Bayesuvius,
a small visual dictionary of Bayesian Networks

Robert R. Tucci
www.ar-tiste.xyz

July 10, 2020

Figure 1: View of Mount Vesuvius from Pompeii

Figure 2: Mount Vesuvius and Bay of Naples

Contents

0.1 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 B net. 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 B nets (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

0.2 Notational Conventions

bnet=B net=Bayesian Network

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 $A = \mathbb{Z}, \mathbb{R}$.

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.

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

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

$$P_{\underline{x},\underline{y}}(x,y) = P(\underline{x} = x, \underline{y} = y) = P(x,y) \tag{2}$$

$$P_{\underline{x}|\underline{y}}(x|y) = P(\underline{x} = x|\underline{y} = y) = P(x|y) = \frac{P(x,y)}{P(y)} \tag{3}$$

Kronecker delta function: For x, y in discrete set S,

$$\delta(x,y) = \begin{cases} 1 \text{ if } x = y \\ 0 \text{ if } x \neq y \end{cases} \tag{4}$$

Dirac delta function: For $x, y \in \mathbb{R}$,

$$\int_{-\infty}^{+\infty} dx\ \delta(x-y) f(x) = f(y) \tag{5}$$

Transition probability matrix 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 node by sums over new states, and Dirac delta functions by Kronecker delta functions. More precisely, consider a function $f : S \to \mathbb{R}$. Let $S_{\underline{x}} \subset S$ and $S \to S_{\underline{x}}$ upon discretization (binning). Then

$$\int_S dx\ P_{\underline{x}}(x) f(x) \to \frac{1}{n_{\underline{x}}} \sum_{x \in S_{\underline{x}}} f(x)\ . \tag{6}$$

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

$$\int_S dx\ \delta(x-y) f(x) = f(y) \to \sum_{x \in S_{\underline{x}}} \delta(x,y) f(x) = f(y)\ . \tag{7}$$

Indicator function:

$$\hat{1}(\mathcal{S}) = \begin{cases} 1 \text{ if } \mathcal{S} \text{ is true} \\ 0 \text{ if } \mathcal{S} \text{ is false} \end{cases} \tag{8}$$

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

$$\vec{x} = (x[0], x[1], x[2] \ldots, x[nsam(\vec{x}) - 1]) = x[:] \tag{9}$$

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

$$x[i] \sim P_{\underline{x}} \text{ (i.e. } P_{\underline{x[i]}} = P_{\underline{x}}) \tag{10}$$

$$P(\underline{x} = x) = \frac{1}{nsam(\vec{x})} \sum_i \hat{1}(x[i] = x) \tag{11}$$

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.

$$P(\vec{x}) = \prod_i P(x[i]) \tag{12}$$

$$\sum_{\vec{x}} = \prod_i \sum_{x[i]} \tag{13}$$

$$\partial_{\vec{x}} = [\partial_{x[0]}, \partial_{x[1]}, \partial_{x[2]}, \ldots, \partial_{x[nsam(\vec{x})-1]}] \tag{14}$$

$$\begin{aligned} P(\vec{x}) &\approx [\prod_x P(x)^{P(x)}]^{nsam(\vec{x})} && (15)\\ &= e^{nsam(\vec{x}) \sum_x P(x) \log P(x)} && (16)\\ &= e^{-nsam(\vec{x}) H(P_{\underline{x}})} && (17) \end{aligned}$$

$$f^{[1,\partial_x,\partial_y]}(x,y) = [f, \partial_x f, \partial_y f] \tag{18}$$

$$f^+ = f^{[1,\partial_x,\partial_y]} \tag{19}$$

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

- Entropy:

$$H(p) = -\sum_x p(x) \log p(x) \geq 0 \tag{20}$$

- Kullback-Liebler divergence:

$$D_{KL}(p \parallel q) = \sum_x p(x) \log \frac{p(x)}{q(x)} \geq 0 \tag{21}$$

- Cross entropy:

$$\begin{aligned} CE(p \to q) &= -\sum_x p(x) \log q(x) && (22)\\ &= H(p) + D_{KL}(p \parallel q) && (23) \end{aligned}$$

Normal Distribution: $x, \mu, \sigma \in \mathbb{R}$, $\sigma > 0$

$$\mathcal{N}(\mu, \sigma^2)(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{24}$$

