$\epsilon_\tau = y_\tau - \sum_{j=1}^{p} \hat{\alpha}_j y_{\tau-j} \tag{69.162}$$

One can estimate σ^2 from those residuals using

$$\widehat{\sigma^2} = \frac{1}{t-p} \sum_{\tau=p+1}^{t} \epsilon_\tau^2 \ . \tag{69.163}$$

So far, we have assumed an uninformative prior. Alternatively, one can assume the Gaussian prior

$$P(y_{\leq p}|\theta) = \frac{1}{(2\pi\sigma^2)^{\frac{p}{2}}} \exp\left[-y_{\leq p}^T \frac{\Gamma^{-1}}{2} y_{\leq p}\right] \tag{69.164}$$

where

$$\Gamma_{i,j} = \left\langle \underline{y}_i, \underline{y}_j \right\rangle = \gamma_{j-i} \tag{69.165}$$

for $i, j \in \mathbb{Z}_{[1,p]}$. We are assuming that all $\underline{y}_\tau$ have been demeaned; i.e., the mean

$$\hat{\mu} = \frac{1}{t} \sum_{\tau=1}^{t} y_\tau \tag{69.166}$$

has been subtracted from each y_τ for $\tau \in \mathbb{Z}_{[1,t]}$.

Figure 69.16: Bnet for PL of $MA(3)$. For clarity, we show only the arrows entering nodes $\underline{y}_t$ and n_t.

69.15.2 PL of $MA(q)$

Recall that if $\{\underline{y}_t\}_{\forall t}$ is an $MA(q)$ t-series,

$$y_t = \sum_{j=1}^{q} \nu_j n_{t-j} + n_t \tag{69.167}$$

Let $\theta = (\sigma^2, \nu_{[1,q]})$. We want to learn the parameters θ of the $MA(q)$ bnet Fig.69.16, assuming the TPMs, printed in blue, are as follows.

For $t \geq q+1$, let

$$P(y_t|n_{\leq t}, \theta) = \mathbb{1}\left(y_t = \text{see Eq.(69.167)}\right) \tag{69.168}$$

$$P(n_t|\theta) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[\frac{-(n_t)^2}{2\sigma^2}\right] \tag{69.169}$$

Note that

$$P(y_{\leq t}|\theta) = \sum_{n_{[q+1,t]}} \left\{ \prod_{\tau \in [q+1,t]} P(y_\tau|n_{[\tau-q,\tau-1]}, n_\tau) P(n_\tau|\theta) \right\} P(n_{\leq q}|\theta) \tag{69.170}$$

$$= \prod_{\tau \in [q+1,t]} \left\{ \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[\frac{-\left(y_\tau - \sum_{j=1}^{q} \nu_j n_{\tau-j}\right)^2}{2\sigma^2}\right] \right\} P(n_{\leq q}|\theta) \tag{69.171}$$

The log likelihood function $\mathcal{L}_t(\theta)$ for this bnet is defined by

$$\mathcal{L}_t(\theta) = \frac{1}{t} \ln P(y_{\leq t}|\theta) \tag{69.172}$$

$$\mathcal{L}_t(\theta) \quad = \quad \underbrace{\frac{1}{t}\ln P(y_{\leq q}|\theta)}_{\mathcal{L}_t^{prior}(\theta)} + \mathcal{L}'_t(\theta) \tag{69.173a}$$

where

$$\mathcal{L}'_t(\theta) = \frac{-(t-q)}{t}\ln\sqrt{2\pi\sigma^2} - \frac{1}{2t\sigma^2}\sum_{\tau=q+1}^{t}\left(y_\tau - \sum_{j=1}^{q}\nu_j n_{\tau-j}\right)^2 \tag{69.173b}$$

Henceforth, we will assume that $n_{\leq q} = 0$. By Eq.(69.167), this implies that $y_{\leq q} = 0$, so the prior log-likelihood $\mathcal{L}_t^{prior}(\theta)$ is independent of θ.

Figure 69.17: Bnet for PL of $MA(2)$. We assume $\underline{n}_1 = \underline{n}_2 = 0$. For clarity, we show only the arrows entering node $\underline{y}_{2+\tau}$ for $\tau = 1, 2, 3$.

Eq.(69.173) expresses $\mathcal{L}'_t(\theta)$ in terms of y_τ's and n_τ's, but the input data consist of only y_τ's. Luckily, Eq.(69.167) can be used to express all the $n_{\leq t}$ in terms of the $y_{\leq t}$, assuming the input $n_{\leq q} = 0$. Next, we will explain how to do this. For definiteness and simplicity, we will assume $q = 2$ and $\nu_1 = \nu_2 = 1$, and we will use Fig.69.17 as a pedagogical aid.

For $q = 2$, Eq.(69.167) gives $\underline{y}_\tau$ as a linear combination of 3 $\underline{n}$ nodes, $\underline{n}_\tau, \underline{n}_{\tau-1}$ and $\underline{n}_{\tau-2}$ But since it's a linear equation, it can be used to express the latest $\underline{n}$ node, i.e., $\underline{n}_\tau$, in terms of $\underline{y}_\tau$ and the 2 previous $\underline{n}$ nodes.

Hence, we see from Fig.69.17 that:

From $\underline{n}_1 = \underline{n}_2 = 0$, we get $\underline{y}_1 = \underline{y}_2 = 0$.

From the $\tau = 1$ bnet, we get

$$\underline{n}_3 = \underline{y}_3 \tag{69.174}$$

From the $\tau = 2$ bnet, we get

$$\underline{n}_4 = \underline{y}_4 - \underline{n}_3 \tag{69.175}$$

From the $\tau = 3$ bnet, we get

$$\underline{n}_5 = \underline{y}_5 - \underline{n}_3 - \underline{n}_4 \tag{69.176}$$

69.15.3 PL of $ARMA(p, q)$

Figure 69.18: Bnet for PL of $AR(3, 2)$. For clarity, we show only the arrows entering nodes $\underline{y}_t$ and $\underline{n}_t$.

Recall that if $\{\underline{y}_t\}_{\forall t}$ is an $AR(p, q)$ t-series, then

$$y_t = \sum_{j=1}^{p} \alpha_j y_{t-j} + n_t + \sum_{j=1}^{q} \nu_j y_{t-j} \tag{69.177}$$

Let $\theta = (\sigma^2, \alpha_{[1,p]}, \nu_{[1,q]})$. We want to learn the parameters θ of the $ARMA(p, q)$ bnet Fig.69.18, in which the TPMs, printed in blue, are as follows.

For $t \geq \max(p + 1, q + 1)$, let

$$P(y_t | y_{<t}, n_{\leq t}, \theta) = \mathbb{1}\left(y_t = \text{see Eq.(69.177)}\right) \tag{69.178}$$

$$P(n_t | \theta) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[\frac{-(n_t)^2}{2\sigma^2}\right] \tag{69.179}$$

As we did for $AR(p)$ and $MA(q)$, we can now calculate a log likelihood function and maximize it to obtain the parameters θ. But first we will have to assume that $n_{\leq q} = 0$ and $y_{[1,t]}$ is known, and we will have to express the unknown $n_{[q+1,t]}$ in terms of the known $y_{[1,t]}$ using the t-series definition Eq.(69.177).

69.16 $VAR(p)$

Let $\underline{\vec{x}}_t = (\underline{x}_t^1, \underline{x}_t^2, \ldots, \underline{x}_t^{nr})^T \in \mathbb{R}^{nr\times 1}$. Thus, $\underline{\vec{x}}_t$ for each t is a column vector with nr rows. A **vector time series**, denoted variously by $\{\underline{\vec{x}}_1, \underline{\vec{x}}_2, \ldots, \underline{\vec{x}}_{nt}\} = \{\underline{\vec{x}}_t\}_{t=1}^{nt} = \{\underline{\vec{x}}_t\}_{\forall t}$, is a set of $\mathbb{R}^{nr\times 1}$ vectors index by a discrete set of times $\mathbb{Z}_{[0,nt]}$.

Henceforth, we will use the Einstein summation convention for uppercase indices like $A, B, C \in \mathbb{Z}_{[1,nr]}$; i.e., they should be summed over from 1 to nr if they are repeated in a product. For example, $x^A y^A = \sum_{A=1}^{nr} x^A y^A$.

By **vector white noise** $\{\underline{\vec{n}}_t\}_{\forall t} \sim WN(0, \Sigma)$, where $\Sigma \in \mathbb{R}^{nr\times nr}$, we mean a vector t-series $\{\underline{\vec{n}}_t\}_{\forall t}$ that satisfies, for $A, B \in \mathbb{Z}_{[1,nr]}$ and $t, t' \in \mathbb{Z}_{[1,nt]}$,

$$E[\underline{n}_t^A] = 0 \tag{69.180}$$

$$\left\langle \underline{n}_t^A, \underline{n}_{t'}^B \right\rangle = \Sigma^{A,B}\delta(t, t') \tag{69.181}$$

Suppose $\{\underline{\vec{y}}_t\}_{\forall t}$ is a zero mean vector t-series. Hence $\underline{y}_t^A = \underline{Y}_t^A - \mu^A$, $E[\underline{Y}_t^A] = \mu^A$, $E[\underline{y}_t^A] = 0$. Suppose also that $\{\underline{\vec{n}}_t\}_{\forall t} \sim WN(0, \Sigma)$. Then we define a **Vector Auto-Regressive t-series** $VAR(p)$ by

$$\underline{y}_t^A = \sum_{j=1}^{p} \alpha_j^{A,B} \underline{y}_{t-j}^B + \underline{n}_t^A \tag{69.182}$$

The bnet for $VAR(p)$ is the same as the bnet for $AR(p)$ except that now, for all t, the nodes $\underline{y}_t$ and $\underline{n}_t$ are replaced by $\underline{\vec{y}}_t$ and $\underline{\vec{n}}_t$ and now the node weights α_t are $nr \times nr$ matrices with entries $\alpha_t^{A,B}$. For example, for $nr = 2$, we have

$$\begin{bmatrix} \underline{y}_t^1 \\ \underline{y}_t^2 \end{bmatrix} = \sum_{j=1}^{p} \begin{bmatrix} \alpha_j^{1,1} & \alpha_j^{1,2} \\ \alpha_j^{2,1} & \alpha_j^{2,2} \end{bmatrix} \begin{bmatrix} \underline{y}_{t-j}^1 \\ \underline{y}_{t-j}^2 \end{bmatrix} + \begin{bmatrix} \underline{n}_t^1 \\ \underline{n}_t^2 \end{bmatrix} \tag{69.183}$$

Chapter 70

Transfer Learning

Historically, the term Transfer Learning (TL) has been used in AI mostly by Artificial Neural Net (ANN) proponents (see the Wikipedia article on TL, Ref.[125]). Recently, however, a theory of causal TL has begun to emerge (see Chapter 72).

TL in AI is a fairly wide topic that is somewhat ambiguously defined. Some subjects that can be lumped under the heading of TL in AI are:

- data fusion/combining models
- model generalization
- transportability of causal knowledge, external validity (see Chapter 72)

Most AI researchers will agree that it is highly desirable to have TL in AI, because the human brain obviously does plenty of TL to great advantage. Although reams of papers have been written about the subject of TL in AI, a systematic theory of TL in AI that is universally accepted and popular remains elusive. The current theory of TL for ANN looks to me like a grab bag of heuristic approaches that are fragile, meaning they can be easily spoofed. The theory of TL for bnets (see Chapter 72) seems to me to be in much better shape: it's more elaborate, systematic, and it yields more robust results.

In this brief chapter, I will limit myself to describing a possible way of classifying, from a bnet perspective, the various approaches to TL in AI. Note that this method of classification even works for TL for ANNs, because ANNs can be viewed as bnets with deterministic nodes and a layered structure. One can describe TL in AI as a systematic way of defining a bnet $\mathcal{B}^*$ using a bnet $\mathcal{B}$ and other information. Bnet $\mathcal{B}$ is associated with a dataset $\mathcal{D}$, and bnet $\mathcal{B}^*$ is associated with a dataset $\mathcal{D}^*$. A bnet $\mathcal{B} = (\mathcal{S}, \theta)$ comprises a DAG structure $\mathcal{S}$ and a TPM for each node of the DAG. We'll denote the TPMs (aka parameters) of $\mathcal{B}$ by θ. So let's classify the various approaches to TL in AI by specifying what parts of the structure $\mathcal{S}$ and parameters θ of $\mathcal{B}$ are transferred to the structure $\mathcal{S}^*$ and parameters θ^* of $\mathcal{B}^*$, and what parts of $\mathcal{B}^* = (\mathcal{S}^*, \theta^*)$ are new.

1. **Fine tune parameters**. $\mathcal{S}^* = \mathcal{S}$, $\theta^* \approx \theta$.

   In this approach, we use the dataset $\mathcal{D}^*$ associated with bnet $\mathcal{B}^*$ to adjust slightly the parameters from θ to θ^*.

2. **Replace final layers of $\mathcal{S}$ and of θ by new ones**. $\mathcal{S}^* = \mathcal{S}$ and $\theta^* = \theta$ except for final layers,

   For example, in an ANN, replace the final layers by new ones, and use $\mathcal{D}^*$ to find the parameters of those new final layers.

3. **Transfer only the TPM of a single node $\underline{y}$ of $\mathcal{B}$**. $\mathcal{S}^*$ and θ^* are new except $P^*(y|pa(y)) = P(y|pa(y))$.

Chapter 71

Transformer Networks

This chapter in based on Refs.[18] and [126].

Transformer Networks (TN) have been taking the field of Natural Language Processing (NLP) by storm in recent years. They were introduced in 2017 and already are the basis of BERT (Bidirectional Encoder Representations from Transformers) and GPT (Generative Pre-trained Transformer), two TN libraries that have been trained with huge databases such as all of English Wikipedia (2,500M words).

