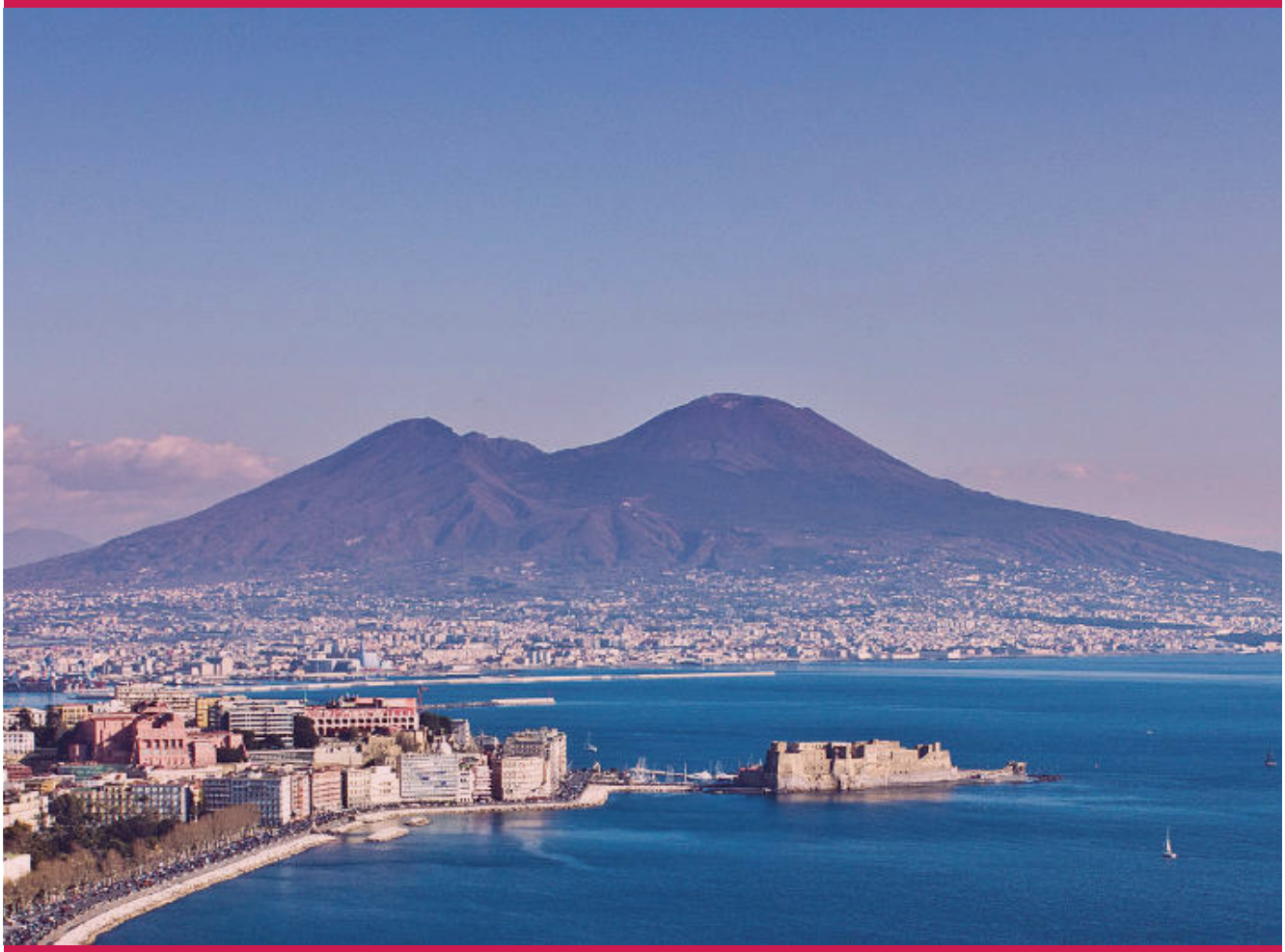
BAYESUVIUS

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

February 3, 2024

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

Figure 1: View of Mount Vesuvius from Pompeii

Figure 2: Mount Vesuvius and Bay of Naples

Contents

Foreword **17**

Appendices **18**

A Navigating the ocean of Judea Pearl's Books **19**

B CI-2-3 track **20**

C Notational Conventions and Preliminaries **24**
C.1 Some abbreviations frequently used throughout this book 24
C.2 $\mathcal{N}(!a)$ 24
C.3 Indicator function (a.k.a. Truth function) 24
C.4 One hot vector 25
C.5 L^p norm 25
C.6 Special sets 27
C.7 Kronecker delta function 27
C.8 Dirac delta function 27
C.9 Majority function 27
C.10 Underlined letters indicate random variables 27
C.11 Probability distributions 28
C.12 Discretization of continuous probability distributions 28
C.13 Samples, i.i.d. variables 29
C.14 Expected Value and Variance 29
C.15 Conditional Expected Value 30
C.16 Notation for covariances 30
C.17 Conditional Covariance 31
C.18 Normal Distribution 32
C.19 Uniform Distribution 33
C.20 Softmax function (a.k.a. Boltzmann Distribution) 33
C.21 Sigmoid and log-odds functions 34
C.22 Estimand, Estimator (curve-fit), Estimate, Bias 35
C.23 Maximum Likelihood Estimate, Likelihood Ratio Test 36
C.24 Mean Square Error (MSE) 37
C.25 Cramer-Rao Bound 39

C.26 Bayes Rule, Bayesian Updating And Conjugate Priors 43
C.27 Linear regression, Ordinary Least Squares (OLS) 44
C.27.1 LR, assuming x_σ are non-random 45
Derivation of LR From Minimization of Error 46
Geometry of LR with non-random x_σ. 47
LR Goodness of Fit, R^2 48
C.27.2 LR, assuming x_σ are random 50
Transforming expressions from non-random to random x_σ . . 51
LR with random x_σ, expressed in derivative notation 53
Double regression of y 57
R^2 with random x_σ 59
C.28 Logistic Regression (LoR) 60
C.29 Entropy, Kullback-Leibler divergence, Cross-Entropy 60
C.30 Definition of various entropies used in Shannon Information Theory . 61
C.31 Mean log likelihood asymptotic behavior 62
C.32 Arc Strength (Arc Force) 64
C.33 Pearson Chi-Squared Test 64
C.34 Demystifying Population and Sample Variances 65
C.35 Independence of $\widehat{\mu}$ and $\widehat{\sigma^2}$ 67
C.36 Chi-square distribution 68
C.37 Student's t-distribution 69
C.38 Hypothesis testing and 3 classic test statistics (Likelihood, Score, Wald) 72
C.39 Error Bars 75
C.40 Confidence Interval 76
C.41 Score p-value 78
C.42 Convex/Concave functions, Jensen's Inequality 80
C.43 Chebyshev's inequality 81
C.44 Short Summary of Boolean Algebra 83
C.45 Laplace transform 84
C.45.1 Examples 86
C.45.2 Properties 87
C.46 Z-transform 93
C.46.1 Examples 96
C.46.2 Properties 97
C.47 Legendre Transformation (dual functions) 100
C.47.1 Examples 101
C.47.2 Properties 103
C.47.3 Connection to Fourier transform and Quantum Mechanics . . 106
C.48 Numpy tensor methods 107

D Definition of a Bayesian Network **113**

E Bayesian Networks, Causality and the Passage of Time **117**
E.1 Unifying Principle of this book . . . 117
E.2 You say tomato, I say tomato . . . 118
E.3 A dataset is causal model free . . . 118
E.4 What is causality? . . . 119
E.5 Bayesian Networks and the passage of time . . . 120
E.6 Advice for the DAG-phobic . . . 121

1 AdaBoost **122**
1.1 AdaBoost for general ensemble of w-classifiers . . . 122
1.2 AdaBoost for ensemble of tree stumps . . . 126

2 ANOVA **128**
2.1 Law of Total Variance . . . 128
2.2 Sum of Squares Estimates . . . 129
2.3 F-statistic and hypothesis testing . . . 131

3 ARACNE structure learning **133**

4 Backdoor Adjustment Formula **135**
4.1 Examples . . . 136

5 Back Propagation (Automatic Differentiation) **140**
5.1 Toy Example . . . 140
5.2 General Theory . . . 141
5.2.1 Jacobians . . . 141
5.2.2 Bnets for function composition, forward propagation and back propagation . . . 142
5.3 Application to Neural Networks . . . 144
5.3.1 Absorbing b_i^λ into $w_{i|j}$. . . . 144
5.3.2 Bnets for function composition, forward propagation and back propagation for NN . . . 145
5.4 General bnets instead of Markov chains induced by layered structure of NNs . . . 148

6 Bell and Clauser-Horne Inequalities in Quantum Mechanics **149**

7 Berkson's Paradox **150**

8 Binary Decision Diagrams **152**

9 Chow-Liu Trees and Tree Augmented Naive Bayes (TAN) **156**
9.1 Chow-Liu Trees . . . 156
9.2 Tree Augmented Naive Bayes (TAN) . . . 160