Uniform Distribution: $a < b$, $x \in [a, b]$

$$\mathcal{U}(a, b)(x) = \frac{1}{b-a} \tag{25}$$

Expected Value

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

$$E_{\underline{x}}[f(\underline{x})] = E_{x \sim P(x)}[f(x)] = \sum_x P(x) f(x) \tag{26}$$

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

$$E_{\underline{x}|\underline{y}}[f(\underline{x}, \underline{y})] = \sum_x P(x|\underline{y}) f(x, \underline{y}) \ , \tag{27}$$

$$E_{\underline{x}|\underline{y}=y}[f(\underline{x}, y)] = E_{\underline{x}|y}[f(\underline{x}, y)] = \sum_x P(x|y) f(x, y) \ . \tag{28}$$

Note that

$$\begin{aligned} E_{\underline{y}}[E_{\underline{x}|\underline{y}}[f(\underline{x}, \underline{y})]] &= \sum_{x,y} P(x|y) P(y) f(x, y) && (29) \\ &= \sum_{x,y} P(x, y) f(x, y) && (30) \\ &= E_{\underline{x},\underline{y}}[f(\underline{x}, \underline{y})] \ . && (31) \end{aligned}$$

Chapter 1

Basic Curve Fitting Using Gradient Descent

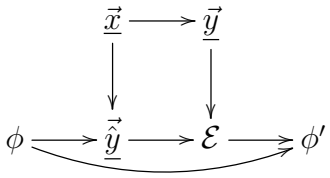

Figure 1.1: Basic curve fitting bnet.

Samples $(x[i], y[i]) \in S_{\underline{x}} \times S_{\underline{y}}$ 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

$$P_{\underline{x},\underline{y}}(x,y) = \frac{1}{nsam(\vec{x})} \sum_i \hat{1}(x = x[i], y = y[i]) \ . \tag{1.1}$$

Let

$$\mathcal{E}(\vec{x},\vec{y},\phi) = \frac{1}{nsam(\vec{y})} \sum_i |y[i] - \hat{y}(x[i];\phi)|^2 \tag{1.2}$$

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

Best fit is parameters ϕ^* such that

$$\phi^* = \text{argmin}_\phi \mathcal{E}(\vec{x},\vec{y},\phi) \ . \tag{1.3}$$

The node transition matrices for the basic curve fitting bnet Fig.1.1 are printed below in blue.

$$P(\phi) = \text{given} \ . \tag{1.4}$$

On the first time, $P(\phi)$ is arbitrary. After the first time, it is determined by previous stage.

$$P(\vec{x}) = \prod_i P_{\underline{x}}(x[i]) \tag{1.5}$$

$$P(\vec{y}|\vec{x}) = \prod_i P_{\underline{y}|\underline{x}}(y[i] \mid x[i]) \tag{1.6}$$

$$P(\hat{y}[i]|\phi, \vec{x}) = \delta(\hat{y}[i], \hat{y}(x[i]; \phi)) \tag{1.7}$$

$$P(\mathcal{E}|\vec{\hat{y}}, \vec{y}) = \delta(\mathcal{E}, \frac{1}{nsam(\vec{x})} \sum_i |y[i] - \hat{y}[i]|^2) \ . \tag{1.8}$$

$$P(\phi'|\phi, \mathcal{E}) = \delta(\phi', \phi - \eta \partial_\phi \mathcal{E}) \tag{1.9}$$

$\eta > 0$ 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 2

Bell and Clauser-Horne Inequalities in Quantum Mechanics

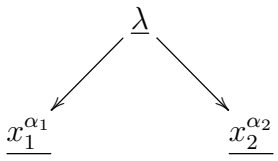

Figure 2.1: bnet used to discuss Bell and Clauser-Horne inequalities in Quantum Mechanics.

I wrote a post about this in 2008 for my blog "Quantum Bayesian Networks". See Ref.[1].