Recurrent Neural Nets (RNNs) are discussed in Chapter 55. TNs are quickly displacing RNNs, an older method, in NLP. TNs are better than RNNs for doing NLP in several important ways. Whereas RNNs analyze the tokens (words) of a sentence sequentially (like a Kalman Filter), TNs analyze them in parallel, and thus are more amenable to parallel computing. Also, because RNNs analyze the words of a sentence sequentially, they tend to give more importance to the end of a sentence than to its beginning. That's because RNNs start forgetting the beginning of a sentence by the time they reach its end, like a patient with Alzheimer's. TNs do not suffer from this malady.

Dynamic bnets are discussed in Chapter 20. In Chapter 55, we showed that RNNs are dynamic bnets. The goal of this chapter is to define TNs, and to show that they too are dynamic bnets.

Let

$\mathcal{S}$ be the set of words in a sentence,

$h_i^t \in \mathbb{R}^{nh}$ be an nh dimensional column vector for word $i \in \mathcal{S}$ at time t.

$Q^t, K^t, V^t \in \mathbb{R}^{nh \times nh}$ be the **prior-attention matrices for time slice** t. These matrices are learned by training the net. The letters Q, K, V stand for Query, Key and Value, respectively.

Fig.71.1 represents a TN of a 3-word sentence as a dynamic bnet. The TPMs, printed in blue, for the nodes of the bnet Fig.71.1, are as follows:

$$P(V^t) = \delta(V^t, V_0^t) \tag{71.1a}$$

Figure 71.1: Time slice t of dynamic bnet for a transformer network (TN) of a 3-word sentence. The following arrows were not drawn explicitly for clarity: arrows pointing from node $\underline{V}^t$ to nodes $\underline{v}_i^t$ for $i = 0, 1, 2$, from node $\underline{Q}^t$ to nodes $\underline{q}_i^t$ for $i = 0, 1, 2$, and from node $\underline{K}^t$ to nodes $\underline{k}_i^t$ for $i = 0, 1, 2$. Note that k_i^t for all i points to $\underline{h}_j^{t+1}$ for all j. Likewise, $\underline{v}_i^t$ for all i points to $\underline{h}_j^{t+1}$ for all j.

$$P(Q^t) = \delta(Q^t, Q_0^t) \tag{71.1b}$$

$$P(K^t) = \delta(K^t, K_0^t) \tag{71.1c}$$

$$P(v_i^t | V^t, h_i^t) = \mathbb{1}(\quad v^t = V^t h_i^t \quad) \tag{71.2}$$

$$P(q_i^t | Q^t, h_i^t) = \mathbb{1}(\quad q^t = Q^t h_i^t \quad) \tag{71.3}$$

$$P(k_i^t | K^t, h_i^t) = \mathbb{1}(\quad k^t = K^t h_i^t \quad) \tag{71.4}$$

$$P(h_i^{t+1} | v_\cdot^t, q_i^t, k_\cdot^t) = \mathbb{1}(\quad h_i^{t+1} = \underbrace{\sum_{j \in \mathcal{S}} v_j^t \quad \underbrace{\frac{e^{(q_i^t)^T k_j^t}}{\sum_{j' \in \mathcal{S}} e^{(q_i^t)^T k_{j'}^t}}}_{w_{j|i} = [\mathrm{softmax}((q_i^t)^T k_j^t)]_j}}_{E_{\underline{j}|i}[v_{\underline{j}}^t] = \textbf{Attention}} \quad) \tag{71.5}$$

In Eqs.71.1, the 3 root nodes $\underline{V}^t, \underline{Q}^t, \underline{K}^t$ have delta function priors. To improve stability of the dynamic bnet, it is more common instead to use for these priors, multiple delta functions (aka **multi-heads**) labeled $c = 0, 1, \ldots, nc - 1$, as follows.

$$P(V^t | c) = \delta(V^t, V_c^t) \tag{71.6a}$$

$$P(Q^t | c) = \delta(Q^t, Q_c^t) \tag{71.6b}$$

$$P(K^t | c) = \delta(K^t, K_c^t) \tag{71.6c}$$

for $c = 0, 1, \ldots, nc-1$. Even better, one can go fully Bayesian, and make these 3 prior distributions arbitrary categorical distributions to be determined by net training, instead of mere delta functions.

Chapter 72

Transportability of Causal Knowledge

This chapter is mostly based on Refs.[39] and [27].

Figure 72.1: Example of selection bnet $G_{\underline{s}}$ created from bnet G.

Suppose one wants to transfer causal knowledge from a **source population** Σ to a **target population** Σ^*.

Given a bnet G, define a **selection diagram** $G_{\underline{s}}$ as a bnet formed by adding to G a new root node $\underline{s}$ and new arrows pointing from **selection node** $\underline{s}$ to one or more **target nodes** of G. We'll call the set of target nodes of $\underline{s}$ the **target set** $T_{\underline{s}}$. $\underline{s} = 0$ corresponds to population Σ and $\underline{s} = 1$ to population Σ^*. For bnet G, the TPM for a node $\underline{x}$ with parents $pa(\underline{x})$, is given by:

$$P_G(x|pa(x)) = P(x|pa(x)) \tag{72.1}$$

For bnet $G_{\underline{s}}$, nodes $\underline{x}$ with parents $pa(\underline{x})$, where $\underline{s} \notin pa(\underline{x})$, have TPMs:

$$P_{G_{\underline{s}}}(x|pa(x)) = P(x|pa(x)) \ . \tag{72.2}$$

Nodes $\underline{x}$ with parents $pa(\underline{x}) \cup \underline{s}$, have TPMs:

$$P_{G_{\underline{s}}}(x|pa(x), s) = \begin{cases} P(x|pa(x), \underline{s} = 0) = P(x|pa(x)) & \text{if } \underline{s} = 0 \\ P(x|pa(x), \underline{s} = 1) = P^*(x|pa(x)) & \text{if } \underline{s} = 1 \end{cases} \tag{72.3}$$

Fig.72.1 shows an example of a selection diagram $G_{\underline{s}}$. In that figure, the target set of $\underline{s}$ is $\{\underline{x}, \underline{z}\}$.

All this can be generalized so as to have more than one selection node, with the target sets of the selection nodes being disjoint, and such that a selection node can have more than 2 states.

Observational data is data that determines a probability distribution without do operators in it. **Experimental data** determines a probability distribution with do's in it.

Do-transport formulae

Claim 76 *(Trivial Memoryless Transportability, from Ref.[39])*
If $\underline{s}$ *where* $\underline{s} \in \{0,1\}$ *is a selection node, then*

$$\begin{tikzcd} & \underline{z} \arrow[dl] \arrow[dr] \arrow[dd, dashed, bend right] & \underline{s} \arrow[dd] \\ \underline{x} \arrow[rr] & & \underline{y} \arrow[ll, dashed, bend left] \end{tikzcd}$$

$$P^*(y|\mathcal{D}\underline{x} = x, z) = P^*(y|x, z) \quad \textit{(replace } \mathcal{D} \textit{ by } 1\textit{, keep } P^*\textit{)} \tag{72.4}$$

$$\begin{tikzcd} & z \arrow[dr] & \underline{s} = 1 \arrow[d] \\ \mathcal{D}\underline{x} = x \arrow[rr] & & y \end{tikzcd} = \begin{tikzcd} & z \arrow[dr] & \underline{s} = 1 \arrow[d] \\ x \arrow[rr] & & y \end{tikzcd} \tag{72.5}$$

proof: See Claim 25.
QED

Claim 77 *(Direct Transportability, a.k.a. External Validity, from Ref.[39])*
If $\underline{s}$ *where* $\underline{s} \in \{0,1\}$ *is a selection node, then*

$$\begin{tikzcd} \underline{s} \arrow[r] & \underline{z} \arrow[dl] \arrow[dr] \arrow[dd, dashed, bend right] & \\ \underline{x} \arrow[rr] & & \underline{y} \arrow[ll, dashed, bend left] \end{tikzcd}$$

$$P^*(y|\mathcal{D}\underline{x} = x, z) = P(y|\mathcal{D}\underline{x} = x, z) \quad \textit{(replace } P^* \textit{ by } P\textit{, keep } \mathcal{D}\textit{)} \tag{72.6}$$

$$\begin{tikzcd} \underline{s} = 1 \arrow[r] & z \arrow[dr] & \\ \mathcal{D}\underline{x} = x \arrow[rr] & & y \end{tikzcd} = \begin{tikzcd} & z \arrow[dr] & \\ \mathcal{D}\underline{x} = x \arrow[rr] & & y \end{tikzcd} \tag{72.7}$$

Furthermore,

$$P^*(y|\mathcal{D}\underline{x} = x) = \sum_z P(y|\mathcal{D}\underline{x} = x, z) P^*(z) \tag{72.8}$$

$$\begin{tikzcd} \underline{s} = 1 \arrow[dr] & \\ \mathcal{D}\underline{x} = x \arrow[r] & y \end{tikzcd} = \begin{tikzcd} \underline{s} = 1 \arrow[r] & \sum z \arrow[dr] & \\ \mathcal{D}\underline{x} = x \arrow[rr] & & y \end{tikzcd} \tag{72.9}$$

proof: See Claim 26.
QED

Claim 78 *(S-Admisssible Transportability, from Ref.[39])*

If $\underline{s}$ *where* $\underline{s} \in \{0,1\}$ *is a selection node, then*

$$P^*(y|\mathcal{D}\underline{x} = x) = \sum_a P(y|\mathcal{D}\underline{x} = x, a)P^*(a) \tag{72.10}$$

$$\begin{tikzcd} \underline{s} = 1 \arrow[dr] & \\ \mathcal{D}\underline{x} = x \arrow[r] & y \end{tikzcd} = \begin{tikzcd} \underline{s} = 1 \arrow[r] & \sum a \arrow[d] \\ \mathcal{D}\underline{x} = x \arrow[r] & y \end{tikzcd} \tag{72.11}$$

proof: See Claim 27.
QED

Claim 79 *(Non-transportability, from Ref.[39])*

If $\underline{h}$ $\underline{s}$ *where* $\underline{s} \in \{0,1\}$ *is a selection node, then*

$$P^*(y|\mathcal{D}\underline{x} = x) = P^*(y|\mathcal{D}\underline{x} = x) \tag{72.12}$$

$$\begin{tikzcd} & \underline{s} = 1 \arrow[d] \\ \mathcal{D}\underline{x} = x \arrow[r] & y \end{tikzcd} = same \tag{72.13}$$

proof: See Claim 28.
QED

Claim 80 *(from Ref.[39])*

If $\underline{s}$ $\underline{h}$ $\underline{z}$ *where* $\underline{s} \in \{0,1\}$ *is a selection node, then*

$$P^*(y|\mathcal{D}\underline{x} = x) = P(y|\mathcal{D}\underline{x} = x) \tag{72.14}$$

$$\begin{tikzcd} \underline{s} = 1 \arrow[dr] & & \\ \mathcal{D}\underline{x} = x \arrow[r] & y \end{tikzcd} \quad = \quad \begin{tikzcd} \mathcal{D}\underline{x} = x \arrow[r] & y \end{tikzcd} \tag{72.15}$$

proof: See Claim 29.
QED

Claim 81 *(from Ref.[39])*

If $\begin{tikzcd} \underline{s} \arrow[r] \arrow[dr] \arrow[ddr] & \underline{h} \arrow[dl] \arrow[d] \arrow[dd, bend right] \\ \underline{x} \arrow[r] & \underline{z} \arrow[r] & \underline{y} \arrow[l, bend left] \end{tikzcd}$ *where* $\underline{s} \in \{0,1\}$ *is a selection node, then*

$$P^*(y|\mathcal{D}\underline{x} = x) = \sum_z P(y|\mathcal{D}\underline{x} = x, z) P^*(z|x) \tag{72.16}$$

$$\begin{tikzcd} \underline{s} = 1 \arrow[dr] & \mathcal{D}\underline{x} = x \arrow[d] \\ & y \end{tikzcd} \quad = \quad \begin{tikzcd} \underline{s} = 1 \arrow[dr] & & \mathcal{D}\underline{x} = x \arrow[d] \\ x \arrow[r] & \sum z \arrow[r] & y \end{tikzcd} \tag{72.17}$$

proof: See Claim 30.
QED

Chapter 73

Turbo Codes

This chapter is based on Ref.[26].

In this chapter, vectors with n components will be indicated by an n superscript. For example, $a^n = (a_0, a_1, \ldots, a_{n-1})$.

Consider an n-letter message $u^n = (u_0, u_1, \ldots, u_{n-1})$, where for all i, $u_i \in \mathcal{A}$ is an element of an alphabet $\mathcal{A}$, and where for all i, the $\underline{u}_i$ are i.i.d.. Suppose u^n is encoded deterministically in two different ways, $e_1(u^n)$ and $e_2(u^n)$. After passing through the same memoryless channel, the variables u^n, e_1, e_2 become $\tilde{u}^n, \tilde{e}_1, \tilde{e}_2$, respectively. The letter u stands for unencoded, and e for encoded. Quantities with a tilde $\tilde{u}^n, \tilde{e}_1, \tilde{e}_2$ occur after channel passage and are visible (measurable). Quantities without a tilde u^n, e_1, e_2 are hidden (unmeasurable).

The situation just described can be represented by the bnet Fig.73.1, or by its abridged version Fig.73.2. But note that the abridged version does not show explicitly that the u_i are i.i.d. or that the channel is memoryless (i.e., that the u_i for all i pass independently through the channel).

Define

$$x = (u^n, e_1, e_2) \tag{73.1}$$

and

$$\tilde{x} = (\tilde{u}^n, \tilde{e}_1, \tilde{e}_2) \ . \tag{73.2}$$

Fig.73.1 implies that

$$P(x, \tilde{x}) = P(\tilde{u}^n | u^n) \left[\prod_{r=1,2} P(\tilde{e}_r | e_r) P(e_r | u^n) \right] P(u^n) \ . \tag{73.3}$$

Because the u^n are i.i.d.,

$$P(u^n) = \prod_i P(u_i) \ . \tag{73.4}$$

Figure 73.1: Turbo coding Bnet representing a message being encoded two different ways and then the original message and the 2 encodings pass through a memoryless channel.