10 Control Theory (linear, deterministic) 162
10.1 Basic feedback model . . . 163
10.2 Classical model (analog) . . . 164
10.3 Modern model (analog) . . . 167
10.4 Classical model (digital) . . . 171
10.5 Modern model (digital) . . . 171
10.5.1 Discretizing derivatives . . . 172
10.5.2 Solving Difference Equation . . . 173
10.6 Higher than first order differential (or difference) equations . . . 175
10.6.1 Differential Equations . . . 175
10.6.2 Difference Equations . . . 176
10.7 Time-Invariance, Causality, Stability . . . 177
10.8 Controllability, Observability . . . 178
10.9 Signal Flow Graph . . . 178

11 Copula 183
11.1 Examples . . . 186

12 Counterfactual Reasoning 189
12.1 The 3 Rungs of Causal AI . . . 189
12.2 Do operator . . . 190
12.3 Imagine operator . . . 190

13 Cross-Validation 194

14 DAG Extraction From Text (DEFT) 197

15 Dataset Shift and Batch Normalization 198
15.1 Covariate Shift . . . 199
15.2 Concept Shift . . . 199
15.3 Batch Normalization . . . 200

16 Decision Trees 201
16.1 Transforming a dtree into a bnet . . . 203
16.2 Structure Learning for Dtrees . . . 204
16.2.1 Information Gain, Gini . . . 205
16.3 Information Gain Ratio . . . 208
16.3.1 Pseudo-code . . . 209

17 Decisions Based on Rungs 2 and 3: COMING SOON 211

18 Difference-in-Differences **212**
18.1 John Snow, DID and a cholera transmission pathway 212
18.2 PO analysis 214
18.3 Linear Regression 216

19 Diffusion Models **219**
19.1 Bnet for DM 219
19.2 Mean Values $M^{t-1}(x^t)$ and $M_\theta^{t-1}(x^t)$ 222
19.3 Loss function $\mathcal{L}$ 225
19.4 Algorithms for training and sampling DM 227

20 Digital Circuits **229**
20.1 Mapping any dcircuit to a bnet 229
20.1.1 Option A of Fig.20.2 229
20.1.2 Option B of Fig.20.2 230

21 Do Calculus **231**
21.1 3 Rules of Do Calculus 234
21.2 Parent Adjustment Formula 235
21.3 Backdoor Adjustment Formula 237
21.4 Frontdoor Adjustment Formula 238
21.5 Comparison of Backdoor and Frontdoor adjustment formulae 239
21.6 Do operator for DEN diagrams 240

22 Do Calculus proofs **243**

23 D-Separation **261**

24 D-Separation in Quantum Mechanics **264**

25 Dynamical Bayesian Networks **265**

26 Expectation Maximization **267**
26.1 The EM algorithm: 268
26.1.1 Motivation 269
26.2 Minorize-Maximize (MM) algorithms 269
26.3 Examples 271
26.3.1 Gaussian mixture 271
26.3.2 Blood Genotypes and Phenotypes 272
26.3.3 Missing Data/Imputation 274

27 Factor Graphs **275**

28 Frisch-Waugh-Lovell (FWL) theorem **278**
28.1 FWL, assuming x^σ are non-random . . . 278
28.2 FWL, assuming x^σ are random . . . 279

29 Frontdoor Adjustment Formula **281**
29.1 Examples . . . 282

30 G-formula (Sequential Backdoor Adjustment Formula) **283**

31 Gaussian Nodes with Linear Dependence on Parents **287**

32 Generalized Linear Model (GLM) **290**
32.1 Exponential Family of Distributions . . . 290
32.2 GLM . . . 292

33 Generative Adversarial Networks (GANs) **297**

34 Goodness of Causal Fit **302**

35 Gradient Descent **303**

36 Granger Causality **305**

37 Hidden Markov Model **308**
37.1 Calculating $P(x_t, v^n)$ and $P(x_t, x_{t+1}, v^n)$ . . . 310
37.2 Calculating $\mathcal{F}_t$ and $\overline{\mathcal{F}}_t$ . . . 311
37.3 Calculating $P(x^n|v^n)$ . . . 312
37.4 Calculating $P(v^n|A, B, \pi)$ . . . 313
37.5 Calculating $\widehat{x}^n$ (Viterbi algorithm) . . . 314
37.6 Calculating $\widehat{A}$, $\widehat{B}$, $\widehat{\pi}$ (Baum-Welch algorithm) . . . 316

38 Identification of do queries via LDEN diagrams **318**

39 Influence Diagrams & Utility Nodes **320**

40 Instrumental Inequality and beyond **322**
40.1 I-inequality . . . 322
40.1.1 I-inequality for binary z,d,y . . . 324
40.2 Bounds on Effect of IV on treatment outcome y . . . 325

41 Instrumental Variables **328**
41.1 δ with unmeasured confounder . . . 328
41.2 δ (with unmeasured confounder) can be inferred via IV . . . 329
41.3 More general bnets with IVs . . . 330
41.4 Instrumental Inequality . . . 331

42 Jackknife Resampling **332**
42.1 Case $A = A^n(\vec{x}) = \frac{1}{n}\sum_\sigma x^\sigma$ 334

43 Junction Tree Algorithm **336**

44 Kalman Filter **337**
44.1 Prediction Problem 338
44.2 Solution 339
44.3 Simple Example 340
44.4 Invariants 341
44.5 Derivation of Solution 341

45 LATE (Local Average Treatment Effect) **343**

46 LDEN with feedback loops **349**

47 Linear and Logistic Regression **356**
47.1 Generalization to x with multiple components (features) 358
47.2 Alternative $V(b, m)$ for logistic regression 358

48 Linear Deterministic Bnets with External Noise **360**
48.1 Example of LDEN diagram 360
48.2 LDEN equations and their 2 solutions 361
48.3 Fully connected LDEN diagrams 362
48.3.1 Fully connected LDEN diagram with $nx = 2$ 362
48.3.2 Fully connected LDEN diagram with $nx = 3$ 364
48.3.3 Fully connected LDEN diagram with arbitrary nx 366
48.4 Not fully connected LDEN diagrams 368
48.5 LDEN diagram with conditioned nodes 369
48.6 SCuMpy 369
48.7 Non-linear DEN diagrams 369

49 Marginalizer Nodes **371**

50 Markov Blankets **373**

51 Markov Chain Monte Carlo (MCMC) **375**
51.1 Inverse Cumulative Sampling 375
51.2 Rejection Sampling 377
51.3 Metropolis-Hastings Sampling 378
51.4 Gibbs Sampling 381
51.5 Importance Sampling 382

52 Markov Chains **384**

53 Mediation Analysis **385**

54 Mendelian Randomization **392**

55 Message Passing and Bethe Free Energy **394**
55.1 2MRFs . . . 394
55.2 Message Passing Intuition . . . 395
55.3 $-\ln Z_\theta$= Free Energy (FE) . . . 399
55.4 $-\ln Z_{\theta^*}$= Minimum FE . . . 400
55.5 $-\ln Z_\theta^{tree}$=Tree FE (a.k.a. Bethe FE) . . . 401
55.6 $-\ln Z_{\theta^*}^{tree}$= Tree Minimum FE, and message passing . . . 402

56 Message Passing, Pearl's theory **406**
56.1 Distributed Soldier Counting . . . 406
56.2 Spring Systems . . . 408
56.3 BP for Markov Chains . . . 408
56.4 BP Algorithm for Polytrees . . . 416
56.4.1 How BP algo for polytrees reduces to the BP algo for Markov chains . . . 419
56.5 Derivation of BP Algorithm for Polytrees . . . 420
56.6 Example of BP algo for a Tree . . . 423
56.7 Bipartite bnets . . . 427
56.8 BP for bipartite bnets (BP-BB) . . . 428
56.8.1 BP-BB and general BP agree on Markov chains . . . 430
56.8.2 BP-BB and general BP agree on tree bnets. . . . 432
56.9 BP-BB and sum-product decomposition . . . 434

57 Message Passing in Quantum Mechanics **435**

58 Meta-learners for estimating ATE **436**

59 Missing Data, Imputation **440**
59.1 Imputation via EM . . . 441
59.2 Imputation via MCMC . . . 444
59.3 Multiple Imputations . . . 445

60 Modified Treatment Policy **446**
60.1 One time MTP . . . 446
60.2 $\Delta_{|c}$ estimand . . . 450
60.3 Estimates of $\Delta_{|c}$ . . . 453
60.3.1 Empirical estimate of $\Delta_{|c}$ . . . 453
60.3.2 OR estimate of $\Delta_{|c}$ . . . 453
60.4 Other Estimands besides $\Delta_{|c}$ . . . 456
60.5 Multi-time MTP . . . 456

61 Monty Hall Problem **459**

62 Multi-armed Bandits **461**
62.1 Bnet for MAB . . . 462
62.2 Reward functions . . . 464
62.3 Regret functions . . . 466
62.4 Strategies with random exploration . . . 467
62.4.1 ϵ-greedy algorithm . . . 467
62.4.2 ϵ_t-greedy algorithm . . . 468
62.5 Strategies with nonrandom exploration . . . 468
62.5.1 Upper Confidence Bounds (UCB) algorithms . . . 468
Frequentist UCB (UCB1) algorithm . . . 469
Bayesian UCB algorithm . . . 469
62.5.2 Thompson Sampling MAB (TS-MAB) algorithm . . . 471
Bnet for general TS-MAB algorithm . . . 471
TS-MAB algorithm with Beta agent and Bernoulli environment 472
TS-MAB algorithm, skeletal reprise . . . 473
62.5.3 Grad-MAB algorithm . . . 474