Chapter 3

Generative Adversarial Network (GAN)

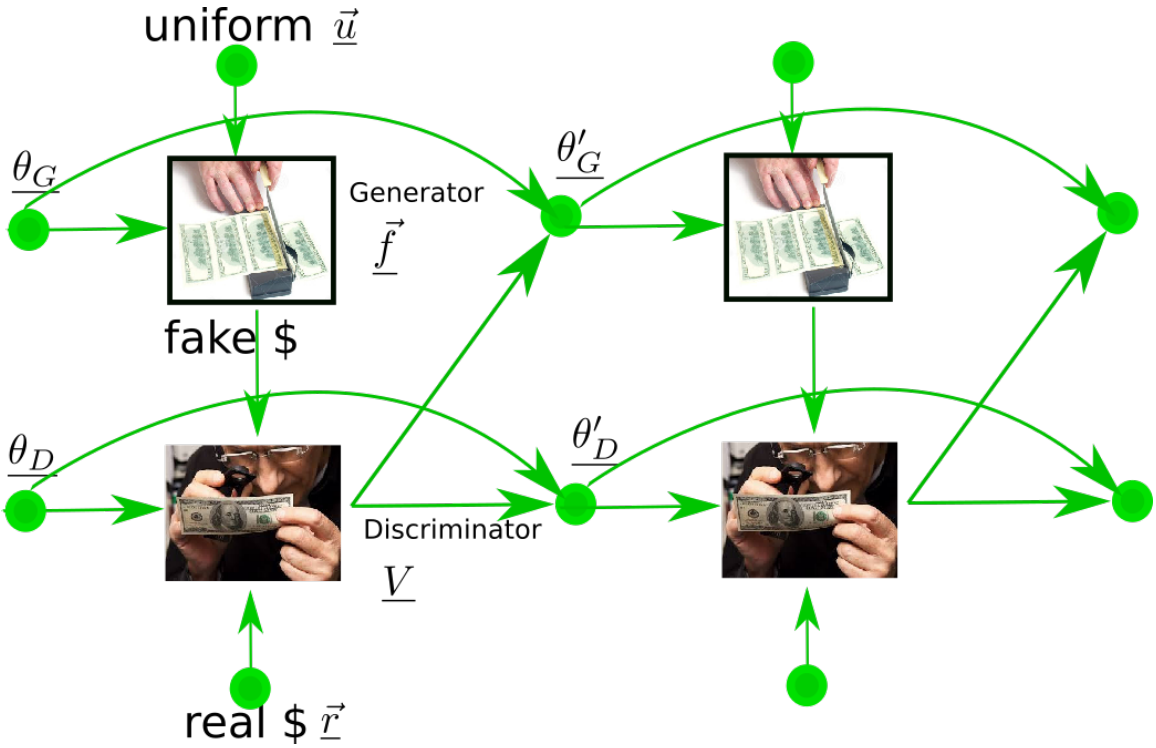

Figure 3.1: Generative Adversarial Network (GAN)

Original GAN, Ref.[2](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 veredict $V \in [0, 1]$. G depends on parameter θ_G and D on parameter θ_D. Veredict V and initial θ_G, θ_D are used to get new parameters θ'_G, θ'_D.Process is repeated (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.