Figure 73.2: Abridged version of Fig.73.1.

Because the channel is memoryless,

$$P(\tilde{u}^n|u^n) = \prod_i P(\tilde{u}_i|u_i) \ . \tag{73.5}$$

Because the encoding is deterministic, we must have for $r = 1, 2$

$$P(e_r|u^n) = \delta(e_r, e_r(u^n)) \ . \tag{73.6}$$

Define the belief functions

$$BEL_i = BEL_i(\underline{u}_i = a) = P(\underline{u}_i = a|\tilde{x}) \ . \tag{73.7}$$

The best estimate of u_j given all visible evidence $\tilde{x}$ is

$$\hat{u}_i = \underset{u_i}{\operatorname{argmax}}\, BEL_i(u_i) \ . \tag{73.8}$$

Define the probability functions

$$\pi_i = \pi_i(u_i) = P(u_i) \ , \tag{73.9}$$

and the likelihood functions

$$\lambda_i = \lambda_i(u_i) = P(\tilde{u}_i|u_i) \ . \tag{73.10}$$

For $r = 1, 2$, define the Kernel functions

$$K_r = K_r(u^n) = P(\tilde{e}_r|e_r = e_r(u^n)) \ . \tag{73.11}$$

In this book, $\mathcal{N}(!a)$ denotes a normalization constant that does not depend on a. Define

$$\mathcal{N}_i = \mathcal{N}(!u_i) \ . \tag{73.12}$$

Claim 82

$$BEL_i = \mathcal{N}_i \lambda_i \pi_i \mathcal{T}_i^{K_1 K_2}[\prod_{j \neq i} \lambda_j \pi_j] \ , \tag{73.13}$$

where $\mathcal{T}_i^K(\cdot)$ with $K = K_1 K_2$ is an operator (transform) that acts on functions of u^n:

$$\mathcal{T}_i^K(\cdot) = \sum_{u^n} \delta(u_i, a) K(u^n)(\cdot) \ . \tag{73.14}$$

proof:

$$\begin{aligned}
P(\underline{u}_i = a|\tilde{x}) = & \\
= & \sum_x \delta(u_i, a) P(x|\tilde{x}) && (73.15)\\
= & \sum_x \delta(u_i, a) \frac{P(\tilde{x}|x)P(x)}{P(\tilde{x})} && (73.16)\\
= & \mathcal{N}(!a) \sum_x \delta(u_i, a) P(\tilde{x}|x) P(x) && (73.17)\\
= & \mathcal{N}(!a) \sum_x \delta(u_i, a) P(u^n) \left[\prod_{r=1,2} P(\tilde{e}_r|e_r) \delta(e_r, e_r(u^n)) \right] \prod_j P(\tilde{u}_j|u_j) && (73.18)\\
= & \mathcal{N}(!a) \lambda_i(a) \pi_i(a) R \ , && (73.19)
\end{aligned}$$

where

$$
\begin{aligned}
R &= \sum_{u^n} \delta(u_i, a) \left[\prod_{r=1,2} P(\tilde{e}_r | e_r(u^n)) \right] \prod_{j \neq i} P(\tilde{u}_j | u_j) P(u_j) && (73.20) \\
&= \sum_{u^n} \delta(u_i, a) \left[\prod_{r=1,2} K_r(u^n) \right] \prod_{j \neq i} \lambda_j(u_j) \pi_j(u_j) && (73.21) \\
&= \mathcal{T}_i^{K_1 K_2} [\prod_{j \neq i} \lambda_j(u_j) \pi_j(u_j)] \ . && (73.22)
\end{aligned}
$$

Hence

$$
BEL_i(a) = \mathcal{N}(!a) \lambda_i(a) \pi_i(a) \mathcal{T}_i^{K_1 K_2} [\prod_{j \neq i} \lambda_j(u_j) \pi_j(u_j)] \ . \tag{73.23}
$$

QED

73.1 Decoding Algorithm

The Turbo algorithm for decoding the encode message is as follows. For $m = 0$, let

$$
\pi_j^{(0)}(u_j) = \frac{1}{n_{\underline{u}_j}} \ . \tag{73.24}
$$

Then for $m = 1, 2, \ldots,$ let

$$
\pi_i^{(m)} = \mathcal{N}_i \mathcal{T}_i^{K_{m\%2}} [\prod_{j \neq i} \lambda_j \pi_j^{(m-1)}] \ , \tag{73.25}
$$

where $m\%2 = 1$ if m is odd and $m\%2 = 2$ if m is even. Furthermore, for $m > 0$, let

$$
\begin{aligned}
BEL_i^{(m)} &= \mathcal{N}_i \lambda_i \pi_i^{(m-1)} \pi_i^{(m)} && (73.26) \\
&= \mathcal{N}_i \lambda_i \pi_i^{(m-1)} \mathcal{T}_i^{K_{m\%2}} [\prod_{j \neq i} \lambda_j \pi_j^{(m-1)}] \ . && (73.27)
\end{aligned}
$$

As $m \to \infty$, $BEL_i^{(m)}$ given by Eq.(73.27) is expected to converge to the the exact BEL_i given by Eq.(73.13).

Turbo decoding can be represented by the bnets Figs.73.3 and 73.4.

The node TPMs, printed in blue, for Fig.73.3, are given by:

$$
P(d_i^{(m)} = a \mid \tilde{u}^n, \tilde{e}_{m\%2}) = BEL_i^{(m)}(a) \ . \tag{73.28}
$$

Figure 73.3: Bnet describing Turbo code generation of $BEL_i^{(m)}(a)$ for $m = 1, 2, \ldots$.

Figure 73.4: Bnet describing Turbo code generation of $BEL^{n(m)}(\cdot)$ and $\pi^{n(m)}(\cdot)$ for $m = 0, 1, 2 \ldots$. The following arrows were not drawn for clarity: Arrows pointing from node $\underline{\lambda}^n(\cdot)$ to nodes $\underline{\pi}^{n(m)}(\cdot)$ and $\underline{BEL}^{n(m)}(\cdot)$ for $m = 0, 1, 2, \ldots$.

The TPMs, printed in blue, for Fig.73.4, are given by:

$$P((\lambda^n)'(\cdot)|\tilde{u}^n) = \delta((\lambda^n)'(\cdot), \lambda^n(\cdot)) \tag{73.29}$$

$$P(\pi^{n(m)}(\cdot)|\lambda^n(\cdot), \pi^{n(m-1)}(\cdot), \tilde{e}_{m\%2}) = \prod_i \prod_{u_i} \delta(\pi_i^{(m)}(u_i), \mathcal{N}_i \mathcal{T}_i^{K_{m\%2}}[\prod_{j \neq i} \lambda_j \pi_j^{(m-1)}]) \tag{73.30}$$

$$P(BEl^{n(m)}(\cdot)|\lambda^n(\cdot), \pi^{n(m)}(\cdot), \pi^{n(m-1)}(\cdot)) = \prod_i \prod_{u_i} \delta(BEL_i(u_i), \mathcal{N}_i \lambda_i \pi_i^{(m-1)} \pi_i^{(m)}) \tag{73.31}$$

73.2 Message Passing Interpretation of Decoding Algorithm

Ref.[26] shows that the Turbo code decoding algo can be interpreted as an application of Message Passing. We leave all talk of Message Passing to a separate chapter, Chapter 41.

Chapter 74

Uplift Modelling

This chapter is based on many references, including Ref.[13, 9, 127, 44].

Uphill Modelling (UP) deals with the application of Rubin's Theory of Potential Outcomes (PO) to advertisement and marketing.

PO, which is discussed in Chapter 52, is a subset of Pearl's Causal Inference. Besides UP, other applications of PO theory that are discussed in this book are: Regression Discontinuity (Chapter 56), Difference-in-Differences (Chapter 13) and Synthetic Controls (Chapter 68).

In UP, each **participant person** is interrogated at two well anticipated, fairly closely spaced times t_0 and t_1 (as opposed to Difference-in-Differences (DID), where t_0 and t_1 might be years apart, and long before the DID analysis is attempted.). In between those two times, a treatment which we will refer to as the **UP diagnostic test** is applied. For example, at times t_0 and t_1, every participant might be asked how important he/she rates climate change on a scale of 1 to 10. In between times t_0 and t_1, every participant might be sent a brochure on climate change. In UP, as in all other PO applications, each **sample** σ is in the treated or control groups, but not both. But in UP, the same participant can be in both the treated and control groups. If so, that participant is considered two different samples σ; for example, $\sigma = treatedBob, controlBob$. In UP, the samples are aware of which of those groups they are in, so they are not "treatment blind".

74.1 UP types

Let $y_t^B \in \mathbb{R}$ for $t = t_0, t_1$ be the treatment response at time t for participant B. (We are using here the same notation as in Chapter 52). Call $\delta^B = y_{t_1}^B - y_{t_0}^B$ the **participant uplift** for participant B. As shown in Fig.74.1, UP classifies participants into 4 **UP-types**: Persuadables, SureThings, LostCauses, and SleepyDogs. The UP-type of a participant depends on the changes that are induced on that participant by an **UP-diagnostic-test**.

- For a **Persuadable** participant, $\delta^B > 0$.

Figure 74.1: UP diagnostic test can be used to classify all participants of the population into 4 UP-types. This figure assumes $y \in \{0, 1\}$. More generally, $y \in \mathbb{R}$. t represents time. $t = t_0$ corresponds to $d = 0 = untreated$, and $t = t_1$ corresponds to $d = 1 = treated$.

- For a **SleepyDogs** participant, $\delta^B < 0$.
- For a **SureThings** participant, $\delta^B \approx 0$ and $y^B_{t_0}$ is high.
- For a **LostCauses** participant, $\delta^B \approx 0$ and $y^B_{t_0}$ is low.

Suppose B belongs to stratum A_x. What is commonly called the **uplift** is the **stratum-uplift** $\delta_x = ACE_x$. Strata can also be classified into the 4 UP-types, depending on the sign and size of their δ_x. A participant may not be typical for his stratum and may have different **participant and stratum UP-types**. For example, he may have positive participant uplift and therefore have a Persuadable participant UP-type, but his stratum-uplift might be negative, so he has the SleepyDogs stratum UP-type.

Advertisers are very interested in finding the Persuadable strata in a population so as to focus their resources on them. For example, UP was used very successfully during the Obama presidential campaigns. Team Obama conducted UP-diagnostic tests much like the climate change one described earlier. This allowed them to identify voters who might be sitting on the fence on whether to vote for Obama or not. Then Team Obama spent the lion share of resources on those fence-sitters.

74.2 Some Relevant Technical Formulas from Chapter 52

Recall the following technical formulae that were proven in Chapter 52:

- Recall Eq.(52.143):

$$ACE = \sum_x P(x) \underbrace{\sum_y y \left[P(y|d=1,x) - P(y|d=0,x)\right]}_{ACE_x} \tag{74.1}$$

If $\underline{y} \in \{0,1\}$, then

$$\underbrace{ACE_x}_{\delta_x} = \underbrace{P_{\underline{y}|\underline{d},\underline{x}}(1|1,x)}_{Y_x^1} - \underbrace{P_{\underline{y}|\underline{d},\underline{x}}(1|0,x)}_{Y_x^0} \ . \tag{74.2}$$

- Recall Eq.(52.158):

$$\underbrace{\widehat{ACE_x}}_{\delta_x} = \underbrace{\frac{1}{N_x} \sum_{\sigma \in A_x} \frac{d^\sigma y^\sigma}{g_{1|x^\sigma}}}_{Y_x^1} - \underbrace{\frac{1}{N_x} \sum_{\sigma \in A_x} \frac{(1-d^\sigma) y^\sigma}{g_{0|x^\sigma}}}_{Y_x^0} \tag{74.3}$$

74.3 UP Analysis

The input to UP is a PO dataset $DS = \{(\sigma, d^\sigma, x^\sigma, y^\sigma) : \sigma = 0,1,2,\ldots,nsam-1\}$. where $d^\sigma \in \{0,1\}$, $x^\sigma \in S_{\underline{x}}$, $y^\sigma \in \mathbb{R}$. A participant B is assigned two different σ if he/she belongs to both the treated and control groups. We will assume $S_{\underline{x}}$ is a finite set. In general, $x = (x_0, x_1, \ldots, x_{n-1})$ is an n dimensional vector of features x_i. If any of the x_i is a priori continuous, we will assume it has been binned into a finite number of bins.

Starting with DS, UP performs the following steps. Fig.74.2 is a pictorial representation of the quantities that are calculated during these steps.

1. Find A_x for each observed $x \in S_{\underline{x}}$. Set $A_x = \emptyset$ for unobserved $x \in S_{\underline{x}}$.
2. Calculate δ_x for each $x \in S_{\underline{x}}$. Set $\delta_x = 0$ if $A_x = \emptyset$.
3. Calculate the set

$$\{\Delta_c\}_{c=0,1,\ldots,nc-1} = \{\delta_x : x \in S_{\underline{x}}\} \tag{74.4}$$

of distinct uplifts δ_x. The class labels c should be assigned so that the sequence of Δ_c is monotonic and non-increasing; i.e.,

$$\Delta_0 \geq \Delta_1 \geq \cdots \geq \Delta_{nc-1} \ . \tag{74.5}$$

Figure 74.2: Pictorial representation of the sequence $\{(\mathcal{X}_c, \Delta_c)\}_{c=0,1,\ldots,nc-1}$.