63 Naive Bayes **477**

64 Neural Networks **478**
64.1 Activation Functions $\mathcal{A}_i^\lambda : \mathbb{R} \to \mathbb{R}$ . . . 479
64.2 Weight optimization via supervised training and gradient descent . . 480
64.3 Non-dense layers . . . 482
64.4 Autoencoder NN . . . 484

65 Noisy-OR gate **485**
65.1 3 ways to interpret the parameters π_i . . . 486

66 Non-negative Matrix Factorization **490**
66.1 Bnet interpretation . . . 490
66.2 Simplest recursive algorithm . . . 491

67 Observationally Equivalent DAGs **492**
67.1 Examples . . . 492

68 Omitted Variable Bias **495**

69 Personalized Expected Utility **501**
69.1 Goal of PEU Theory . . . 502
69.2 Bnets for PEU Theory . . . 503
69.3 Bounds on *EU* for unspecified bnet . . . 503
69.4 Bounds on *EU* for specific bnet families . . . 506

70 Personalized Treatment Effects **507**
70.1 Goal, Strategy and Rationale of PTE theory . . . 508
70.2 Bnets for PTE theory . . . 510
70.3 $ATE = PB - PH$ . . . 511
70.4 Probabilities Relevant to PTE theory . . . 512
70.5 Symmetry . . . 517
70.6 Linear Programming Problem . . . 518
70.7 Special constraints . . . 519
70.8 Matrix representation of probabilities . . . 522
70.9 Bounds on Exp. Probs. imposed by Obs. Probs. . . . 525
70.10 Bounds on $PNS3$ for unspecified bnet . . . 527
70.11 Bounds on $PNS3$ for specific bnet families . . . 533
70.12 Bounds on ATE imposed by Obs. Probs. . . . 533
70.13 Bounds on PNS in terms of ATE and Obs. Probs. . . . 533
70.14 Numerical Examples . . . 534

71 Plate Notation **536**

72 Potential Outcomes and Beyond **538**
72.1 G and G_{den} bnets, the starting point bnets . . . 539
72.2 G bnet with nodes $y^\sigma(0), y^\sigma(1)$ added to it. . . . 541
72.3 Expected Values of treatment outcome y^σ . . . 543
72.4 Translation Dictionary . . . 543
72.5 $\mathcal{Y}_{|d,x} = \mathcal{Y}_{d|d,x}$ (SUTVA) . . . 544
72.6 Conditional Independence Assumption (CIA) . . . 545
72.7 Treatment Effects . . . 545
72.8 Insights into what makes treatment effects equal and $\mathcal{Y}_{1|0} = \mathcal{Y}_1$ . . . 548
72.9 G_{do+} bnet . . . 549
72.10 $ACE = ATE$ . . . 550
72.11 Good, Bad Controls . . . 551
72.12 PO Confounder Sensitivity Analysis . . . 552
72.13 (SDO, ATE) space . . . 554
72.14 Strata-Matching . . . 557
72.14.1 Exact strata-matching . . . 557
Estimates of Treatment Effects . . . 557
Example, estimation of treatment effects . . . 559
72.14.2 Approximate strata-matching . . . 561
72.14.3 Unbiased strata-matching estimates . . . 561
72.15 Propensities . . . 563
72.16 Propensity based estimates of treatment effects . . . 567
72.17 Positivity . . . 568
72.18 Multi-time PO bnets (Panel Data) . . . 569

73 Program evaluation and review technique (PERT) 573
73.1 Example . . . 575

74 Random Forest and Bagging 579
74.1 Bagging (with fully-featured bags) . . . 579
74.2 Bagging (with randomly-shortened bags) . . . 581

75 Recurrent Neural Networks 582
75.1 Language Sequence Modeling . . . 585
75.2 Other types of RNN . . . 585
75.2.1 Long Short Term Memory (LSTM) unit (1997) . . . 587
75.2.2 Gated Recurrence Unit (GRU) (2014) . . . 589

76 Regression Discontinuity Design 591
76.1 PO analysis . . . 591
76.2 Linear Regression . . . 593

77 Regularization of Loss Functions 594
77.1 L^p norm ROLF . . . 595
77.1.1 L^1 norm ROLF can lead to sparsity . . . 595
77.1.2 L^2 norm ROLF for Least Squares . . . 597
77.2 Proximal functions . . . 598
77.3 Proximal ROLF . . . 600
77.4 Unobserved Nodes of a bnet . . . 601

78 Reinforcement Learning (RL) 603
78.1 Exact RL bnet . . . 606
78.2 Actor-Critic RL bnet . . . 608
78.3 Q function learning RL bnet . . . 610

79 Reliability Box Diagrams and Fault Tree Diagrams 612
79.1 Minimal Cut Sets . . . 618

80 Restricted Boltzmann Machines 620

81 ROC curves 622
81.1 Terminology Table Adapted from Wikipedia Ref.[149] . . . 625

82 Scoring the Nodes of a Learned Bnet 627
82.1 Probability Distributions and Special Functions . . . 628
82.2 Single node with no parents . . . 630
82.3 Multiple nodes with any number of parents . . . 632
82.4 Bayesian Scores . . . 634
82.5 Information Theoretic scores . . . 634

83 Selection Bias Removal **636**
83.1 Pre and Post Switch Nodes . . . 637
83.2 Removing SB from passive query $P(y|x)$ . . . 639
83.3 Removing SB from active query $P(y|\mathcal{D}x)$ . . . 641

84 Sentence Splitting with SentenceAx **643**
84.1 Preliminary Conventions . . . 643
84.1.1 Tensor Notation . . . 643
84.1.2 PyTorch conventions . . . 644
84.2 Bayesian Network for this model . . . 649
84.3 Loss $\mathcal{L}$ for this model . . . 651

85 Shannon Information Theory **653**

86 Shapley Explainability **654**
86.0.1 Numerical examples of SHAP . . . 657

87 Simpson's Paradox **660**
87.1 Pearl Causality . . . 662
87.2 Numerical Example . . . 664

88 Structure and Parameter Learning for Bnets **665**
88.1 Overview . . . 665
88.2 Score based SL algorithms . . . 667
88.3 Constraint based SL algorithms . . . 668
88.4 Pseudo-code for some bnet learning algorithms . . . 669

89 Support Vector Machines And Kernel Method **671**
89.1 Learning Algorithm for SVM Classifier . . . 672
89.2 Linear (dot-product) Kernel . . . 673
89.3 Alternatives to Linear Kernel . . . 677
89.4 Random Forest and Kernel Method . . . 677

90 Survival Analysis **678**
90.1 $S(t)$ estimates . . . 680
90.1.1 No-censoring estimate of $S(t)$ . . . 680
90.1.2 Kaplan-Meier estimate of $S(t)$ . . . 680
90.2 $\lambda(t)$ models . . . 685
90.2.1 $\lambda(t)$ independent of covariates Z . . . 685
90.2.2 $\lambda(t)$ dependent on covariates Z . . . 686
90.3 $S_0(t)$ estimates . . . 689

91 Synthetic Controls **691**
91.1 PO analysis . . . 693

92 Targeted Estimator **695**
92.1 Goal, Strategy, and Rationale of TE theory . . . 695
92.2 Functional Calculus . . . 697
92.3 Linear Approximation of $\Psi[P_N]$ . . . 699
92.4 ATE estimand . . . 700
92.5 ATE estimates . . . 701
92.5.1 Ψ^E . . . 701
92.5.2 Ψ^G . . . 702
92.5.3 Ψ^{IPW} . . . 702
92.5.4 Ψ^{LIPW} . . . 702
92.5.5 Ψ^{LIPW++} (a.k.a. Ψ^{TMLE}) . . . 706
92.6 Ψ^{TMLE} in practice . . . 709

93 Time Series Analysis: ARMA and VAR **711**
93.1 White noise . . . 711
93.2 Backshift operator . . . 712
93.3 Metrics . . . 712
93.4 Definition of $ARMA(p,q)$, $AR(p)$ and $MA(q)$. . . . 714
93.5 Solving $AR(p)$ . . . 716
93.6 Solving $MA(q)$ . . . 717
93.7 Solving $ARMA(p,q)$ . . . 718
93.8 Auto-correlation and partial auto-correlation . . . 718
93.9 Generating function of auto-correlation . . . 722
93.10Impulse Response . . . 723
93.11$AR(p)$ and Yule-Walker equations . . . 724
93.12Forecasting . . . 726
93.13Model Learning . . . 732
93.14Differencing and $ARIMA(p,d,q)$ . . . 732
93.15Parameter Learning . . . 736
93.15.1PL of $AR(p)$ . . . 736
93.15.2PL of $MA(q)$ . . . 738
93.15.3PL of $ARMA(p,q)$ . . . 741
93.16$VAR(p)$ . . . 742