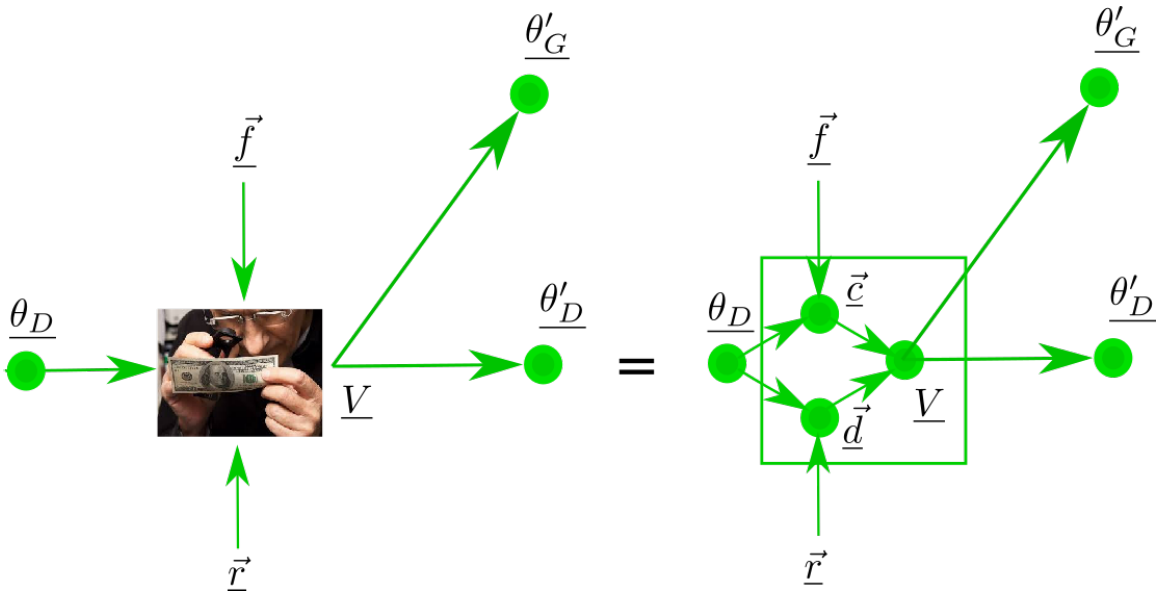

Figure 3.2: Discriminator node $\underline{V}$ in Fig.3.1 can be split into 3 nodes $\underline{\vec{c}}$, $\underline{\vec{d}}$ and $\underline{V}$.

Let $\mathcal{D}$ be the domain of $D(\cdot, \theta_D)$. Assume that for any $x \in \mathcal{D}$,

$$0 \leq D(x, \theta_D) \leq 1 \ . \tag{3.1}$$

For any $S \subset \mathcal{D}$, define

$$\sum_{x \in S} D(x, \theta_D) = \lambda(S, \theta_D) \ . \tag{3.2}$$

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

$$\overline{D}(f, \theta_D) = 1 - D(f, \theta_D) \ . \tag{3.3}$$

Note that

$$0 \leq \overline{D}(f, \theta_D) \leq 1 \ . \tag{3.4}$$

Define:

$$V(\theta_G, \theta_D) = \sum_r P(r) \log D(r, \theta_D) + \sum_u P(u) \log \overline{D}(G(u, \theta_G), \theta_D) \ . \tag{3.5}$$

We want the first variation of $V(\theta_G, \theta_D)$ to vanish.

$$\delta V(\theta_G, \theta_D) = 0 \ . \tag{3.6}$$

This implies

$$\partial_{\theta_G} V(\theta_G, \theta_D) = \partial_{\theta_D} V(\theta_G, \theta_D) = 0 \tag{3.7}$$

and

$$V_{opt} = \min_{\theta_G} \max_{\theta_D} V(\theta_G, \theta_D) \ . \tag{3.8}$$

Node transition probability matrices for Figs.3.1 and 3.2 are given next in blue:

$$P(\theta_G) = \text{ given} \tag{3.9}$$

$$P(\theta_D) = \text{ given} \tag{3.10}$$

$$P(\vec{u}) = \prod_i P(u[i]) \ \text{ (usually uniform distribution)} \tag{3.11}$$

$$P(\vec{r}) = \prod_i P(r[i]) \tag{3.12}$$

$$P(f[i] \mid \vec{u}, \theta_G) = \delta[f[i], G(u[i], \theta_G)] \tag{3.13}$$

$$P(c[i] \mid \vec{f}, \theta_D) = \delta(c[i], \overline{D}(f[i], \theta_D)) \tag{3.14}$$

$$P(d[j] \mid \vec{r}, \theta_D) = \delta(d[j], D(r[j], \theta_D)) \tag{3.15}$$

$$P(V|\vec{d}, \vec{c}) = \delta(V, \frac{1}{N} \log \prod_{i,j} (c[i]d[j])) \tag{3.16}$$

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.

$$P(\theta'_G|V, \theta_G) = \delta(\theta'_G, \theta_G - \eta_G \partial_{\theta_G} V) \tag{3.17}$$

$$P(\theta'_D|V, \theta_D) = \delta(\theta'_D, \theta_D + \eta_D \partial_{\theta_D} V) \tag{3.18}$$

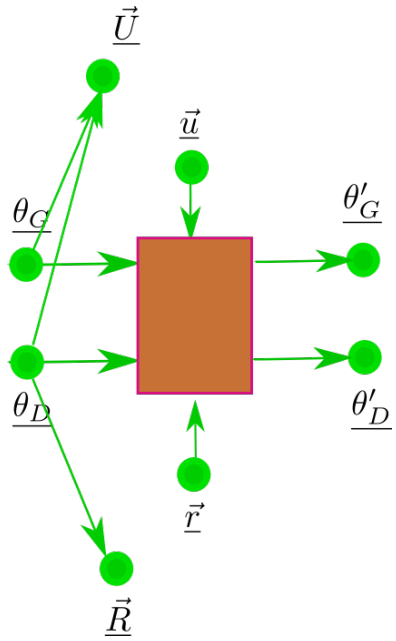

Figure 3.3: GAN, Constraining Bayesian Network

Constraining B net given in Fig.3.3. It adds 2 new nodes, namely $\underline{\vec{U}}$ and $\underline{\vec{R}}$, to the bnet of Fig.3.1. The purpose of these 2 barren (childrenless) nodes is to constrain certain functions to be probability distributions.