Now calculate

$$\mathcal{X}_c = \{x : \delta_x = \Delta_c\} \tag{74.6}$$

for each c. By the end of this step, we will have calculated $\{(\mathcal{X}_c, \Delta_c)\}_{c=0,1,\ldots,nc-1}$. We will refer to the $\mathcal{X}_c$ as **strata-bins**. Note that

$$\Delta_c = \frac{1}{|\mathcal{X}_c|} \sum_{x \in \mathcal{X}_c} \delta_x \tag{74.7}$$

$$= \underbrace{\frac{1}{|\mathcal{X}_c|} \sum_{x \in \mathcal{X}_c} Y_x^1}_{Y_c^1} - \underbrace{\frac{1}{|\mathcal{X}_c|} \sum_{x \in \mathcal{X}_c} Y_x^0}_{Y_c^0} \ . \tag{74.8}$$

4. For each c, calculate

$$\Sigma_{d,c} = \cup_{x \in \mathcal{X}_c} A_{d,x} \tag{74.9}$$

for $d \in \{0, 1\}$ and

$$\Sigma_c = \Sigma_{0,c} \cup \Sigma_{1,j} \ . \tag{74.10}$$

Fig.74.3 is a way of plotting the results of UP in an intuitive way that even a business type can understand. UP software often plots something called a Qini curve, but I find Qini curves opaque, confusingly defined in the literature, unnecessary and not very well motivated. So I don't use them.

Figure 74.3: Plot of UP results. Alternative to Qini curves.

74.4 UP Decision Trees

In this section, we will describe how to build UP decision trees (UP dtrees), and explain why they are needed for UP.

Generic dtrees are described in Chapter 12. This section complements rather than replaces that chapter so the reader is advised to read that chapter first.

Ref.[44] is an excellent paper on the use of dtrees in UP.

The analysis described previously in Section 74.3, although theoretically correct, will work very poorly in practice. The strata-bins of Section 74.3 correspond to the classification classes of a dtree. But strata-bins are very specific so they severely overfit the data. Although dtrees can also suffer from overfitting, there are known methods of preventing or mitigating overfitting in dtrees.

There are also tasks that dtrees can do well and the methods explained so far cannot do well. For example, suppose we have a classless dataset $DS^- = \{(\sigma, x^\sigma) : \sigma \in \Sigma^-\}$ and we want to predict the class c^σ and uplift Δ_{c^σ} for each of these individuals $\sigma \in \Sigma^-$. A dtree can easily do that. The alternative is to use the classy dataset $DS = \{(\sigma, x^\sigma, c^\sigma) : \sigma \in \Sigma\}$ to prepare a dictionary that orders the elements of $S_{\underline{x}}$ and gives a class c and an uplift value Δ_c for each feature vector $x \in S_{\underline{x}}$. But such a dictionary overfits and says nothing for feature vectors x that do not show up in the classy dataset DS; i.e., the dictionary doesn't guess (interpolate). Dtrees, on the other hand, do guess.

So, without further ado, let us describe how to modify the results of Chapter 12 on generic dtrees to the case of UP dtrees. The main difference, as we will explain

in detail next, is that the Information Gain metric used for generic dtrees needs to be replaced by another metric.

$$\{N_j^d(c, x_j)\}_{c \in S_{\underline{c}}, x_j \in S_{\underline{x}_j}}$$

$$\sum_{c \in S_{\underline{c}}} N_j^d(c, x_j) = N_j^d(x_j)$$

$$\sum_{x_j \in S_{\underline{x}_j}} N_j^d(c, x_j) = N_j^d(c) \qquad \sum_{c \in S_{\underline{c}}} N_j^d(c) = \sum_{x_j \in S_{\underline{x}_j}} N_j^d(x_j) = N_j^d$$

Figure 74.4: Fig.12.4 with d dependence added. $d \in \{0, 1\}$ is the treatment dose.

Fig.74.4 was obtained from Fig.12.4 by adding d dependence. $d \in \{0, 1\}$ is the treatment dose. Note that in UP, we build a dtree in which every node carries a double ($d = 0, 1$) TPV. This is in contrast to the generic dtrees built in Chapter 12, in which each node carries a single TPV. $N_j^d(c, x_j)$ is the number of individuals σ in the population that reaches node $\underline{x}_j$ with $d \in \{0, 1\}$, belonging to class $c \in S_{\underline{c}}$ and having $\underline{x}_j = x_j$. From these population numbers, we can define the bnet in Fig.74.5. The TPMs, printed in blue, for the (non-root) nodes of this bnet, are as follows

$$P(c|x_j, j, d) = \frac{N_j^d(c, x_j)}{N_j^d(x_j)} \tag{74.11}$$

$$P(x_j|j, d) = \frac{N^d(x_j)}{N_j^d} \tag{74.12}$$

Figure 74.5: Bnet derived from population numbers in Fig.74.4

In Chapter 12, we used Information Gain (a mutual information) as the SAM (Separation Ability Measure) in SL (Structure Learning) of dtrees (Decision Trees). Information Gain is a bad SAM for SL of UP dtrees, because it knows nothing about $d = 0, 1$ and the double TPVs of nodes in UP dtrees. For UP dtrees, we need a SAM specifically designed to separate $d = 0, 1$, and generate classes that are uplift bins (i.e., uplift intervals).

Ref.[44] proposes and studies the following 3 SAMs for doing SL of UP dtrees.

1. **SAM_DD** (DD=Delta Delta)

   For $d \in \{0, 1\}$ and $c, c' \in S_{\underline{c}}$, define the increments

$$\partial_d f(d) = f(1) - f(0) \tag{74.13}$$

   and

$$\partial_{c',c} f(c) = f(c') - f(c) \ . \tag{74.14}$$

   Let

$$\begin{aligned} \Delta_{c|j} &= P(c|j,1) - P(c|j,0) && (74.15) \\ &= \partial_d P(c|j,d) && (74.16) \end{aligned}$$

$$\begin{aligned} SAM_DD_j &= \max_{c,c'} |\partial_{c',c} \partial_d P(c|j,d)| && (74.17) \\ &= \max_{c,c'} |\partial_{c',c} \Delta_{c|j}| && (74.18) \end{aligned}$$

2. **SAM_KL** (KL=Kullback Liebler)

$$SAM_KL_j = \left[\sum_{x_j \in S_{\underline{x}_j}} P(x_j|j) D_{KL}(P_{\underline{c}|x_j,j,1} \parallel P_{\underline{c}|x_j,j,0}) \right] - D_{KL}(P_{\underline{c}|j,1} \parallel P_{\underline{c}|j,0}) \tag{74.19}$$

$$= \left[\sum_{x_j \in S_{\underline{x}_j}} P(x_j|j) \sum_{c \in S_{\underline{c}}} P(c|x_j,j,1) \ln \frac{P(c|x_j,j,1)}{P(c|x_j,j,0)} \right] - \sum_{c \in S_{\underline{c}}} P(c|j,1) \ln \frac{P(c|j,1)}{P(c|j,0)} \tag{74.20}$$

   SAM_KL_j **can be negative.**

3. **SAM_E** (E=Euclidean)

   SAM_E_j is defined the same way as SAM_KL_j except with the KL divergence $D_{KL}(P \parallel Q)$ in SAM_KL replaced by the Euclidean distance squared.

$$D(P,Q) = \sum_x (P(x) - Q(x))^2 \tag{74.21}$$

The intuitive reason for using these quantities as SAMs is that they maximize the change in uplift between successive tree levels, so that the uplift increases as quickly as possible as we descend down the UP tree. In the case of generic dtrees for which we use Information Gain as SAM, we are maximizing the correlation between classes and nodes as we descend down the tree. These two goals are related. In fact, in the limit where the number of control individuals becomes zero, SAM_KL_j and IG_j become the same, as will be shown later.

Next we show that SAM_KL_j satisfies the following 3 axioms[1]

Claim 83 .

1. *SAM_KL_j is minimum iff $P(c|x_j, j, 0) = P(c|x_j, j, 1)$ for all c and x_j.*
2. *If $P(c|j, d) = P(c|d)$ for all c, d, then $SAM_KL_j = 0$.*
3. *Suppose $N_r^0 = 0$ for all nodes $r \in J_0$ (i.e., no control population) and we use the Laplace Correction when warranted. Then*

$$SAM_KL_j = H(\underline{c} : \underline{x}_j | j, 1) \tag{74.22}$$
$$= IG_j \quad \textit{for treated population} . \tag{74.23}$$

proof:

The proof of items 1 and 2 follow by inspection of Eq.(74.20). Item 3 is proven in Claim 84 below.

QED

Let $N_{\underline{c}} = |S_{\underline{c}}|$. Define the uniform probability distribution

$$U_{\underline{c}}(c) = \frac{1}{N_{\underline{c}}} \tag{74.24}$$

for all $c \in S_{\underline{c}}$.

[1] We won't show it here, but according to Ref.[44], SAM_E_j also satisfies these 3 axioms, but SAM_DD_j satisfies only the first two.

Eq.(74.11) for the TPM of node $\underline{c}$ in the bnet Fig.74.5 can be "Laplace Corrected" as follows so that it is no longer undefined when its denominator vanishes:

$$P(c|j,d) = \begin{cases} \frac{N_j^d(c)}{N_j^d} & \text{if } N_j^d > 0 \\ U_{\underline{c}}(c) & \text{if } N_j^d = 0 \text{ (Laplace Correction)} \end{cases} \tag{74.25}$$

Claim 84 *Suppose $N_r^0 = 0$ for all dtree nodes $r \in J_0$ and we use the Laplace Correction when warranted. Then*

$$SAM_KL_j = H(\underline{c} : \underline{x}_j|j,1) \ . \tag{74.26}$$

proof:

For all nodes $r \in J_0$, we must have

$$P_{\underline{c}|r,0} = U_{\underline{c}} \tag{74.27}$$

so

$$\begin{aligned} D_{KL}(P_{\underline{c}|r,1} \parallel P_{\underline{c}|r,0}) &= D_{KL}(P_{\underline{c}|r,1} \parallel U_{\underline{c}}) && (74.28) \\ &= \ln(N_{\underline{c}}) - H(\underline{c}|r,1) \ . && (74.29) \end{aligned}$$

For all $x_j \in S_{\underline{x}_j}$, we must also have

$$N_j = N_j^1, N(x_j) = N^1(x_j) \tag{74.30}$$

so

$$P(x_j|j) = P(x_j|j,1) \ . \tag{74.31}$$

Now using Eqs.(74.29) and (74.31), we get

$$\begin{aligned} SAM_KL_j &= -\left[\sum_{x_j \in S_{\underline{x}_j}} P(x_j|j) H(\underline{c}|x_j,j,1)\right] + H(\underline{c}|j,1) && (74.32) \\ &= -\left[\sum_{x_j \in S_{\underline{x}_j}} P(x_j|j,1) H(\underline{c}|x_j,j,1)\right] + H(\underline{c}|j,1) && (74.33) \\ &= -H(\underline{c}|\underline{x}_j,j,1) + H(\underline{c}|j,1) && (74.34) \\ &= H(\underline{c} : \underline{x}_j|j,1) && (74.35) \end{aligned}$$

QED

74.4.1 Appendix, connection between Δ_c and $\Delta_{c|j}$

Recall Eq.(74.8):

$$\Delta_c = \underbrace{\frac{1}{|\mathcal{X}_c|}\sum_{x\in\mathcal{X}_c} Y_x^1}_{Y_c^1} - \underbrace{\frac{1}{|\mathcal{X}_c|}\sum_{x\in\mathcal{X}_c} Y_x^0}_{Y_c^0} \tag{74.36}$$

$$= \partial_d Y_c^d \,. \tag{74.37}$$

Compare that to Eq.(74.16):

$$\Delta_{c|j} = P(c|j,1) - P(c|j,0) \tag{74.38}$$

$$= \partial_d P(c|j,d) \tag{74.39}$$

What is the connection between these 2 deltas, Δ_c and $\Delta_{c|j}$? Are they equal?

First off, notice that $\Delta_{c|j}$ is defined for all nodes j of the dtree. Let $j(c)$ be the leaf node for which $\Delta_c \approx \Delta_{c|j(c)}$. Assume $y^\sigma \in \{0,1\}$. Then

$$P(c|j=j(c),d) = \frac{N_{j(c)}^d(c)}{N_{j(c)}^d} \approx Y_c^d \tag{74.40}$$

So the two deltas are indeed approximately equal when $y^\sigma \in \{0,1\}$ and $j = j(c)$.

Chapter 75

Variational Bayesian Approximation

For more info and references about this topic, see Ref.[128].

The Variational Bayesian approximation (VBA) is an analytic (as opposed to numerical) approximation to the probability distribution $P(h|\vec{x})$, where h are the hidden variables and $\vec{x}$ is the data.

More precisely, suppose $\underline{h} \in S_{\underline{h}}$ and $\underline{q} \in S_{\underline{h}}$. Suppose $\underline{\vec{x}} \in S_{\underline{x}}^{nsam}$ is a vector of $nsam$ samples and the samples $\underline{x}[\sigma] \in S_{\underline{x}}$ are i.i.d.. The VBA is simply an approximation $P_{\underline{q}|\underline{\vec{x}}}$ to $P_{\underline{h}|\underline{\vec{x}}}$:

$$P_{\underline{h}|\underline{\vec{x}}}(h|\vec{x}) \approx P_{\underline{q}|\underline{\vec{x}}}(h|\vec{x}) \tag{75.1}$$

obtained by minimizing the Kullback-Liebler divergence $D_{KL}(P_{\underline{q}|\underline{\vec{x}}} \parallel P_{\underline{h}|\underline{\vec{x}}})$ over all $P_{\underline{q}|\underline{\vec{x}}}$. The minimization is usually subject to some constraints on the admissible forms of $P_{\underline{q}|\underline{\vec{x}}}$.

$D_{KL}(Q \parallel P) \neq D_{KL}(P \parallel Q)$; i.e., D_{KL} is not symmetric. So why do we use $D_{KL}(P_{\underline{q}|\underline{\vec{x}}} \parallel P_{\underline{h}|\underline{\vec{x}}})$ instead of $D_{KL}(P_{\underline{h}|\underline{\vec{x}}} \parallel P_{\underline{q}|\underline{\vec{x}}})$? Because $D_{KL}(P_{\underline{h}|\underline{\vec{x}}} \parallel P_{\underline{q}|\underline{\vec{x}}})$ requires knowledge of $P_{\underline{h}|\underline{\vec{x}}}$, but calculating $P_{\underline{h}|\underline{\vec{x}}}$ is what we are trying to do in the first place.