94 Transfer Learning **743**

95 Transformer Networks **745**
95.1 Tensor Notation . . . 746
95.2 Recurrent Neural Net with Attention . . . 747
95.2.1 Single Head Attention . . . 747
95.2.2 Multi-Head Attention . . . 750
95.3 Vanilla tranet . . . 752
95.3.1 Single Head Attention . . . 757

95.3.2 Multi-Head Attention . . . 758
95.3.3 Encoder . . . 760
95.3.4 Decoder . . . 761
95.4 BERT . . . 764

96 Transportability of Causal Knowledge 765

97 Turbo Codes 769
97.1 Decoding Algorithm . . . 772
97.2 Message Passing Interpretation of Decoding Algorithm . . . 774

98 Uplift Modelling 775
98.1 UP types . . . 775
98.2 Some Relevant Technical Formulas from Chapter 72 . . . 776
98.3 UP Analysis . . . 777
98.4 UP Decision Trees . . . 779
98.4.1 Appendix, connection between Δ_c and $\Delta_{c|j}$ . . . 784

99 Variational Bayesian Approximation for Medical Diagnosis 785

100 Variational Bayesian Approximation via D_{KL} 789
100.1Free Energy $\mathcal{F}(\vec{x})$ . . . 791

101 XGBoost 794
101.1Divergences . . . 794
101.2Minimizing Cost function for single tree . . . 796
101.3Leaf Splitting . . . 799
101.4Pruning . . . 799
101.5Feature Binning . . . 801
101.6Final estimate of target attribute . . . 801
101.7Bnet for XGBoost . . . 802

102 Zero Information Transmission (Graphoid Axioms) 804
102.1Consequences of Eq.(102.5) . . . 805

Bibliography 807

Appendices

Chapter 84

Sentence Splitting with SentenceAx

The Openie6 (O6) software (at github repo Ref.[2]) splits complex or compound sentences into simple ones.[1] This is a necessary step in doing DAG extraction from text (DEFT) (See Chapter 14).

The O6 software is described by its creators in the paper Ref.[30], which we will henceforth refer to as the O6 paper.

The SentenceAx (Sax) software (at github repo Ref.[78]) is a complete re-write of the O6 software. Sax is 95% identical algorithmically to O6, but it has been rewritten in what we hope is a friendlier form.

Sax is a fine tuning of the BERT model. What this means in the language of Bayesian Networks is that we use BERT as a prior probability. The BERT model is the encoder part of the vanilla Transformer network which we discuss in Chapter 95.

This chapter describes the technical aspects of Sax. Although this chapter can be read without reading the O6 paper, we highly recommend to our readers that they read the O6 paper also. Some parts of this chapter are taken almost verbatim from the O6 paper. Other parts try to fill-in gaps in the explanations provided by the O6 paper or to improve those explanations. Yet others parts describe small changes that we made to the O6 software, in an effort to improve its clarity.

In this chapter, we will use the Numpy-like tensor notation discussed in Section C.48. In particular, note that $[n] = [0:n] = \{0, 1, \ldots, n-1\}$ and that $T^{[n],[m]}$ is an $n \times m$ matrix.

84.1 Preliminary Conventions

84.1.1 Tensor Notation

Our tensor notation is discussed in Section C.48. Here is a quick review of some of the more salient facts in that section on tensors. Below, we will often accompany

[1] Simple sentences are essentially the same as the triples (subject, relationship, object) which when visualized as a directed or undirected graph is called a "knowledge graph".

an equation in tensor component notation with the equivalent matrix equation, in parenthesis.

We use Greek letters for tensor indices.

Let $\alpha \in [a]$, $\beta \in [b]$, $\gamma \in [c]$, $\delta \in [d]$, $\nu \in [n]$, $\Delta \in [D]$.

- **reshaping**

$$T^{\nu,\delta} \to T^{\Delta} \quad \left(T^{[n_{\underline{h}}],[d]} \to T^{[D]}\right) \tag{84.1}$$

$$T^{\Delta} \to T^{\nu,\delta} \quad \left(T^{[D]} \to T^{[n_{\underline{h}}],[d]}\right) \tag{84.2}$$

- **concatenation**

$$T^{[n]} = (T^0, T^1, \ldots, T^{n-1}) = (T^{\nu})_{\nu \in [n]} \tag{84.3}$$

- **Hadamard product (element-wise, entry-wise multiplication)**

$$T^{[n]} * S^{[n]} = (T^{\nu} S^{\nu})_{\nu \in [n]} \tag{84.4}$$

- **Matrix multiplication**

$T^{[n]} = T^{[n],[1]}$ is a column vector.

$$(T^{[n]})^T S^{[n]} = \text{scalar} \tag{84.5}$$

$$T^{[a],[b]} S^{[b],[c]} = \left[\sum_{\beta \in [b]} T^{\alpha,\beta} S^{\beta,\gamma}\right]_{\alpha \in [a], \gamma \in [c]} \tag{84.6}$$

Most treatments of Transformer Networks (tranets), including the the O6 paper and PyTorch, order the operations chronologically from left to right (L2R). So if A occurs before B, they write AB. This is contrary to what is done in Linear Algebra, where one orders the operations chronologically from right to left (R2L), and one writes BA. In Chapter 95 on tranets, we followed the Linear Algebra convention. In this chapter, we will follow the PyTorch convention, because Sax is written with PyTorch so it uses the PyTorch convention.

84.1.2 PyTorch conventions

- **Linear**

Some pseudo-code

```
lin = nn.Linear(b, a)
y^[n],[b] = lin(x^[n],[a])
```

In the L2R (left to right) convention followed by PyTorch

$$x^{\nu,[a]} \to y^{\nu,[b]} = x^{\nu,[a]} W^{[a],[b]} \tag{84.7}$$

for all $\nu \in [n]$. Alternatively, in the R2L convention followed in Linear Algebra,

$$x^{[a],\nu} \to y^{[b],\nu} = W^{[b],[a]} x^{[a],\nu} \tag{84.8}$$

Note that in PyTorch, the rightmost index of the input is the one that is summed over.

The weights matrix $W^{[b],[a]}$ is learned by training.

- **Dropout**

  Some pseudo-code

```
dropo = nn.Dropout(p_drop)
y^[n],[a] = dropo(x^[n],[a])
```

$$x^{\nu,[a]} \to y^{\nu,[a]} = x^{\nu,[a]} \mathcal{W}_R^{[a],[a]} \quad \text{(in L2R)} \tag{84.9}$$

$$x^{\nu,[a]} \to y^{\nu,[a]} = \mathcal{W}_L^{[a],[a]} x^{\nu,[a]} \quad \text{(in R2L)} \tag{84.10}$$

for all $\nu \in [n]$.

`Dropout` learns a weight matrix W just like `Linear`. But at the end of the training, `Dropout` flips a coin for each row of $W_L^{[a],[a]}$ (resp., column of $W_R^{[a],[a]}$), with

$$P(Heads) = p_{drop}, \text{ and } P(Tails) = 1 - p_{drop} = p_{keep}.$$

If the coin lands on T, it keeps that row of $W_L^{[a],[a]}$ (resp., column of $W_R^{[a],[a]}$), whereas if lands on H, it sets that row (resp., column) to zero. Then the matrix is divided by p_{keep}. The final matrix after all these operations is denoted by $\mathcal{W}_L$ (resp., $\mathcal{W}_R$).

- **Embedding**

  Some pseudo-code

```
emb = nn.Embedding(num_embeddings=L, embedding_dim=d)
Y^[n1],[n2],[d] = emb(λ^[n1],[n2])
```

In Sax, we use $L = 100$ and $d = 768$ (for BERT base). The d is the "hidden dimension" of BERT. The L could be as large as the vocab size of BERT, but since we consider only sentences with 100 words at most, we may set $L = 100$. L doesn't appear in the final answer because we sum over $\lambda \in [L]$.

Next, we explain in more detail the meaning of the tensors λ and Y.

Let

$L =$ number of embeddings

$d =$ embedding dimension

$\lambda \in [L]$, $\alpha \in [\ell]$, $\nu_1 \in [n_1]$, $\nu_2 \in [n_2]$

$\ell = \nu_1 \nu_2$.

Consider matrices Y, E, X such that

$$Y^{\delta,\alpha} = \sum_{\lambda} E^{\delta,\lambda} X^{\lambda,\alpha} \quad \left(Y^{[d],[\ell]} = E^{[d],[L]} X^{[L],[\ell]}\right) \tag{84.11}$$

Assume that matrix X has 1-hot columns

$$X^{\lambda,\alpha} = \delta(\lambda, \lambda(\alpha)) \tag{84.12}$$

where $\lambda() : [\ell] \to [L]$.