Node transition probabilities for the 2 new nodes given next in blue.

$$P(U[i] \mid \theta_G) = \frac{\overline{D}(G(U[i], \theta_G), \theta_D))}{\overline{\lambda}(\theta_G, \theta_D)} \tag{3.19}$$

where $S_{\underline{U[i]}} = S_{\underline{u}}$ and $\overline{\lambda}(\theta_G, \theta_D) = \sum_u \overline{D}(G(u, \theta_G), \theta_D))$.

$$P(R[i] \mid \theta_G, \theta_D) = \frac{D(R[i], \theta_D)}{\lambda(\theta_D)} \tag{3.20}$$

where $S_{\underline{R[i]}} = S_{\underline{r}}$ and $\lambda(\theta_D) = \sum_r D(r, \theta_D)$.

$$P(V|\vec{u}, \vec{r}) = \delta(V, \frac{1}{N} \log \prod_{i,j} (P(\underline{R[i]} = r[i] \mid \theta_G, \theta_D) P(\underline{U[i]} = u[j] \mid \theta_G))) \tag{3.21}$$

where $N = nsam(\vec{r}) nsam(\vec{u})$.

$\mathcal{L}$ = likelihood

$$\mathcal{L} = P(\vec{r}, \vec{u}|\theta_G, \theta_D) \tag{3.22}$$

$$= \prod_{i,j} \left[\frac{D(r[i], \theta_D)}{\lambda(\theta_D)} \frac{\overline{D}(G(u[j], \theta_G), \theta_D))}{\overline{\lambda}(\theta_G, \theta_D)} \right] \tag{3.23}$$

$$\log \mathcal{L} = N[V(\theta_G, \theta_D) - \log \lambda(\theta_D) - \log \overline{\lambda}(\theta_G, \theta_D)] \tag{3.24}$$

Chapter 4

Kalman Filter

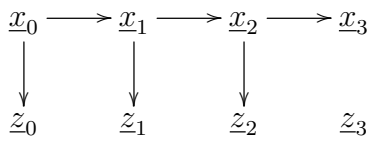

Figure 4.1: Kalman Filter bnet.

Let $t = 0, 1, 2, \ldots, T-1$.
$\underline{x}_t \in S_{\underline{x}}$ are random variables that represent the hidden (unobserved) true state of the system.
$\underline{z}_t \in S_{\underline{z}}$ are random variables that represent the measured (observed) state of the system.
The Kalman Filter bnet Fig.4.1 has the following node probability transition matrices, printed in blue:

$$P(x_t|x_{t-1}) = \mathcal{N}(F_t x_{t-1} + B_t u_t, Q_t) \ , \tag{4.1}$$

where F_t, Q_t, B_t, u_t are given. $P(x_t|x_{t-1})$ becomes $P(x_t)$ for $t = 0$.

$$P(z_t|x_t) = \mathcal{N}(H_t x_t, R_t) \ , \tag{4.2}$$

where H_t, R_t are given.
Define

$$\underline{Z}_t = (\underline{z}_{t'})_{t' \le t} \ . \tag{4.3}$$

Define $\hat{x}_t$ and P_t by

$$P(x_t|Z_t) = \mathcal{N}(\hat{x}_t, P_t) \ . \tag{4.4}$$

Problem: Find $\hat{x}_t$ and P_t in terms of

1. current (at time t) given values of F, Q, H, R, B, u

2. current (at time t) observed value of z

3. prior (previous) value (at time $t-1$) of $\hat{x}$ and P.

See Fig.4.2. For that figure,

$$P(\hat{x}_t, P_t | z_t, \hat{x}_{t-1}, P_{t-1}) = \delta(\hat{x}_t, ?)\delta(P_t, ?) \ . \tag{4.5}$$

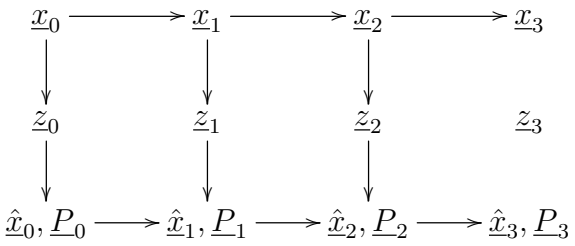

Figure 4.2: Kalman Filter bnet with deterministic nodes for $\hat{x}_t, P_t$.