Figure 75.1: If $P_{\underline{q}}(h)$ is Gaussian shaped and $P_{\underline{h}}(h)$ has multiple bumps (modes) then $D_{KL}(P_{\underline{q}} \parallel P_{\underline{h}})$ is minimized when $P_{\underline{q}}$ fits one of the modes of $P_{\underline{h}}$. That is because $D_{KL}(P_{\underline{q}} \parallel P_{\underline{h}}) = \sum_h P_{\underline{q}}(h) \ln \frac{P_{\underline{q}}(h)}{P_{\underline{h}}(h)}$ is a weighted average with weights $P_{\underline{q}}$, so nothing going on outside the support of $P_{\underline{q}}$ influences much the final average.

See Fig.75.1 for some intuition on what minimizing $D_{KL}(P_{\underline{q}|\underline{\vec{x}}} \parallel P_{\underline{h}|\underline{\vec{x}}})$ means.

Figure 75.2: $\underline{q}$ and $\underline{h}$ have $nh = 2$ mirroring components and those of $\underline{q}$ are independent at fixed $\underline{\vec{x}}$.

Suppose $\underline{h} = (\underline{h}_0, \underline{h}_1, \ldots, \underline{h}_{nh-1})$ and $\underline{q} = (\underline{q}_0, \underline{q}_1, \ldots, \underline{q}_{nh-1})$ where $\underline{h}_i \in S_{\underline{h}_i}$ and $\underline{q}_i \in S_{\underline{h}_i}$ for all i. We say $\underline{q}$ and $\underline{h}$ have nh mirroring components and those of $\underline{q}$ are independent at fixed $\underline{\vec{x}}$ if

$$P_{\underline{q}|\underline{\vec{x}}}(h|\vec{x}) = \prod_i P_{\underline{q}_i|\underline{\vec{x}}}(h_i|\vec{x}) \ . \tag{75.2}$$

The bnet Fig.75.2 describes the scenario that we have in mind: The samples $\underline{x}[\sigma]$ are i.i.d.. Each component $\underline{h}_i$ of $\underline{h}$ has a mirroring component $\underline{q}_i$ in $\underline{q}$. The components of $\underline{h}$ are correlated whereas those of $\underline{q}$ are independent at fixed $\underline{\vec{x}}$.

Claim 85 *If $\underline{q}$ and $\underline{h}$ have nh mirroring components and those of $\underline{q}$ are independent at fixed $\underline{\vec{x}}$ and $D_{KL}(P_{\underline{q}|\underline{\vec{x}}} \parallel P_{\underline{h}|\underline{\vec{x}}})$ is minimum over all $P_{\underline{q}|\underline{\vec{x}}}$, then*

$$P_{\underline{q}_i|\underline{\vec{x}}}(q_i|\vec{x}) = \mathcal{N}(!q_i) e^{E_{(\underline{q}_j)_{j\neq i}}[\ln P_{\underline{h}|\underline{\vec{x}}}(\underline{h}=q|\vec{x})]} \tag{75.3}$$

$$= \mathcal{N}(!q_i) e^{E_{(\underline{q}_j)_{j\neq i}}[\ln P_{\underline{h},\underline{\vec{x}}}(\underline{h}=q,\vec{x})]} \tag{75.4}$$

for all i.

proof:

Since all quantities in Eq.(75.3) are conditioned on $\vec{x}$, let us omit all mention of $\vec{x}$ in this proof.

Let

$$\mathcal{L} = \mathcal{L}_0 + \mathcal{L}_1 \tag{75.5}$$

where

$$\begin{aligned}
\mathcal{L}_0 &= D_{KL}(P_{\underline{q}} \parallel P_{\underline{h}}) && (75.6)\\
&= \sum_h P_{\underline{q}}(h) \ln \frac{P_{\underline{q}}(h)}{P_{\underline{h}}(h)} && (75.7)\\
&= \sum_h P_{\underline{q}}(h) \ln P_{\underline{q}}(h) - \sum_h P_{\underline{q}}(h) \ln P_{\underline{h}}(h) && (75.8)\\
&= \sum_i \sum_{h_i} P_{\underline{q}_i}(h_i) \ln P_{\underline{q}_i}(h_i) - \sum_h P_{\underline{q}}(h) \ln P_{\underline{h}}(h) && (75.9)
\end{aligned}$$

and

$$\mathcal{L}_1 = \sum_i \lambda_i \left[\sum_{h_i} P_{\underline{q}_i}(h_i) - 1 \right] . \tag{75.10}$$

Then

$$\delta\mathcal{L} = \sum_i \sum_{h_i} \delta P_{\underline{q}_i}(h_i) \left[\ln P_{\underline{q}_i}(h_i) + 1 + \lambda_i - \frac{1}{nh} \sum_{(h_j)_{j\neq i}} \prod_{(h_j)_{j\neq i}} \{P_{\underline{q}_j}(h_j)\} \ln P_{\underline{h}}(h) \right] . \tag{75.11}$$

Hence,

$$P_{\underline{q}_i}(h_i) = \mathcal{N}(!h_i) e^{\sum_{(h_j)_{j\neq i}} \left\{ \prod_{(h_j)_{j\neq i}} P_{\underline{q}_j}(h_j) \right\} \ln P_{\underline{h}}(h)} . \tag{75.12}$$

QED

Note that Eq.(75.3) yields a system of nh nonlinear equations in nh unknowns $(P_{\underline{q}_i|\underline{\vec{x}}})_{i=0,1,\ldots,nh-1}$. This system is usually solved recursively.

75.1 Free Energy $\mathcal{F}(\vec{x})$

To simplify the notation below, let us introduce the following abbreviations:

$$P(h|\vec{x}) = P_{\underline{h}|\underline{\vec{x}}}(h|\vec{x}) \tag{75.13}$$

$$P(h,\vec{x}) = P_{\underline{h},\underline{\vec{x}}}(h,\vec{x}) \tag{75.14}$$

$$P(\vec{x}) = P_{\underline{\vec{x}}}(\vec{x}) \tag{75.15}$$

Note that

$$
\begin{aligned}
D_{KL}(P_{\underline{q}|\underline{\vec{x}}} \parallel P_{\underline{h}|\underline{\vec{x}}}) &= \sum_h P_{\underline{q}|\underline{\vec{x}}}(h|\vec{x}) \ln \frac{P_{\underline{q}|\underline{\vec{x}}}(h|\vec{x})}{P(h|\vec{x})} && (75.16)\\
&= \sum_h P_{\underline{q}|\underline{\vec{x}}}(h|\vec{x}) \ln \frac{P_{\underline{q}|\underline{\vec{x}}}(h|\vec{x})}{P(h,\vec{x})} + \ln P(\vec{x}) && (75.17)\\
&= \mathcal{F}(\vec{x}) + \ln P(\vec{x}) && (75.18)
\end{aligned}
$$

Hence, the **Free energy** $\mathcal{F}(\vec{x})$ is defined as

$$
\begin{aligned}
\mathcal{F}(\vec{x}) &= \sum_h P_{\underline{q}|\underline{\vec{x}}}(h|\vec{x}) \ln \frac{P_{\underline{q}|\underline{\vec{x}}}(h|\vec{x})}{P(h,\vec{x})} && (75.19)\\
&= E_{\underline{q}|\underline{\vec{x}}}\left[\ln \frac{P_{\underline{q}|\underline{\vec{x}}}(q|\vec{x})}{P_{\underline{h},\underline{\vec{x}}}(q,\vec{x})}\right] . && (75.20)
\end{aligned}
$$

The name free energy is justified because

$$
\mathcal{F}(\vec{x}) = \underbrace{-\sum_h P_{\underline{q}|\underline{\vec{x}}}(h|\vec{x}) \ln P_{\underline{h},\underline{\vec{x}}}(h,\vec{x})}_{U,\text{ Internal Energy}} + \underbrace{\sum_h P_{\underline{q}|\underline{\vec{x}}}(h|\vec{x}) \ln P_{\underline{q}|\underline{\vec{x}}}(h|\vec{x})}_{-S,\text{ minus Entropy}} . \tag{75.21}
$$

It is also common to define a quantity called "ELBO" to be the negative of the free energy.

$$
ELBO(\vec{x}) = -\mathcal{F}(\vec{x}) \tag{75.22}
$$

ELBO stands for "Evidence Lower BOund". That name is justified because

$$
\underbrace{\ln P_{\underline{\vec{x}}}(\vec{x})}_{\text{evidence}\leq 0} = \underbrace{D_{KL}(P_{\underline{q}|\underline{\vec{x}}} \parallel P_{\underline{h}|\underline{\vec{x}}})}_{\geq 0} - |ELBO(\vec{x})| . \tag{75.23}
$$

Some properties of $\mathcal{F}$ are:

- **$\mathcal{F}$ is non-negative.**

$$
\underbrace{D_{KL}(P_{\underline{q}|\underline{\vec{x}}} \parallel P_{\underline{h}|\underline{\vec{x}}})}_{\geq 0} + \underbrace{\ln \frac{1}{P_{\underline{\vec{x}}}(\vec{x})]}}_{\geq 0} = \mathcal{F}(\vec{x}) \tag{75.24}
$$

- **KL divergence is min iff $\mathcal{F}$ is min at fixed $P(\vec{x})$.**

Figure 75.3: $D_{KL} + \ln \frac{1}{P(\vec{x})} = \mathcal{F}$.

During a variation δ that holds $P(\vec{x})$ fixed, the KL divergence and $\mathcal{F}$ change by the same amount:

$$\delta D_{KL}(P_{\underline{q}|\underline{\vec{x}}} \parallel P_{\underline{h}|\underline{\vec{x}}}) = \delta \mathcal{F}(\vec{x}) \tag{75.25}$$

Chapter 76

XGBoost

This paper is based on the original XGBoost paper Ref.[4] and the excellent StatQuest videos (highly recommended) [49].

Extreme Gradient Boosting (XGBoost) (Chen, Guestrin, 2016) improves gradient boosting (Friedman, 1999) in a number of ways, such as by using a quadratic rather than linear approximation for the variational function.

In XGBoost, one calculates a sequence of functions where each function tries to correct the errors of the previous function. Then a series (i.e., linear combination) of those functions yields an estimate $\hat{y}^\sigma$ of the target attribute y^σ. As will be shown in this chapter, XGBoost can be used in two cases: Regression (continuous target attribute) and Binary Classification (binary target attribute)[1]. An implementation of the XGBoost algorithm is available as open source. It's written in C++, with Python and R interfaces.

Boosting (see this chapter on XGBoost and Chapter 1 on AdaBoost) and bagging (see Chapter 54 on Random Forest) are two methods of building a classifier function from an ensemble of classifier functions. These two methods are most commonly applied to dtrees: Boosting for an ensemble of small dtrees, and Bagging for a random forest (which is an ensemble of dtrees that are usually much more complicated than small dtrees).

76.1 Divergences

To set up a cost function for XGBoost, we begin by defining 2 types of "divergences" and calculating the first and second derivatives of those divergences:

- **Divergence for regression** (i.e., continuous classification, continuous target attribute). For $x, y \in \mathbb{R}$

$$D_{reg}(x, y) = \frac{1}{2}(x - y)^2 \tag{76.1}$$

[1] XGBoost can also be used for classification into more than two classes, as long as a suitable Divergence function can be defined.

- **Divergence for binary classification** (i.e., binary target attribute). For $p, q \in [0, 1]$.

$$\begin{aligned} D_{bc}(p, q) &= -\{p \ln q + (1-p)\ln(1-q)\} & (76.2) \\ &= CE(\{p, 1-p\} \parallel \{q, 1-q\}) & (76.3) \end{aligned}$$

The cross-entropy $CE()$ is defined in Chapter Notational Conventions and Preliminaries.

Claim 86

$$\partial_{\hat{y}} D_{reg}(y, \hat{y}) = \hat{y} - y \tag{76.4}$$

$$\partial^2_{\hat{y}} D_{reg}(y, \hat{y}) = 1 \tag{76.5}$$

$$D(y, \hat{y} + h) \approx D(y, \hat{y}) + (\hat{y} - y)h + \frac{1}{2}h^2 \tag{76.6}$$

proof: Obvious.
QED

Recall from section 0.21 in Chapter Notational Conventions and Preliminaries that

$$\hat{y} = \text{smoid}(\text{lodds}(\hat{y})) \tag{76.7}$$

Let us abbreviate $s() = \text{smoid}()$ and $l() = \text{lodds}()$ so

$$\hat{y} = s(l(\hat{y}))\ . \tag{76.8}$$

Claim 87 [2]

$$\partial_l D(y, \hat{y}(l)) = \hat{y} - y \tag{76.9}$$

$$\partial^2_l D(y, \hat{y}(l)) = \hat{y}(1 - \hat{y}) \tag{76.10}$$

$$D(y, \hat{y}(l + \Delta l)) \approx D(y, \hat{y}(l)) + (\hat{y} - y)\Delta l + \frac{1}{2}\hat{y}(1 - \hat{y})(\Delta l)^2 \tag{76.11}$$

[2]Note that $D_{bc}(y, \hat{y}) \neq D_{bc}(\hat{y}, y)$; i.e., D_{bc} is not symmetric in its two arguments. Normally $0 < \hat{y} < 1$ and $y \in \{0, 1\}$. Since the log is applied only to the second argument, to avoid logs of zero, it is better to have $\hat{y}$ as the second argument.

proof:

$$\begin{aligned}
\frac{\partial D(y,s)}{\partial s} &= -\partial_s\{y\ln s + (1-y)\ln(1-s)\} && (76.12)\\
&= -\frac{y}{s} + \frac{1-y}{1-s} && (76.13)\\
&= \frac{-y(1-s)+(1-y)s}{s(1-s)} && (76.14)\\
&= \frac{-y+s}{s(1-s)} && (76.15)
\end{aligned}$$

Recall that $\text{smoid}'(l) = \text{smoid}(l)[1-\text{smoid}(l)]$ so

$$\frac{\partial s}{\partial l} = s(1-s)\ . \tag{76.16}$$

$$\frac{\partial D(y,s)}{\partial l} = \frac{\partial s}{\partial l}\ \frac{\partial D(y,s)}{\partial s} = -y+s = -y+\hat{y} \tag{76.17}$$

$$\frac{\partial^2 D(y,s)}{\partial l^2} = \frac{\partial s}{\partial l} = s(1-s) \tag{76.18}$$

QED

76.2 Minimizing Cost function for single tree

Let

$\sigma \in \Sigma$ be an individual in a population Σ
$x^\sigma \in S_{\underline{x}}$ be a feature vector $x^\sigma = (x_i^\sigma)_{i=0,1,\ldots,nf-1}$
$t \in \{0,1,\ldots,nt-1\}$ be the tree index
$\mathcal{L}_t$ be the set of leafs of tree t
$\ell \in \mathcal{L}_t$ be a leaf in tree t
$w_t^\ell \in \mathbb{R}$
$f_t : S_{\underline{x}} \to \mathbb{R}$
$\ell_t : \Sigma \to \mathcal{L}_t$, $\ell_t(\Sigma) = \mathcal{L}_t$.
$\ell_t(\sigma)$ be the leaf of individual σ in tree t
$\Sigma_t^\ell = \{\sigma \in \Sigma : \ell_t(\sigma) = \ell\}$
Define the function f_t by

$$f_t(x^\sigma) = \sum_{\ell\in\mathcal{L}_t} w_t^\ell \mathbb{1}(\sigma \in \Sigma_t^\ell) = w_t^{\ell_t(\sigma)}\ . \tag{76.19}$$

$f_t(x^\sigma)$ gives the output value for tree t and feature vector x^σ.