Hence,

$$Y^{\delta,\alpha} = E^{\delta,\lambda(\alpha)} \tag{84.13}$$

If we define

$$\Lambda^{\alpha} = \lambda(\alpha) \tag{84.14}$$

then `emb()` maps

$$\Lambda^{\alpha} \to Y^{\delta,\alpha} = E^{\delta,\lambda(\alpha)} \ \ (\Lambda^{[\ell]} \to Y^{[d],[\ell]}) \tag{84.15}$$

Now replace α by (ν_1, ν_2). `emb()` also maps

$$\Lambda^{\nu_1,\nu_2} \to Y^{\delta,\nu_1,\nu_2} = E^{\delta,\lambda(\nu_1,\nu_2)} \ \ (\Lambda^{[n_1],[n_2]} \to Y^{[d],[n_1],[n_2]}) \tag{84.16}$$

Actually, `emb()` orders the tensor indices of the output so that the δ index is on the right side rather than the left side of the input indices. Thus,

$$Y^{[n_1],[n_2],[d]} = \texttt{emb}(\Lambda^{[n_1],[n_2]}) \tag{84.17}$$

- **Cross Entropy Loss**

  Some pseudo-code

  ```
  loss = nn.CrossEntropyLoss()
  output = loss(input=x^[n_c],[n_s], target=t^[n_s])
  ```

  Cross Entropy in Information Theory:

$$
\begin{aligned}
H(P^{\sigma}_{tar}, P^{\sigma}_{in}) &= -\sum_{\gamma \in [n_{\underline{c}}]} P_{tar}(\gamma|\sigma) \ln P_{in}(\gamma|\sigma) && (84.18)\\
&= -\sum_{\gamma \in [n_{\underline{c}}]} P_{tar}(\gamma|\sigma) \ln \left[\frac{P_{in}(\gamma|\sigma)}{P_{tar}(\gamma|\sigma)} P_{tar}(\gamma|\sigma) \right] && (84.19)\\
&= H(P^{\sigma}_{tar}) + D_{KL}(P^{\sigma}_{in} \parallel P^{\sigma}_{tar}) && (84.20)
\end{aligned}
$$

  Cross Entropy Loss in PyTorch:

  Let

  $n_{\underline{s}}$ = total number of samples being considered (usually batch size). $\sigma \in [n_{\underline{s}}]$

  $n_{\underline{c}}$ = number of classes in classification. $\gamma \in [n_{\underline{c}}]$

  $x^{[n_{\underline{c}}],[n_{\underline{s}}]}$ = input samples

  $t^{[n_{\underline{s}}]}$ = target samples

  Define

$$
\begin{aligned}
P_{in}(\gamma|\sigma) &= \frac{\exp(x^{\gamma,\sigma})}{\sum_{\gamma' \in [n_{\underline{c}}]} \exp(x^{\gamma',\sigma})} && (84.21)\\
&= \text{softmax}(x^{[n_{\underline{c}}],\sigma})(\gamma|\sigma) && (84.22)
\end{aligned}
$$

  Suppose $W^{\gamma} : values(\underline{t}) \to [0,1]$ for all $\gamma \in [n_{\underline{c}}]$.

  Define

$$
P_{tar}(\gamma|\sigma) = \frac{W^{\gamma}(t^{\sigma}) \mathbb{1}(t^{\sigma} \neq -100)}{\sum_{\gamma \in [n_{\underline{c}}]} numerator} \tag{84.23}
$$

  The -100 on the right side of the last equation can be replaced by any other integer in $values(\underline{t})$ for which we want the loss to be zero (for example, it could be an integer used for padding)

  Now define the cross entropy loss $\mathcal{L}_{CE}$ by

$$\mathcal{L}^{\sigma}_{CE}(t^{\sigma}, x^{[n_{\underline{c}}],\sigma}) = H(P_{tar}(\cdot|\sigma), P_{in}(\cdot|\sigma)) \tag{84.24}$$

$$\mathcal{L}_{CE} = \frac{1}{n_{\underline{s}}} \sum_{\sigma \in [n_{\underline{s}}]} \mathcal{L}^{\sigma}_{CE} \tag{84.25}$$

For example, if $W^{\gamma} = 1$, and $n_{\underline{c}} = 2$,

$$\mathcal{L}_{CE} \;=\; \frac{1}{n_{\underline{s}}} \sum_{\sigma \in [n_{\underline{s}}]} [P_{tar}(0|\sigma) \ln P_{in}(0|\sigma) + P_{tar}(1|\sigma) \ln P_{in}(1|\sigma)] \tag{84.26}$$

- **unsqueeze-repeat-gather**

  Some pseudo-code

```
lll_loc = ll_loc0.unsqueeze(2).\
    repeat(1, 1, lll_state.shape[2])
lll_out = torch.gather(
    input=lll_state, dim=1, index=lll_loc)
```

  Sax uses the trio of operations unsqueeze-repeat-gather in the manner of the above pseudo-code. We have already discussed in Section C.48 how each of these 3 operations acts individually. Here we will discuss how they act jointly, when used as in the above pseudo-code.

  Let

  `lll_state=` $S^{[s_{ba}],[a],[d]}$

  `ll_loc0=` $L_0^{[s_{ba}],[a]}$

  `lll_loc=` $L^{[s_{ba}],[b],[d]}$

  `lll_out=` $O^{[s_{ba}],[b],[d]}$

  $\sigma \in s_{ba}, \alpha \in [a], \beta \in [b], \delta \in [d]$

  `unsqueeze(2)` takes

$$L_0^{[s_{ba}],[a]} \rightarrow L_0^{[s_{ba}],[a],0} \tag{84.27}$$

  `lll_state.shape[2]` equals d, and `repeat(1, 1, d)` takes

$$L_0^{[s_{ba}],[a],0} \rightarrow L^{[s_{ba}],[a],[d]} = (\underbrace{L_0^{[s_{ba}],[a],0}, \ldots, L_0^{[s_{ba}],[a],0}}_{d \text{ times}}) \tag{84.28}$$

  Now define

$$\lambda(\sigma,\alpha) = L^{\sigma,\alpha,\delta} = L_0^{\sigma,\alpha} \tag{84.29}$$

Then the `gather()` with `dim=1` outputs.

$$O^{\sigma,\alpha,\delta} = S^{\sigma,\lambda(\sigma,\alpha),\delta} \tag{84.30}$$

84.2 Bayesian Network for this model

Let

$\ell_{pad} = 86$, padding length, for this batch

$\ell_{enc} = 121$, encoded length, for this batch, $\ell_{enc} \geq \ell_{pad}$

$n_{dep} = 5$, number of copies of solid box connected in series, number of depths

$n_{att} = 2$, number of copies of dashed box connected in series, number of iterative (attention) layers.

$d = 768$, hidden dimension per head

$n_{\underline{h}}$, number of heads (BERT base)

$D = dn_{\underline{h}}$, hidden dimension for all heads

$s_{ba} = 24$, batch size

$n_{il} = 6$, number of ilabels

$d_{me} = 300$, merge dimension

Fig.84.1 shows the bnet for Sax.[2]. The structural equations, printed in blue, for that bnet, are as follows.

$\underline{a}^{[86]}$: `ll_greedy_ilabel`

$\underline{B}^{[121],[768]}$: `lll_hidstate`

$\underline{d}^{[121],[768]}$: `lll_hidstate`

$\underline{E}^{[86],[768]}$: `lll_pred_code`

$\underline{G}^{[86],[768]}$: `lll_word_hidstate`

$\underline{I}^{[121],[768]}$: `lll_hidstate`

$\underline{L}^{[86],[6]}$: `lll_word_score`

$\underline{M}^{[86],[300]}$: `lll_word_hidstate`

$\underline{S}^{[86],[768]}$: `lll_word_hidstate`

$\underline{X}^{[86],[6]}$: `lll_word_score`

$$\begin{aligned} a^{[86]} &= \text{argmax}(X^{[86],[6]}; dim = -1) \\ &: \texttt{ll\_greedy\_ilabel} \end{aligned} \tag{84.31a}$$

$$\begin{aligned} B^{[121],[768]} &= \text{BERT}() \\ &: \texttt{lll\_hidstate} \end{aligned} \tag{84.31b}$$

[2]The bnet of Fig.84.1 was drawn with the help of the texnn software (Ref.[80])

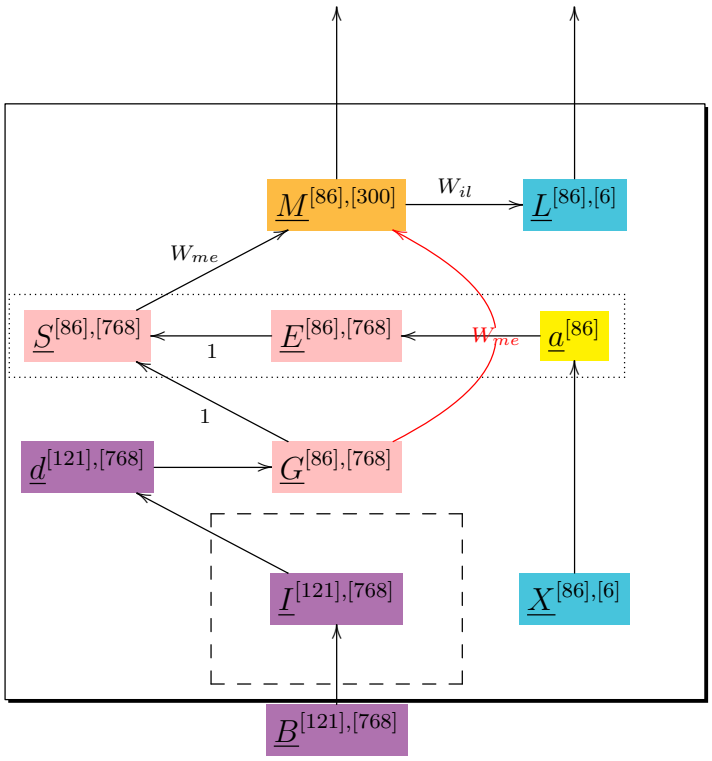


Figure 84.1: Sax bnet. 2 copies of dashed box are connected in series. 5 copies (5 depths) of plain box are connected in series. However, in the first of those 5 plain box copies, the dotted box is omitted and node $\underline{G}$ feeds directly into node $\underline{M}$ (indicated by red arrow). We display the tensor shape superscripts in the PyTorch L2R order. All tensor shape superscripts have been simplified by omitting a $[s_{ba}]$ from their left side, where $s_{ba} = 24$ is the batch size.