Define the **cost function** for tree t as follows

$$\mathcal{C}_t = \sum_\sigma D(\hat{y}_t^\sigma, y^\sigma) + \underbrace{\sum_{\ell \in \mathcal{L}_t} \left[\gamma + \frac{\lambda}{2} (w_t^\ell)^2 \right]}_{\text{regulator}} , \tag{76.20}$$

where $\gamma > 0, \lambda > 0$ are regulator parameters.

The estimate $\hat{y}_t^\sigma$, using trees from 0 to t, of the target attribute y^σ, is defined by

$$\begin{aligned}
\hat{y}_0^\sigma &= f_0(x^\sigma) = \text{arbitrary constant, XGBoost default for this is 0.5} && (76.21)\\
\hat{y}_1^\sigma &= f_0(x^\sigma) + f_1(x^\sigma) && (76.22)\\
\hat{y}_2^\sigma &= f_0(x^\sigma) + f_1(x^\sigma) + f_2(x^\sigma) && (76.23)\\
&\vdots && (76.24)\\
\hat{y}_t^\sigma &= \sum_{t' \le t} f_{t'}(x^\sigma) = \hat{y}_{t-1}^\sigma + f_t(x^\sigma) && (76.25)
\end{aligned}$$

In the cost function $\mathcal{C}_t$, we approximate the divergence $D()$ by a second order Taylor approximation:

$$\begin{aligned}
D(y^\sigma, \hat{y}_t^\sigma) &= D(y^\sigma, \hat{y}_{t-1}^\sigma + \underbrace{f_t(x^\sigma)}_{\delta}) && (76.26)\\
&\approx D(y^\sigma, \hat{y}_{t-1}^\sigma) + a_t^\sigma \delta + \frac{1}{2} b_t^\sigma \delta^2 && (76.27)\\
&= D(y^\sigma, \hat{y}_{t-1}^\sigma) + a_t^\sigma w_t^{\ell_t(\sigma)} + \frac{1}{2} b_t^\sigma (w_t^{\ell_t(\sigma)})^2 , && (76.28)
\end{aligned}$$

where

$$a_t^\sigma = [\partial_{\hat{y}} D(y^\sigma, \hat{y})]_{\hat{y} = \hat{y}_{t-1}^\sigma} , \tag{76.29}$$

$$b_t^\sigma = [\partial_{\hat{y}}^2 D(y^\sigma, \hat{y})]_{\hat{y} = \hat{y}_{t-1}^\sigma} . \tag{76.30}$$

a_t^σ is called g for gradient and b_t^σ is called h for Hessian in Ref.[4]. Table 76.1gives the values for a_t^σ and b_t^σ for Regression (reg) and Binary classification (bc).

Define the **residual** for tree t and individual σ by

$$r_t^\sigma = y^\sigma - \hat{y}_{t-1}^\sigma . \tag{76.31}$$

Note that

$$\sum_\sigma = \sum_{\ell \in \mathcal{L}_t} \sum_{\sigma \in \Sigma_t^\ell} . \tag{76.32}$$

	Regression ($y^\sigma \in \mathbb{R}, \hat{y}_t^\sigma \in \mathbb{R},\ D = D_{reg}$)	Binary Classification ($y^\sigma \in \{0,1\},\ \hat{y}_t^\sigma \in [0,1],\ D = D_{bc}$)
a_t^σ	$\hat{y}_{t-1}^\sigma - y^\sigma =$ neg.residual	$\hat{y}_{t-1}^\sigma - y^\sigma =$ neg.residual
b_t^σ	1	$\hat{y}_{t-1}^\sigma(1-\hat{y}_{t-1}^\sigma)$

Table 76.1: The first (a_t^σ) and second (b_t^σ) derivatives for Regression (reg) and Binary classification (bc).

Define

$$A_t^\ell = \sum_{\sigma \in \Sigma_t^\ell} a_t^\sigma \tag{76.33}$$

and

$$B_t^\ell = \sum_{\sigma \in \Sigma_t^\ell} b_t^\sigma \ . \tag{76.34}$$

Now we can rewrite the cost function as

$$\begin{aligned} \mathcal{C}_t &= \underbrace{\sum_\sigma D(\hat{y}_{t-1}^\sigma, y^\sigma)}_{\mathcal{K}} + \sum_{\ell \in \mathcal{L}_t} \left[A_t^\ell w_t^\ell + \frac{1}{2} B_t^\ell (w_t^\ell)^2 \right] + \sum_{\ell \in \mathcal{L}_t} \left[\gamma + \frac{\lambda}{2} (w_t^\ell)^2 \right] && (76.35) \\ &= \sum_{\ell \in \mathcal{L}_t} \left[\gamma + A_t^\ell w_t^\ell + \frac{1}{2}(B_t^\ell + \lambda)(w_t^\ell)^2 \right] \quad \text{(absorbed } \mathcal{K} \text{ into } \gamma) && (76.36) \end{aligned}$$

The cost function $\mathcal{C}_t$ can be minimized over w_t^ℓ:

$$0 = \delta \mathcal{C}_t = \sum_{\ell \in \mathcal{L}_t} \delta w_t^\ell \left[A_t^\ell + (B_t^\ell + \lambda) w_t^\ell \right] \tag{76.37}$$

Therefore, the cost is minimized when

$$w_t^\ell = \frac{-A_t^\ell}{B_t^\ell + \lambda} \ . \tag{76.38}$$

The optimum cost is

$$\mathcal{C}_t = \sum_{\ell \in \mathcal{L}_t} \underbrace{\left[\gamma - \underbrace{\frac{(A_t^\ell)^2}{2(B_t^\ell + \lambda)}}_{SS_t^\ell} \right]}_{c_t^\ell} \tag{76.39}$$

SS_t^ℓ is called the **similarity score** for leaf ℓ and tree t. Note that an increase in similarity decreases the cost.

76.3 Leaf Splitting

Suppose leaf ℓ_0 splits into leafs ℓ_L and ℓ_R. Then

$$\Sigma_t^{\ell_0} = \Sigma_t^{\ell_L} \cup \Sigma_t^{\ell_R}, \quad \Sigma_t^{\ell_L} \cap \Sigma_t^{\ell_R} = \emptyset \tag{76.40}$$

so

$$A_t^{\ell_0} = A_t^{\ell_L} + A_t^{\ell_R} \tag{76.41}$$

and

$$B_t^{\ell_0} = B_t^{\ell_L} + B_t^{\ell_R} \ . \tag{76.42}$$

Hence,

$$\mathcal{C}_t^{\ell_j} = \gamma - \frac{(A_t^{\ell_j})^2}{2(B_t^{\ell_j} + \lambda)} \quad \text{for } j = L, R \tag{76.43}$$

$$\mathcal{C}_t^{\ell_0} = \gamma - \frac{(A_t^{\ell_L} + A_t^{\ell_R})^2}{2(B_t^{\ell_L} + B_t^{\ell_R} + \lambda)} \tag{76.44}$$

Define the **XGBoost branch Gain** for a binary tree branch by

$$GainXGB = \mathcal{C}_t^{\ell_0} - \mathcal{C}_t^{\ell_L} - \mathcal{C}_t^{\ell_R} \tag{76.45}$$

$$= \underbrace{\left\{ SS_t^{\ell_L} + SS_t^{\ell_R} - SS_t^{\ell_0} \right\}}_{\Delta SS_t} - \gamma \tag{76.46}$$

By splitting a leaf, we create a new binary branch with 2 new leafs. We successively add new branches to the tree until some stopping criterion is satisfied. We continue adding branches to the tree as long as they have a positive gain, and as long as the number of levels is smaller than an upper bound input parameter (XGBoost's default maximum number of tree levels is six). XGBoost also rules out leafs that have a denominator quantity B_t^{ℓ} (called the **cover**) smaller than a lower bound input parameter. This branching process is illustrated in Figs.76.1 and 76.2.

76.4 Pruning

If we raise γ or raise λ, branches that previously had positive gain may acquire a negative gain. Get rid of branches from highest level that now have negative gain. This will generate new branches. Get rid of new branches from the highest level that have negative gain. Continue this process until all highest level branches have positive gain. This may reduce a tree to a single node or even rule out the entire tree.

λ measures **level of insensitivity** to observations, and γ measures **level of tree simplicity**.

Figure 76.1: Plot of target attribute y^σ versus a feature vector x^σ with a single component. More generally, the feature vector $x^\sigma = (x^\sigma_j)_{j=0,1,\ldots,nf-1}$ can have multiple components called features or attributes. The left plot refers to a population of 4 individuals and regression so $y^\sigma, \hat{y}^\sigma \in \mathbb{R}$. The right plot refers to a population of 4 individuals and binary classification so $y^\sigma \in \{0,1\}, \hat{y}^\sigma \in [0,1]$. $X_{j,j+1}$ for $j = 0,1,2$ is the average between 2 adjacent values of x^σ.

Figure 76.2: This figure refers to the situation of Fig.76.1. Out of all allowed tree branches, we choose the one with the highest gain.

76.5 Feature Binning

For really large population sizes $|\Sigma|$, it is convenient to bin the set $\{x_i^\sigma : \sigma \in \Sigma\}$ for each i.

A common bin type is quantiles. Quantiles are bins $[X_{j-1}, X_j]$ for $j = 0, 1, \ldots, nbins-1$ that all contain approximately the same number of points. For example, in Fig.76.2, with 1 point quantile bins, use $[X_{j-1}, X_j]$ for $j = 0, 1, 2, 3$. With 2 point quantile bins, use $[X_{j-1}, X_j]$ for $j = 0, 1$.

Use the right edge of each bin as the X_j in the question $x^\sigma < X_j$?, and choose the question which yields the highest gain.

76.6 Final estimate of target attribute

In this section, we will give a formula for the final estimate of the target attribute. Previously, we set the learning rate η to one. Here we restore η to an arbitrary value between 0 and 1.

Instead of using

$$f(x^\sigma) = \sum_{t=0}^{nt-1} f_t(x^\sigma) \ , \tag{76.47}$$

we will use

$$f(x^\sigma) = \sum_{t=0}^{nt-1} (\eta)^t f_t(x^\sigma) \ , \tag{76.48}$$

where $\eta \in [0, 1]$ is called the **learning rate**. This has the effect of compensating for the f_t's with $t > nt - 1$ that would have been included had we used an infinite series. Also, think of Eq.(76.47) as an approximation (a truncated Taylor expansion in powers of Δx) of a function $f(x + \Delta x)$, and think of Eq.(76.48) as an analogous approximation of the function $f(x + \eta\Delta x)$.

For the case of Regression, we get:

$$f(x^\sigma) \ = \ \sum_t (\eta)^t \left(\frac{-A_t^{\ell_t(\sigma)}}{B_t^{\ell_t(\sigma)} + \lambda} \right) \ . \tag{76.49}$$

For the case of Binary Classification, we get

$$f(x^\sigma) = \sum_{t=0}^{nt-1} (\eta)^t \underbrace{\text{smoid} \underbrace{\left(\frac{-A_t^{\ell_t(\sigma)}}{B_t^{\ell_t(\sigma)} + \lambda} \right)}_{\text{lodds(probability)}}}_{\text{probability}} \ . \tag{76.50}$$

76.7 Bnet for XGBoost

Figure 76.3: Bnet for XGBoost assuming $nt = 3$. Nodes that appear multiple times (namely $\underline{\vec{x}}$ and $\underline{\vec{y}}$) should be considered the same node, drawn multiple times for clarity. The parameters λ, γ, η are taken to be global. Recall that $\ell_t : \Sigma \to \mathcal{L}_t$. From the function $\ell_t()$, we can derive its range $\mathcal{L}_t = \{\ell_t(\sigma) : \sigma \in \Sigma\})$, and $\Sigma_t^\ell = \{\sigma \in \Sigma : \ell_t(\sigma) = \ell\}$ for all $\ell \in \mathcal{L}_t$.

Fig.76.3 gives our bnet for the XGBoost algo assuming $nt = 3$. The TPMs, printed in blue, for this bnet, are as follows:

For $t = 0$, ℓ_0 describes a single node "tree" such that $\ell_0(x^\sigma) = f_0(x^\sigma) = 0.5$ for all $\sigma \in \Sigma$. For $t \geq 1$,

$$P(\ell_t|\vec{y}, [\hat{y}_{t-1}^\sigma]_\sigma) = \begin{array}{l}\text{build tree } \ell_t \text{ using } y^\sigma \text{ and } \hat{y}_{t-1}^\sigma \text{ for all } \sigma \in \Sigma. \\ \lambda \text{ and } \gamma \text{ are used here.}\end{array} \tag{76.51}$$

$$P([\hat{y}_t^\sigma]_\sigma \mid f_t, \vec{x}) = \prod_{\sigma\in\Sigma} \mathbb{1}(\quad \hat{y}_t^\sigma = f_t(x^\sigma) \quad) \tag{76.52}$$

$$P([A_t^\ell]_{\ell\in\mathcal{L}_t} \mid \vec{y}, [\hat{y}_{t-1}^\sigma]_\sigma, \ell_t) = \prod_{\ell\in\mathcal{L}_t} \mathbb{1}(\quad A_t^\ell = -\sum_{\sigma\in\Sigma_t^\ell} \underbrace{(y^\sigma - \hat{y}_{t-1}^\sigma)}_{r_t^\sigma} \quad) \tag{76.53}$$

$$P(B_t^\ell]_{\ell\in\mathcal{L}_t} \mid [\hat{y}_{t-1}^\sigma]_\sigma, \ell_t) = \prod_{\ell\in\mathcal{L}_t} \mathbb{1}\left(\quad B_t^\ell = \sum_{\sigma\in\Sigma_t^\ell} \left\{ \begin{array}{ll} 1 & \text{if reg.} \\ \hat{y}_{t-1}^\sigma(1-\hat{y}_{t-1}^\sigma) & \text{if b.c.} \end{array} \right\} \quad \right) \tag{76.54}$$

$$P(f_t \mid [A_t^\ell]_{\ell\in\mathcal{L}_t}, [B_t^\ell]_{\ell\in\mathcal{L}_t}, \ell_t, \vec{x}) = \prod_{\sigma\in\Sigma} \mathbb{1}\left(f_t(x^\sigma) = \left\{ \begin{array}{ll} 1(& \text{if reg.} \\ \text{smoid}(& \text{if b.c.} \end{array} \right\} \frac{-A^{\ell_t(\sigma)}}{2(B^{\ell_t(\sigma)}+\lambda)}) \right) \tag{76.55}$$

$$P(f \mid [f_t]_t, \vec{x}) = \prod_{\sigma\in\Sigma} \mathbb{1}\left(\quad f(x^\sigma) = \sum_t (\eta)^t f_t(x^\sigma) \quad \right) \tag{76.56}$$

Chapter 77

Zero Information Transmission (Graphoid Axioms)

This chapter assumes that you have read Chapter 17 on d-separation.

The following quantities play a very prominent role in the d-separation Theorem that we enunciated in Chapter 17.

- the mutual information (MI)
  (aka information transmission) $H(\underline{a} : \underline{b})$
- the conditional mutual information (CMI)
  (aka conditional information transmission) $H(\underline{a} : \underline{b}|\underline{c})$

MI can be viewed as the special case of CMI, when the set of variables being conditioned on is empty. Particularly prominent in d-separation discussions are probability distributions for which CMI vanishes. The goal of this chapter is to study such probability distributions.

Recall that CMI is non-negative and symmetric in its first two variables (i.e., $H(\underline{a} : \underline{b}|\underline{c}) = H(\underline{b} : \underline{a}|\underline{c})$). Another very useful property of CMI is its chain rule (easy to prove from the definition of CMI):

$$H(\underline{y} : \underline{x}^n) = \sum_i H(\underline{y} : \underline{x}_i|\underline{x}_{<i}) \ , \tag{77.1}$$

where $\underline{x}^n = (\underline{x}_0, \underline{x}_1, \ldots, \underline{x}_{n-1})$ and $\underline{x}_{<i} = (\underline{x}_0, \underline{x}_1, \ldots, \underline{x}_{i-1})$.

A trivial but very useful consequence of the chain rule for CMI is:

$$\boxed{H(\underline{y} : \underline{x}^n) = 0 \iff H(\underline{y} : \underline{x}_i|\underline{x}_{<i}) = 0 \text{ for all } i} \ . \tag{77.2}$$

77.1 Consequences of Eq.(77.2)

Table 77.1 gives a set of statements about CMI referred to as the Graphoid Axioms in chapter 1 of Ref.[37]. See Ref.[37] to learn the history of these axioms. The purpose

of this section is to prove that the graphoid axioms are all a simple consequence of Eq.(77.2).

Symmetry	$\underline{a} \perp_P \underline{b} \implies \underline{b} \perp_P \underline{a}$ $H(\underline{a}:\underline{b}) = 0 \implies H(\underline{b}:\underline{a}) = 0$
Decomposition	$\underline{a} \perp_P \underline{b},\underline{c} \implies \underline{a} \perp_P \underline{b}$ and $\underline{a} \perp_P \underline{c}$ $H(\underline{a}:\underline{b},\underline{c}) = 0 \implies H(\underline{a}:\underline{b}) = 0$ and $H(\underline{a}:\underline{c}) = 0$
Weak Union	$\underline{a} \perp_P \underline{b},\underline{c} \implies \underline{a} \perp_P \underline{b}\vert\underline{c}$ and $\underline{a} \perp_P \underline{c}\vert\underline{b}$ $H(\underline{a}:\underline{b},\underline{c}) = 0 \implies H(\underline{a}:\underline{b}\vert\underline{c}) = 0$ and $H(\underline{a}:\underline{c}\vert\underline{b}) = 0$
Contraction	$\underline{a} \perp_P \underline{b}\vert\underline{c}$ and $\underline{a} \perp_P \underline{c} \implies \underline{a} \perp_P \underline{b},\underline{c}$ $H(\underline{a}:\underline{b}\vert\underline{c}) = 0$ and $H(\underline{a}:\underline{c}) = 0 \implies H(\underline{a}:\underline{b},\underline{c}) = 0$
Intersection	$\underline{a} \perp_P \underline{b}\vert\underline{c},\underline{d}$ and $\underline{a} \perp_P \underline{d}\vert\underline{c},\underline{b} \implies \underline{a} \perp_P \underline{b},\underline{d}\vert\underline{c}$ $H(\underline{a}:\underline{b}\vert\underline{c},\underline{d}) = 0$ and $H(\underline{a}:\underline{d}\vert\underline{c},\underline{b}) = 0 \implies H(\underline{a}:\underline{b},\underline{d}\vert\underline{c}) = 0$

Table 77.1: Graphoid Axioms

Claim 88 *Table 77.1 is true.*

proof:

- **Symmetry**

  Follows trivially from $H(\underline{a}:\underline{b}) = H(\underline{b}:\underline{a})$.

- **Decomposition**

  From the chain rule for CMI, we have

$$H(\underline{a}:\underline{b},\underline{c}) = H(\underline{a}:\underline{b}|\underline{c}) + H(\underline{a}:\underline{c}) \ , \tag{77.3}$$

  and

$$H(\underline{a}:\underline{b},\underline{c}) = H(\underline{a}:\underline{c}|\underline{b}) + H(\underline{a}:\underline{b}) \ . \tag{77.4}$$

  Hence,

$$H(\underline{a}:\underline{b},\underline{c}) = 0 \tag{77.5}$$

  implies

$$H(\underline{a}:\underline{b}|\underline{c}) = H(\underline{a}:\underline{c}) = 0 \ , \tag{77.6}$$

  and

$$H(\underline{a}:\underline{c}|\underline{b}) = H(\underline{a}:\underline{b}) = 0 \ . \tag{77.7}$$

- **Weak Union**

  Already proven in proof of Decomposition.

- **Contraction**

  From chain rule for CMI, we have

$$H(\underline{a} : \underline{b}, \underline{c}) = H(\underline{a} : \underline{b}|\underline{c}) + H(\underline{a} : \underline{c}) \ . \tag{77.8}$$

- **Intersection**

  From the chain rule for CMI, we have

$$H(\underline{a} : \underline{b}, \underline{d}|\underline{c}) = H(\underline{a} : \underline{b}|\underline{d}, \underline{c}) + H(\underline{a} : \underline{d}|\underline{c}) \ , \tag{77.9}$$

  and

$$H(\underline{a} : \underline{b}, \underline{d}|\underline{c}) = H(\underline{a} : \underline{d}|\underline{b}, \underline{c}) + H(\underline{a} : \underline{b}|\underline{c}) \ . \tag{77.10}$$

  Thus,

$$H(\underline{a} : \underline{b}, \underline{d}|\underline{c}) = 0 \tag{77.11}$$

  implies

$$H(\underline{a} : \underline{b}|\underline{d}, \underline{c}) = H(\underline{a} : \underline{d}|\underline{c}) = 0 \ , \tag{77.12}$$

  and

$$H(\underline{a} : \underline{d}|\underline{b}, \underline{c}) = H(\underline{a} : \underline{b}|\underline{c}) = 0 \ . \tag{77.13}$$

.

QED

Bibliography

[1] Elias Bareinboim, Jin Tian, and Judea Pearl. Recovering from selection bias in causal and statistical inference. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014. https://ftp.cs.ucla.edu/pub/stat_ser/r425.pdf.

[2] Dan Bendel. Metropolis-Hastings: A comprehensive overview and proof. https://similarweb.engineering/mcmc/.

[3] Alexandra M Carvalho. Scoring functions for learning Bayesian networks. http://www.lx.it.pt/~asmc/pub/talks/09-TA/ta_pres.pdf.

[4] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. https://arxiv.org/abs/1603.02754.

[5] Carlos Cinelli, Andrew Forney, and Judea Pearl. A crash course in good and bad controls. https://ftp.cs.ucla.edu/pub/stat_ser/r493.pdf.

[6] Scott Cunningham. *Causal inference: The mixtape*. Yale University Press, 2021. https://mixtape.scunning.com/index.html.

[7] Robin J. Evans. Graphical methods for inequality constraints in marginalized DAGs. https://arxiv.org/abs/1209.2978.

[8] Matheus Facure Alves. *Causal Inference for The Brave and True*. 2021. https://matheusfacure.github.io/python-causality-handbook/landing-page.html.

[9] George Fei. Modeling uplift directly: Uplift decision tree with kl divergence and euclidean distance as splitting criteria. https://www.aboutwayfair.com/tech-innovation/modeling-uplift-directly-uplift-decision-tree-with-kl-divergence-and-euclidean-distance-as-splitting-criteria.

[10] Charles Fox, Neil Girdhar, and Kevin Gurney. A causal Bayesian network view of reinforcement learning. https://www.aaai.org/Papers/FLAIRS/2008/FLAIRS08-030.pdf".

[11] Bruno Gonçalves. Model testing and causal search. blog post https://medium.com/data-for-science/causal-inference-part-vii-model-testing-and-causal-search-536b796f0384.

[12] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, David Warde-Farley Bing Xu, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. https://arxiv.org/abs/1406.2661.

[13] Pierre Gutierrez and Jean-Yves Grardy. Causal inference and uplift modelling: A review of the literature. In *Proceedings of The 3rd International Conference on Predictive Applications and APIs*, pages 1–13, 2017. http://proceedings.mlr.press/v67/gutierrez17a.html.

[14] James Douglas Hamilton. *Time series analysis*. Princeton university press, 2020.

[15] Christina Heinze-Deml. Causality, spring semester 2019 at ETH Zurich. https://stat.ethz.ch/lectures/ss19/causality.php#course_materials.

[16] Cecil Huang and Adnan Darwiche. Inference in belief networks: A procedural guide. *International journal of approximate reasoning*, 15(3):225–263, 1996. http://www.ar-tiste.com/Huang-Darwiche1996.pdf.

[17] Michael I. Jordan. course: Stat260-Bayesian modeling and inference, lecture date: February 8-2010, title: The conjugate prior for the Normal distribution. https://people.eecs.berkeley.edu/~jordan/courses/260-spring10/lectures/lecture5.pdf.

[18] Chaitanya K. Joshi. Transformer (machine learning model). https://graphdeeplearning.github.io/post/transformers-are-gnns/.

[19] Chung-Ming Kuan. Introduction to time series analysis, Fall 2014 lectures given at the Department of Finance, National Taiwan University. https://homepage.ntu.edu.tw/~ckuan/pdf/2014fall/Lec-TimeSeries_slide-Fall2014.pdf.

[20] Steffen L Lauritzen and David J Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society: Series B (Methodological)*, 50(2):157–194, 1988. http://www.eecis.udel.edu/~shatkay/Course/papers/Lauritzen1988.pdf.

[21] Sergey Levine. Course CS 285 at UC Berkeley, Deep reinforcement learning. http://rail.eecs.berkeley.edu/deeprlcourse/.

[22] Dimitris Margaritis. Learning Bayesian network model structure from data (thesis, 2003, Carnegie Mellon Univ). https://apps.dtic.mil/sti/citations/ADA461103.

[23] Adam A Margolin, Ilya Nemenman, Katia Basso, Chris Wiggins, Gustavo Stolovitzky, Riccardo Dalla Favera, and Andrea Califano. Aracne: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. In *BMC bioinformatics*, volume 7, page S7. Springer, 2006. https://link.springer.com/article/10.1186/1471-2105-7-S1-S7.

[24] Samuele Mazzanti. Black-box models are actually more explainable than a logistic regression. https://towardsdatascience.com/black-box-models-are-actually-more-explainable-than-a-logistic-regression-f263c22795d.

[25] Samuele Mazzanti. SHAP values explained exactly how you wished someone explained to you. https://towardsdatascience.com/shap-explained-the-way-i-wish-someone-explained-it-to-me-ab81cc69ef30.