$$d^{[121],[768]} = \text{dropout}(I^{[121],[768]}) \quad : \texttt{lll\_hidstate} \tag{84.31c}$$

$$E^{[86],[768]} = \text{embedding}(a^{[86]}) \quad : \texttt{lll\_pred\_code} \tag{84.31d}$$

$$G^{[86],[768]} = \text{gather}(d^{[121],[768]}; dim = -2) \quad : \texttt{lll\_word\_hidstate} \tag{84.31e}$$

$$I^{[121],[768]} = \left[B^{[121],[768]}\mathbb{1}(depth = 0) + M^{[86],[300]}\mathbb{1}(depth > 0)\right] \\ : \texttt{lll\_hidstate} \tag{84.31f}$$

$$L^{[86],[6]} = M^{[86],[300]}W_{il}^{[300],[6]} \\ : \texttt{lll\_word\_score} \tag{84.31g}$$

$$M^{[86],[300]} = \left[G^{[86],[768]}\mathbb{1}(depth = 0) + S^{[86],[768]}\mathbb{1}(depth > 0)\right] W_{me}^{[768],[300]} \\ : \texttt{lll\_word\_hidstate} \tag{84.31h}$$

$$S^{[86],[768]} = E^{[86],[768]} + G^{[86],[768]} \\ : \texttt{lll\_word\_hidstate} \tag{84.31i}$$

$$X^{[86],[6]} = L^{[86],[6]}\mathbb{1}(depth > 0) \\ : \texttt{lll\_word\_score} \tag{84.31j}$$

84.3 Loss $\mathcal{L}$ for this model

The Loss $\mathcal{L}$ is the sum of the Cross Entropy Loss $\mathcal{L}_{CE}$ and 4 penalty losses $\mathcal{L}_i$ for $i \in PL$ where $PL = \{POSC, HVC, HVE, EC\}$.

$$\mathcal{L} = \mathcal{L}_{CE} + \sum_{i \in PL} \lambda_i \mathcal{L}_i \tag{84.32}$$

where the λ_i are hyper-parameters to be determined heuristically.

In an earlier section, we discussed the Cross Entropy Loss at length. In this section, we will discuss the 4 penalty losses.

Below, we will use the standard notation for the **positive-part function** (a.k.a. the **reLU** function)

$$(x)_+ = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases} \tag{84.33}$$

$$= \max(0, x) \tag{84.34}$$

Since loss is supposed to be bounded below (usually it is defined to be greater or equal to zero), the positive-part function comes in handy when defining a loss.

Let

ℓ = number of words, length of sentence. $\alpha \in [\ell]$

M = number of depths. $\mu \in [M]$

w^α = word at position α

$T_{pos} = \{N, V, JJ, RB\}$, POS tags, POS=Part Of Speech, N=Noun, V=Verb, JJ=Adjective, RB=Adverb

$T_{ex} = \{S, R, O, N\}$[3]. extraction tags (extags), S=Subject, R=Relation, O=Object, N=None

$T_{ex} \backslash N = T_{ex} - \{N\}$

$POS^\alpha \in T_{pos}$, Part Of Sentence of w^α.

Importance indicator function.

$$IMP^\alpha = \mathbb{1}(POS^\alpha \in T_{pos}) \tag{84.35}$$

Head verb indicator function. A **head verb** is any verb that isn't a **light verb** (do, be, is, has, etc.).

$$HV^\alpha = \mathbb{1}(w^\alpha \text{ is a head verb}) \tag{84.36}$$

Let $P(\underline{t}^{\mu,\alpha} = t)$ denote an empirical probability for a table element $\underline{t}^{\mu,\alpha} \in T_{ex}$, for all $\mu \in [M]$ and $\alpha \in [\ell]$.

The O6 paper uses the following sentence to exemplify the 4 types of penalty losses.

Obama gained popularity after Oprah endorsed him for the presidency.

Henceforth, we will refer to this sentence as the red-sent.

For the red-sent, the head verbs are gained, endorsed

Two valid extractions from red-sent are: (Obama; gained; popularity) and (Oprah; endorsed him for; the presidency).

1. **Part of Speech Coverage (POSC)**

   Penalize if some important words do not belong to at least one extraction.

   In red-sent: all the words Obama, gained, popularity, Oprah, endorsed, presidency must be covered in the set of extractions.

$$\mathcal{L}_{POSC} = \sum_{\alpha \in [\ell]} IMP^\alpha P_{POSC}(\alpha) \tag{84.37}$$

[3]The Sax software uses a different set for T_{ex} than $T_{ex} = \{S, R, O, N\}$. In Sax, we use for T_{ex} the list `BASE_EXTAGS` (defined globally in the file `sax_globals`.) In `BASE_EXTAGS`, N becomes NONE (or 0) and R becomes REL (or 3). Also note that 2 tranets are trained by Sax, one for extraction (task=ex), and one for splitting (task=cc). For task=cc, T_{ex} is replaced by T_{cc}. In Sax, we use for T_{cc} the list `BASE_CCTAGS` (defined globally in the file `sax_globals`.) In `BASE_CCTAGS`, N becomes NONE (or 0) and R becomes CC (or 3).

$$P_{POSC}(\alpha) = 1 - \max_{\mu\in[M]}\max_{t\in T_{ex}\setminus N} P(\underline{t}^{\mu,\alpha} = t) \tag{84.38}$$

2. **Head Verb Coverage (HVC)**

   Penalize if a head verb is not present in the relation (R) of any extraction.

   In red-sent: (Obama; gained; popularity), (Obama; gained; presidency) is not a comprehensive set of extractions.

$$\mathcal{L}_{HVC} = \sum_{\alpha\in[\ell]} HV^{\alpha} P_{HVC}(\alpha) \tag{84.39}$$

$$P_{HVC}(\alpha) = \left| 1 - \sum_{\mu\in[M]} P(\underline{t}^{\mu,\alpha} = R) \right| \tag{84.40}$$

3. **Head Verb Exclusivity (HVE)**

   Penalize extractions that contain more than one head verb in their relation (R).

   In red-sent: gained popularity after Oprah endorsed is not a good relation as it contains two head verbs

$$\mathcal{L}_{HVE} = \sum_{\mu\in[M]} \left(\sum_{\alpha\in[\ell]} HV^{\alpha} P(\underline{t}^{\mu,\alpha} = R) - 1 \right)_{+} \tag{84.41}$$

4. **Extraction Count (EC)**

   Penalize if the total number of extractions with head verbs in the relation (R) is too small; i.e., it is smaller than the number of head verbs in the sentence.

$$\mathcal{L}_{EC} = \left(\sum_{\alpha\in[\ell]} HV^{\alpha} - \sum_{\mu\in[M]} EC^{\mu} \right)_{+} \tag{84.42}$$

$$EC^{\mu} = \max_{\alpha\in[\ell]} HV^{\alpha} P(\underline{t}^{\mu,\alpha} = R) \tag{84.43}$$

Bibliography

[1] Alan Agresti. *An introduction to categorical data analysis*. John Wiley & Sons, 2018.

[2] Data Analytics and IIT Delhi Intelligence Research (DAIR) Group. Openie6. https://github.com/dair-iitd/openie6.

[3] 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.

[4] Dan Bendel. Metropolis-Hastings: A comprehensive overview and proof. https://similarweb.engineering/mcmc/.

[5] David Benkeser and Antoine Chambaz. A ride in targeted learning territory. https://achambaz.github.io/tlride/tlride-book.pdf.

[6] Alexandra M Carvalho. Scoring functions for learning Bayesian networks. http://www.lx.it.pt/~asmc/pub/talks/09-TA/ta_pres.pdf.

[7] Bo Chang. Copula: a very short introduction, article in Bo's Blog. https://bochang.me/blog/posts/copula/.

[8] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. https://arxiv.org/abs/1603.02754.

[9] Victor Chernozhukov, Carlos Cinelli, Whitney Newey, Amit Sharma, and Vasilis Syrgkanis. Long story short: Omitted variable bias in causal machine learning. https://www.nber.org/system/files/working_papers/w30302/w30302.pdf.

[10] 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.

[11] Carlos Cinelli and Chad Hazlett. Making sense of sensitivity: Extending omitted variable bias. https://carloscinelli.com/files/Cinelli%20and%20Hazlett%20(2020)%20-%20Making%20Sense%20of%20Sensitivity.pdf.

[12] Scott Cunningham. *Causal inference: The mixtape*. Yale University Press, 2021. https://mixtape.scunning.com/index.html.

[13] Robin J. Evans. Graphical methods for inequality constraints in marginalized DAGs. https://arxiv.org/abs/1209.2978.

[14] Matheus Facure Alves. *Causal Inference for The Brave and True*. 2021. https://matheusfacure.github.io/python-causality-handbook/landing-page.html.

[15] 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.

[16] 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".

[17] 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.

[18] 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.

[19] Pierre Gutierrez and Jean-Yves Gérardy. 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.

[20] James Douglas Hamilton. *Time series analysis*. Princeton University Press, 2020.

[21] Sebastian Haneuse and Andrea Rotnitzky. Estimation of the effect of interventions that modify the received treatment. *Statistics in medicine*, 32(30):5260–5277, 2013. Main:https://sci-hub.se/10.1002/sim.5907, Supplement: https://onlinelibrary.wiley.com/action/downloadSupplement?doi=10.1002%2Fsim.5907&file=sim5907-sup-0001-Appendix.pdf.

[22] Christina Heinze-Deml. Causality, spring semester 2019 at ETH Zurich. https://stat.ethz.ch/lectures/ss19/causality.php#course_materials.

[23] MA Hernán and J Robins. Causal inference: What if. Boca Raton: Chapman & Hill/CRC, https://www.hsph.harvard.edu/miguel-hernan/causal-inference-book/.

[24] Katherine Hoffman. An illustrated guide to modified treatment policies, part 1: Introduction and motivation, article in KHstats blog. https://www.khstats.com/blog/lmtp/lmtp/.

[25] Katherine Hoffman. An illustrated guide to TMLE, article in KHstats blog. https://www.khstats.com/blog/tmle/tutorial/.

[26] 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.

[27] Tommi S. Jaakkola and Michael I. Jordan. Variational probabilistic inference and the QMR-DT network. http://arxiv.org/abs/1105.5462.

[28] 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.

[29] Chaitanya K. Joshi. Transformer (machine learning model). https://graphdeeplearning.github.io/post/transformers-are-gnns/.

[30] Keshav Kolluru, Vaibhav Adlakha, Samarth Aggarwal, Mausam, and Soumen Chakrabarti. Openie6: Iterative grid labeling and coordination analysis for open information extraction. https://arxiv.org/abs/2010.03147.

[31] 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.

[32] 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.

[33] Sergey Levine. Course CS 285 at UC Berkeley, Deep reinforcement learning. http://rail.eecs.berkeley.edu/deeprlcourse/.

[34] Ang Li. Ph.D. Thesis, UCLA 2021. https://ftp.cs.ucla.edu/pub/stat_ser/r507.pdf.

[35] Dimitris Margaritis. Learning Bayesian network model structure from data (thesis, 2003, Carnegie Mellon Univ). https://apps.dtic.mil/sti/citations/ADA461103.

[36] 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.

[37] 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.

[38] 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.

[39] Robert J. McEliece, David J. C. MacKay, and Jung-Fu Cheng. Turbo decoding as an instance of Pearl's belief propagation algorithm. http://authors.library.caltech.edu/6938/1/MCEieeejstc98.pdf.

[40] Scott Mueller, Ang Li, and Judea Pearl. Causes of effects: Learning individual responses from population data. *arXiv preprint arXiv:2104.13730*, 2021. https://arxiv.org/abs/2104.13730.

[41] Brady Neal. Introduction to causal inference, Fall 2020, lectures and book. https://www.bradyneal.com/causal-inference-course.

[42] Richard E Neapolitan. *Learning Bayesian networks*. Pearson Prentice Hall, 2004.

[43] Andrew Ng. Lecture at deeplearning.ai on recurrent neural networks. http://www.ar-tiste.com/ng-lec-rnn.pdf.

[44] Gregory Nuel. Tutorial on exact belief propagation in Bayesian networks: from messages to algorithms. https://arxiv.org/abs/1201.4724.

[45] paperspace.com. PyTorch 101, Part 1: Understanding graphs, automatic differentiation and autograd. https://blog.paperspace.com/pytorch-101-understanding-graphs-and-automatic-differentiation/.

[46] Judea Pearl. Linear models: A useful microscope for causal analysis. https://ftp.cs.ucla.edu/pub/stat_ser/r409-corrected-reprint.pdf.

[47] Judea Pearl. Mediating instrumental variables. https://ftp.cs.ucla.edu/pub/stat_ser/r210.pdf.

[48] Judea Pearl. On the testability of causal models with latent and instrumental variables. https://arxiv.org/abs/1302.4976.

[49] Judea Pearl. Reverend Bayes on inference engines: A distributed hierarchical approach. https://www.aaai.org/Papers/AAAI/1982/AAAI82-032.pdf, 1982.

[50] Judea Pearl. *Probabilistic Inference in Intelligent Systems*. Morgan Kaufmann, 1988.

[51] 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.

[52] Judea Pearl. *Causality: Models, Reasoning, and Inference, Second Edition*. Cambridge University Press, 2013.

[53] Judea Pearl. Causal and counterfactual inference. *The Handbook of Rationality*, pages 1–41, 2019. https://ftp.cs.ucla.edu/pub/stat_ser/r485.pdf.

[54] 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.

[55] Judea Pearl, Madelyn Glymour, and Nicholas P Jewell. *Causal inference in statistics: A primer*. John Wiley & Sons, 2016.

[56] Judea Pearl and Dana Mackenzie. *The book of why: the new science of cause and effect*. Basic Books, 2018.

[57] Judea Pearl and James M Robins. Probabilistic evaluation of sequential plans from causal models with hidden variables. *arXiv preprint arXiv:1302.4977*, 2013. https://arxiv.org/abs/1302.4977.

[58] Ashwin Rao and Tikhon Jelvis. *Foundations of Reinforcement Learning with Applications in Finance*. https://stanford.edu/~ashlearn/RLForFinanceBook/book.pdf.

[59] ReliaSoft. System analysis reference. http://reliawiki.org/index.php/System_Analysis_Reference.

[60] 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.

[61] Scholarpedia. Granger causality. http://www.scholarpedia.org/article/Granger_causality.

[62] Marco Scutari. bnlearn. https://www.bnlearn.com/.

[63] 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.

[64] 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.

[65] StatQuest. XGBoost Parts 1-4 (videos). https://statquest.org/video-index/.

[66] Russell Stuart and Norvig Peter. Artificial intelligence-a modern approach 3rd ed, 2016.

[67] 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/.

[68] theinvestorsbook.com. Pert analysis. https://theinvestorsbook.com/pert-analysis.html.

[69] Jin Tian and Judea Pearl. Probabilities of causation: Bounds and identification. *Annals of Mathematics and Artificial Intelligence*, 28(1):287–313, 2000. https://ftp.cs.ucla.edu/pub/stat_ser/r271-A.pdf.

[70] Robert R. Tucci. Bayesian networks (a.k.a. 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/.

[71] 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/.

[72] Robert R. Tucci. Goodness of causal fit. https://github.com/rrtucci/DAG_Lie_Detector.

[73] Robert R. Tucci. JudeasRx. https://github.com/rrtucci/JudeasRx.

[74] Robert R. Tucci. Mappa Mundi. https://github.com/rrtucci/mappa_mundi.

[75] Robert R. Tucci. Quantum d-separation and quantum belief propagation. https://arxiv.org/abs/2012.09635.

[76] Robert R. Tucci. Quantum Fog. https://github.com/artiste-qb-net/quantum-fog.

[77] Robert R. Tucci. SCuMpy. https://github.com/rrtucci/scumpy.

[78] Robert R. Tucci. SentenceAx. https://github.com/rrtucci/SentenceAx.

[79] Robert R. Tucci. Shannon information theory without shedding tears over delta & epsilon proofs or typical sequences. https://arxiv.org/abs/1208.2737.

[80] Robert R. Tucci. texnn. https://github.com/rrtucci/texnn.

[81] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. https://arxiv.org/abs/1706.03762.

[82] 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/.

[83] Martin J Wainwright and Michael I Jordan. Graphical models, exponential families, and variational inference (book). https://people.eecs.berkeley.edu/~wainwrig/Papers/WaiJor08_FTML.pdf.

[84] 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.

[85] Lilian Weng. What are diffusion models? lilianweng.github.io/lil-log, https://lilianweng.github.io/posts/2021-07-11-diffusion-models/, 2021.

[86] Wikibooks. Control systems. https://en.wikibooks.org/wiki/Control_Systems.

[87] Wikipedia. AdaBoost. https://en.wikipedia.org/wiki/AdaBoost.

[88] Wikipedia. Autoregressive model. https://en.wikipedia.org/wiki/Autoregressive_model.

[89] Wikipedia. Autoregressive moving-average model. https://en.wikipedia.org/wiki/Autoregressive%E2%80%93moving-average_model.

[90] Wikipedia. Baum-Welsh algorithm. https://en.wikipedia.org/wiki/Baum%E2%80%93Welch_algorithm.