[26] Robert J. McEliece, David J. C. MacKay, and Jung-Fu Cheng. Turbo decoding as an instance of Pearls belief propagation algorithm. http://authors.library.caltech.edu/6938/1/MCEieeejstc98.pdf.

[27] Brady Neal. Introduction to causal inference, Fall 2020, lectures and book. https://www.bradyneal.com/causal-inference-course.

[28] Richard E Neapolitan. *Learning Bayesian networks*. Pearson Prentice Hall, 2004.

[29] Andrew Ng. Lecture at deeplearning.ai on recurrent neural networks. http://www.ar-tiste.com/ng-lec-rnn.pdf.

[30] Gregory Nuel. Tutorial on exact belief propagation in Bayesian networks: from messages to algorithms. https://arxiv.org/abs/1201.4724.

[31] Judea Pearl. Linear models: A useful microscope for causal analysis. https://ftp.cs.ucla.edu/pub/stat_ser/r409-corrected-reprint.pdf.

[32] Judea Pearl. Mediating instrumental variables. https://ftp.cs.ucla.edu/pub/stat_ser/r210.pdf.

[33] Judea Pearl. On the testability of causal models with latent and instrumental variables. https://arxiv.org/abs/1302.4976.

[34] Judea Pearl. Reverend Bayes on inference engines: A distributed hierarchical approach. https://www.aaai.org/Papers/AAAI/1982/AAAI82-032.pdf, 1982.

[35] Judea Pearl. *Probabilistic Inference in Intelligent Systems*. Morgan Kaufmann, 1988.

[36] Judea Pearl. The causal mediation formula—a guide to the assessment of pathways and mechanisms. *Prevention science*, 13(4):426–436, 2012. https://apps.dtic.mil/sti/pdfs/ADA557663.pdf.

[37] Judea Pearl. *Causality: Models, Reasoning, and Inference, Second Edition*. Cambridge University Press, 2013.

[38] Judea Pearl. Causal and counterfactual inference. *The Handbook of Rationality*, pages 1–41, 2019. https://ftp.cs.ucla.edu/pub/stat_ser/r485.pdf.

[39] Judea Pearl and Elias Bareinboim. Transportability of causal and statistical relations: A formal approach. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 25, 2011. https://ojs.aaai.org/index.php/AAAI/article/view/7861.

[40] Judea Pearl, Madelyn Glymour, and Nicholas P Jewell. *Causal inference in statistics: A primer*. John Wiley & Sons, 2016.

[41] Judea Pearl and Dana Mackenzie. *The book of why: the new science of cause and effect*. Basic Books, 2018.

[42] Ashwin Rao and Tikhon Jelvis. *Foundations of Reinforcement Learning with Applications in Finance*. https://stanford.edu/~ashlearn/RLForFinanceBook/book.pdf.

[43] ReliaSoft. System analysis reference. http://reliawiki.org/index.php/System_Analysis_Reference.

[44] Piotr Rzepakowski and Szymon Jaroszewicz. Decision trees for uplift modeling with single and multiple treatments. *Knowledge and Information Systems*, 32(2):303–327, 2012. https://link.springer.com/content/pdf/10.1007/s10115-011-0434-0.pdf.

[45] Scholarpedia. Granger causality. http://www.scholarpedia.org/article/Granger_causality.

[46] Marco Scutari. bnlearn. https://www.bnlearn.com/.

[47] Marco Scutari, Catharina Elisabeth Graafland, and José Manuel Gutiérrez. Who learns better Bayesian network structures: Accuracy and speed of structure learning algorithms. *International Journal of Approximate Reasoning*, 115:235–253, 2019. https://arxiv.org/abs/1805.11908.

[48] Nitish Srivastava, G E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. http://www.cs.toronto.edu/~rsalakhu/papers/srivastava14a.pdf.

[49] StatQuest. XGBoost Parts 1-4 (videos). https://statquest.org/video-index/.

[50] Russell Stuart and Norvig Peter. Artificial intelligence-a modern approach 3rd ed, 2016.

[51] Masayoshi Takahashi. Statistical inference in missing data by MCMC and non-MCMC multiple imputation algorithms: Assessing the effects of between-imputation iterations. *Data Science Journal*, 16, 2017. https://datascience.codata.org/articles/10.5334/dsj-2017-037/.

[52] theinvestorsbook.com. Pert analysis. https://theinvestorsbook.com/pert-analysis.html.

[53] Robert R. Tucci. Bayesian networks (aka causal models, DAGs) and the passage of time. blog post in blog Quantum Bayesian Networks, https://qbnets.wordpress.com/2021/07/16/bayesian-networks-aka-causal-models-dags-and-the-passage-of-time/.

[54] Robert R. Tucci. Bell's inequalities for Bayesian statisticians. blog post in blog Quantum Bayesian Networks, https://qbnets.wordpress.com/2008/09/19/bells-inequaties-for-bayesian-statistician/.

[55] Robert R. Tucci. Goodness of causal fit. https://arxiv.org/abs/2105.02172.

[56] Robert R. Tucci. Quantum d-separation and quantum belief propagation. https://arxiv.org/abs/2012.09635.

[57] Robert R. Tucci. Quantum Fog. https://github.com/artiste-qb-net/quantum-fog.

[58] W. E. Vesely, F. F. Goldberg, N. H. Roberts, and D. F. Haasl. Fault tree handbook nureg-0492. https://www.nrc.gov/reading-rm/doc-collections/nuregs/staff/sr0492/.

[59] Lilian Weng. The multi-armed bandit problem and its solutions. lilianweng.github.io/lil-log, http://lilianweng.github.io/lil-log/2018/01/23/the-multi-armed-bandit-problem-and-its-solutions.html, 2018.

[60] Wikipedia. AdaBoost. https://en.wikipedia.org/wiki/AdaBoost.

[61] Wikipedia. Autoregressive model. https://en.wikipedia.org/wiki/Autoregressive_model.

[62] Wikipedia. Autoregressive moving-average model. https://en.wikipedia.org/wiki/Autoregressive%E2%80%93moving-average_model.

[63] Wikipedia. Baum-Welsh algorithm. https://en.wikipedia.org/wiki/Baum%E2%80%93Welch_algorithm.

[64] Wikipedia. Belief propagation. https://en.wikipedia.org/wiki/Belief_propagation.

[65] Wikipedia. Berkson's paradox. https://en.wikipedia.org/wiki/Berkson%27s_paradox.

[66] Wikipedia. Bernoulli distribution. https://en.wikipedia.org/wiki/Bernoulli_distribution.

[67] Wikipedia. Beta distribution. https://en.wikipedia.org/wiki/Beta_distribution.

[68] Wikipedia. Beta function. https://en.wikipedia.org/wiki/Beta_function.

[69] Wikipedia. Binary decision diagram. https://en.wikipedia.org/wiki/Binary_decision_diagram.

[70] Wikipedia. Boolean algebra. https://en.wikipedia.org/wiki/Boolean_algebra.

[71] Wikipedia. Bootstrap aggregating. https://en.wikipedia.org/wiki/Bootstrap_aggregating.

[72] Wikipedia. Categorical distribution. https://en.wikipedia.org/wiki/Categorical_distribution.

[73] Wikipedia. Chi-square distribution. https://en.wikipedia.org/wiki/Chi-square_distribution.

[74] Wikipedia. Chow-Liu tree. https://en.wikipedia.org/wiki/Chow%E2%80%93Liu_tree.

[75] Wikipedia. Cochran's theorem. https://en.wikipedia.org/wiki/Cochran%27s_theorem.

[76] Wikipedia. Conjugate prior. https://en.wikipedia.org/wiki/Conjugate_prior.

[77] Wikipedia. Cross-validation. https://en.wikipedia.org/wiki/Cross-validation_(statistics).

[78] Wikipedia. Data processing inequality. https://en.wikipedia.org/wiki/Data_processing_inequality.

[79] Wikipedia. Dirichlet distribution. https://en.wikipedia.org/wiki/Dirichlet_distribution.

[80] Wikipedia. Errors in variables models. https://en.wikipedia.org/wiki/Errors-in-variables_models.

[81] Wikipedia. Expectation maximization. https://en.wikipedia.org/wiki/Expectation%E2%80%93maximization_algorithm.

[82] Wikipedia. Frisch-Waugh-Lovell theorem. https://en.wikipedia.org/wiki/Frisch%E2%80%93Waugh%E2%80%93Lovell_theorem.

[83] Wikipedia. Gamma distribution. https://en.wikipedia.org/wiki/Gamma_distribution.

[84] Wikipedia. Gamma function. https://en.wikipedia.org/wiki/Gamma_function.

[85] Wikipedia. Gated recurrent unit. https://en.wikipedia.org/wiki/Gated_recurrent_unit.

[86] Wikipedia. Gibbs sampling. https://en.wikipedia.org/wiki/Gibbs_sampling.

[87] Wikipedia. Granger causality. https://en.wikipedia.org/wiki/Granger_causality.

[88] Wikipedia. Hidden Markov model. https://en.wikipedia.org/wiki/Hidden_Markov_model.

[89] Wikipedia. Hoeffding's inequality. https://en.wikipedia.org/wiki/Hoeffding%27s_inequality.

[90] Wikipedia. Importance sampling. https://en.wikipedia.org/wiki/Importance_sampling.

[91] Wikipedia. Instrumental variables estimation. https://en.wikipedia.org/wiki/Instrumental_variables_estimation.

[92] Wikipedia. Inverse transform sampling. https://en.wikipedia.org/wiki/Inverse_transform_sampling.

[93] Wikipedia. Jackknife resampling. https://en.wikipedia.org/wiki/Cross-validation_(statistics).

[94] Wikipedia. Junction tree algorithm. https://en.wikipedia.org/wiki/Junction_tree_algorithm.

[95] Wikipedia. k-means clustering. https://en.wikipedia.org/wiki/K-means_clustering.

[96] Wikipedia. Kalman filter. https://en.wikipedia.org/wiki/Kalman_filter.

[97] Wikipedia. Kernel method. https://en.wikipedia.org/wiki/Kernel_method.

[98] Wikipedia. Kernel perceptron. https://en.wikipedia.org/wiki/Kernel_perceptron.

[99] Wikipedia. Least squares. https://en.wikipedia.org/wiki/Least_squares.

[100] Wikipedia. Likelihood-ratio test. https://en.wikipedia.org/wiki/Likelihood-ratio_test.

[101] Wikipedia. Linear regression. https://en.wikipedia.org/wiki/Linear_regression.

[102] Wikipedia. Long short term memory. https://en.wikipedia.org/wiki/Long_short-term_memory.

[103] Wikipedia. Markov blanket. https://en.wikipedia.org/wiki/Markov_blanket.

[104] Wikipedia. Metropolis-Hastings method. https://en.wikipedia.org/wiki/Metropolis%E2%80%93Hastings_algorithm.

[105] Wikipedia. Minimum spanning tree. https://en.wikipedia.org/wiki/Minimum_spanning_tree.

[106] Wikipedia. Monte Carlo methods. https://en.wikipedia.org/wiki/Category:Monte_Carlo_methods.

[107] Wikipedia. Moving-average model. https://en.wikipedia.org/wiki/Moving-average_model.

[108] Wikipedia. Multinomial distribution. https://en.wikipedia.org/wiki/Multinomial_distribution.

[109] Wikipedia. Multinomial theorem. https://en.wikipedia.org/wiki/Multinomial_theorem.

[110] Wikipedia. Multivariate normal distribution. https://en.wikipedia.org/wiki/Multivariate_normal_distribution.

[111] Wikipedia. Natural experiment. https://en.wikipedia.org/wiki/Natural_experiment.

[112] Wikipedia. Non-negative matrix factorization. https://en.wikipedia.org/wiki/Non-negative_matrix_factorization.

[113] Wikipedia. Ordinary least squares. https://en.wikipedia.org/wiki/Ordinary_least_squares.

[114] Wikipedia. Program evaluation and review technique. https://en.wikipedia.org/wiki/Program_evaluation_and_review_technique.

[115] Wikipedia. Random forest. https://en.wikipedia.org/wiki/Random_forest.

[116] Wikipedia. Receiver operating characteristic. https://en.wikipedia.org/wiki/Receiver_operating_characteristic.

[117] Wikipedia. Rejection sampling. https://en.wikipedia.org/wiki/Rejection_sampling.

[118] Wikipedia. Score test. https://en.wikipedia.org/wiki/Score_test.

[119] Wikipedia. Simple linear regression. https://en.wikipedia.org/wiki/Simple_linear_regression.

[120] Wikipedia. Simpson's paradox. https://en.wikipedia.org/wiki/Simpson's_paradox.

[121] Wikipedia. Spring system. https://en.wikipedia.org/wiki/Spring_system.

[122] Wikipedia. Student's t-distribution. https://en.wikipedia.org/wiki/Student%27s_t-distribution.

[123] Wikipedia. Support vector machine. https://en.wikipedia.org/wiki/Support-vector_machine.

[124] Wikipedia. Time series. https://en.wikipedia.org/wiki/Time_series.

[125] Wikipedia. Transfer learning. https://en.wikipedia.org/wiki/Transfer_learning.

[126] Wikipedia. Transformer (machine learning model). https://en.wikipedia.org/wiki/Transformer_(machine_learning_model).

[127] Wikipedia. Uplift modelling. https://en.wikipedia.org/wiki/Uplift_modelling.

[128] Wikipedia. Variational Bayesian methods. https://en.wikipedia.org/wiki/Variational_Bayesian_methods.

[129] Wikipedia. Vector autoregression. https://en.wikipedia.org/wiki/Vector_autoregression.

[130] Wikipedia. Viterbi algorithm. https://en.wikipedia.org/wiki/Viterbi_algorithm.

[131] Wikipedia. Wald test. https://en.wikipedia.org/wiki/Wald_test.

[132] Hao Wu and Zhaohui Steve Qin. course notes, BIOS731: Advanced statistical computing, 2016 Emory Univ. http://web1.sph.emory.edu/users/hwu30/teaching/statcomp/statcomp.html.