[91] Wikipedia. Belief propagation. https://en.wikipedia.org/wiki/Belief_propagation.

[92] Wikipedia. Berkson's paradox. https://en.wikipedia.org/wiki/Berkson%27s_paradox.

[93] Wikipedia. Bernoulli distribution. https://en.wikipedia.org/wiki/Bernoulli_distribution.

[94] Wikipedia. Beta distribution. https://en.wikipedia.org/wiki/Beta_distribution.

[95] Wikipedia. Beta function. https://en.wikipedia.org/wiki/Beta_function.

[96] Wikipedia. Binary decision diagram. https://en.wikipedia.org/wiki/Binary_decision_diagram.

[97] Wikipedia. Boolean algebra. https://en.wikipedia.org/wiki/Boolean_algebra.

[98] Wikipedia. Bootstrap aggregating. https://en.wikipedia.org/wiki/Bootstrap_aggregating.

[99] Wikipedia. Categorical distribution. https://en.wikipedia.org/wiki/Categorical_distribution.

[100] Wikipedia. Chi-square distribution. https://en.wikipedia.org/wiki/Chi-square_distribution.

[101] Wikipedia. Chow-Liu tree. https://en.wikipedia.org/wiki/Chow%E2%80%93Liu_tree.

[102] Wikipedia. Cochran's theorem. https://en.wikipedia.org/wiki/Cochran%27s_theorem.

[103] Wikipedia. Conjugate prior. https://en.wikipedia.org/wiki/Conjugate_prior.

[104] Wikipedia. Copula. https://en.wikipedia.org/wiki/Copula_(probability_theory).

[105] Wikipedia. Cramer-Rao bound. https://en.wikipedia.org/wiki/Cram%C3%A9r%E2%80%93Rao_bound.

[106] Wikipedia. Cross-validation. https://en.wikipedia.org/wiki/Cross-validation_(statistics).

[107] Wikipedia. Data processing inequality. https://en.wikipedia.org/wiki/Data_processing_inequality.

[108] Wikipedia. Dirichlet distribution. https://en.wikipedia.org/wiki/Dirichlet_distribution.

[109] Wikipedia. Errors in variables models. https://en.wikipedia.org/wiki/Errors-in-variables_models.

[110] Wikipedia. Expectation maximization. https://en.wikipedia.org/wiki/Expectation%E2%80%93maximization_algorithm.

[111] Wikipedia. F-distribution. https://en.wikipedia.org/wiki/F-distribution.

[112] Wikipedia. Frisch-Waugh-Lovell theorem. https://en.wikipedia.org/wiki/Frisch%E2%80%93Waugh%E2%80%93Lovell_theorem.

[113] Wikipedia. Functional derivative. https://en.wikipedia.org/wiki/Functional_derivative.

[114] Wikipedia. Gamma distribution. https://en.wikipedia.org/wiki/Gamma_distribution.

[115] Wikipedia. Gamma function. https://en.wikipedia.org/wiki/Gamma_function.

[116] Wikipedia. Gated recurrent unit. https://en.wikipedia.org/wiki/Gated_recurrent_unit.

[117] Wikipedia. Gibbs sampling. https://en.wikipedia.org/wiki/Gibbs_sampling.

[118] Wikipedia. Granger causality. https://en.wikipedia.org/wiki/Granger_causality.

[119] Wikipedia. Hidden Markov model. https://en.wikipedia.org/wiki/Hidden_Markov_model.

[120] Wikipedia. Hoeffding's inequality. https://en.wikipedia.org/wiki/Hoeffding%27s_inequality.

[121] Wikipedia. Importance sampling. https://en.wikipedia.org/wiki/Importance_sampling.

[122] Wikipedia. Instrumental variables estimation. https://en.wikipedia.org/wiki/Instrumental_variables_estimation.

[123] Wikipedia. Inverse transform sampling. https://en.wikipedia.org/wiki/Inverse_transform_sampling.

[124] Wikipedia. Jackknife resampling. https://en.wikipedia.org/wiki/Cross-validation_(statistics).

[125] Wikipedia. Junction tree algorithm. https://en.wikipedia.org/wiki/Junction_tree_algorithm.

[126] Wikipedia. k-means clustering. https://en.wikipedia.org/wiki/K-means_clustering.

[127] Wikipedia. Kalman filter. https://en.wikipedia.org/wiki/Kalman_filter.

[128] Wikipedia. Kernel method. https://en.wikipedia.org/wiki/Kernel_method.

[129] Wikipedia. Kernel perceptron. https://en.wikipedia.org/wiki/Kernel_perceptron.

[130] Wikipedia. Laplace transform. https://en.wikipedia.org/wiki/Laplace_transform.

[131] Wikipedia. Least squares. https://en.wikipedia.org/wiki/Least_squares.

[132] Wikipedia. Legendre transformation. https://en.wikipedia.org/wiki/Legendre_transformation.

[133] Wikipedia. Likelihood-ratio test. https://en.wikipedia.org/wiki/Likelihood-ratio_test.

[134] Wikipedia. Linear regression. https://en.wikipedia.org/wiki/Linear_regression.

[135] Wikipedia. Long short term memory. https://en.wikipedia.org/wiki/Long_short-term_memory.

[136] Wikipedia. Markov blanket. https://en.wikipedia.org/wiki/Markov_blanket.

[137] Wikipedia. Metropolis-Hastings method. https://en.wikipedia.org/wiki/Metropolis%E2%80%93Hastings_algorithm.

[138] Wikipedia. Minimum spanning tree. https://en.wikipedia.org/wiki/Minimum_spanning_tree.

[139] Wikipedia. Monte Carlo methods. https://en.wikipedia.org/wiki/Category:Monte_Carlo_methods.

[140] Wikipedia. Moving-average model. https://en.wikipedia.org/wiki/Moving-average_model.

[141] Wikipedia. Multinomial distribution. https://en.wikipedia.org/wiki/Multinomial_distribution.

[142] Wikipedia. Multinomial theorem. https://en.wikipedia.org/wiki/Multinomial_theorem.

[143] Wikipedia. Multivariate normal distribution. https://en.wikipedia.org/wiki/Multivariate_normal_distribution.

[144] Wikipedia. Natural experiment. https://en.wikipedia.org/wiki/Natural_experiment.

[145] Wikipedia. Non-negative matrix factorization. https://en.wikipedia.org/wiki/Non-negative_matrix_factorization.

[146] Wikipedia. Ordinary least squares. https://en.wikipedia.org/wiki/Ordinary_least_squares.

[147] Wikipedia. Program evaluation and review technique. https://en.wikipedia.org/wiki/Program_evaluation_and_review_technique.

[148] Wikipedia. Random forest. https://en.wikipedia.org/wiki/Random_forest.

[149] Wikipedia. Receiver operating characteristic. https://en.wikipedia.org/wiki/Receiver_operating_characteristic.

[150] Wikipedia. Rejection sampling. https://en.wikipedia.org/wiki/Rejection_sampling.

[151] Wikipedia. Score test. https://en.wikipedia.org/wiki/Score_test.

[152] Wikipedia. Signal flow graph. https://en.wikipedia.org/wiki/Signal-flow_graph.

[153] Wikipedia. Simple linear regression. https://en.wikipedia.org/wiki/Simple_linear_regression.

[154] Wikipedia. Simpson's paradox. https://en.wikipedia.org/wiki/Simpson's_paradox.

[155] Wikipedia. Spring system. https://en.wikipedia.org/wiki/Spring_system.

[156] Wikipedia. Student's t-distribution. https://en.wikipedia.org/wiki/Student%27s_t-distribution.

[157] Wikipedia. Support vector machine. https://en.wikipedia.org/wiki/Support-vector_machine.

[158] Wikipedia. Survival analysis. https://en.wikipedia.org/wiki/Survival_analysis.

[159] Wikipedia. Time series. https://en.wikipedia.org/wiki/Time_series.

[160] Wikipedia. Transfer learning. https://en.wikipedia.org/wiki/Transfer_learning.

[161] Wikipedia. Transformer (machine learning model). https://en.wikipedia.org/wiki/Transformer_(machine_learning_model).

[162] Wikipedia. Uplift modelling. https://en.wikipedia.org/wiki/Uplift_modelling.

[163] Wikipedia. Variational Bayesian methods. https://en.wikipedia.org/wiki/Variational_Bayesian_methods.

[164] Wikipedia. Vector autoregression. https://en.wikipedia.org/wiki/Vector_autoregression.

[165] Wikipedia. Viterbi algorithm. https://en.wikipedia.org/wiki/Viterbi_algorithm.

[166] Wikipedia. Wald test. https://en.wikipedia.org/wiki/Wald_test.

[167] Wikipedia. Z-transform. https://en.wikipedia.org/wiki/Z-transform.

[168] 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.

[169] Ronghui (Lily) Xu. Lecture notes, MATH 284, Spring 2020, Survival analysis. https://mathweb.ucsd.edu/~rxu/math284/.

[170] Jonathan S Yedidia, William T Freeman, and Yair Weiss. Understanding belief propagation and its generalizations, Mitsubishi Technical Report tr-2001-22. https://merl.com/publications/docs/TR2001-22.pdf.