Solution copied from Wikipedia Ref.[3]:

Define $\eta_{t|t} = \eta_t$ for $\eta = \hat{x}, P$.

=== Predict ===

Predicted (a priori) state estimate

$$\hat{x}_{t|t-1} = F_t \hat{x}_{t-1|t-1} + B_t u_t \tag{4.6}$$

Predicted (a priori) estimate covariance

$$P_{t|t-1} = F_t P_{t-1|t-1} F_t^{\mathsf{T}} + Q_t \tag{4.7}$$

=== Update ===

Innovation (or measurement pre-fit residual)

$$\tilde{y}_{t|t-1} = z_t - H_t \hat{x}_{t|t-1} \tag{4.8}$$

Innovation (or pre-fit residual) covariance

$$S_t = H_t P_{t|t-1} H_t^{\mathsf{T}} + R_t \tag{4.9}$$

Optimal Kalman gain

$$K_t = P_{t|t-1} H_t^{\mathsf{T}} S_t^{-1} \tag{4.10}$$

Updated (a posteriori) state estimate

$$\hat{x}_{t|t} = \hat{x}_{t|t-1} + K_t \tilde{y}_t \tag{4.11}$$

Updated (a posteriori) estimate covariance

$$P_{t|t} = (I - K_t H_t)\, P_{t|t-1} \tag{4.12}$$

Measurement post-fit residual

$$\tilde{y}_{t|t} = z_t - H_t \hat{x}_{t|t} \tag{4.13}$$

Chapter 5

Linear and Logistic Regression

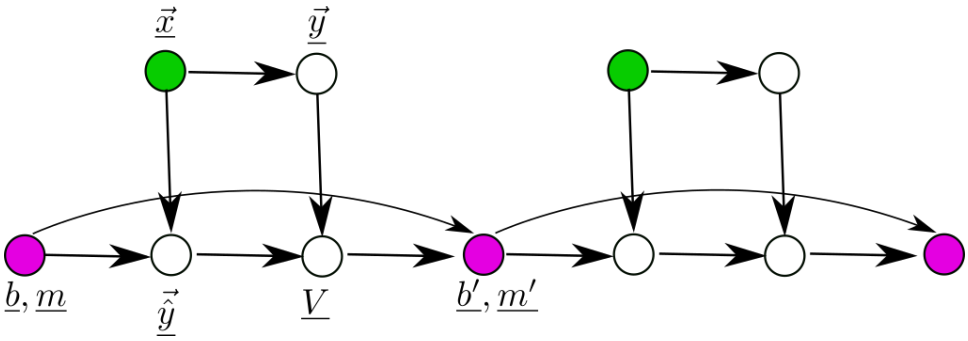

Figure 5.1: Linear Regression

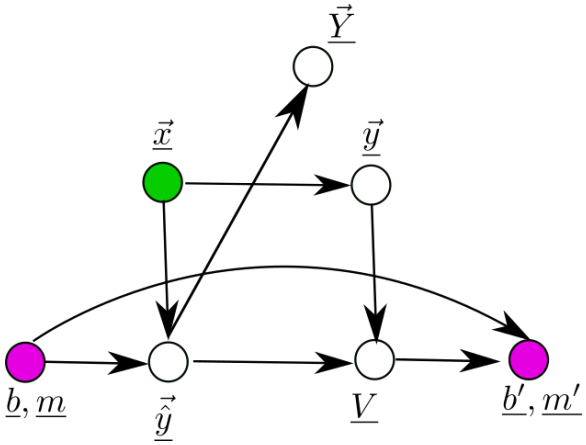

Figure 5.2: B net of Fig.5.1 with new $\vec{Y}$ node.

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 a straight line with y-intercept b and slope m.

$$\hat{y}(x; b, m) = b + mx \tag{5.1}$$

- **Logistic Regression:** $y \in \{0,1\}$. Note $\hat{y} \in [0,1]$. $(x, \hat{y}(x))$ is a sigmoid. Often in literature, b, m are replaced by β_0, β_1.

$$\hat{y}(x; b, m) = \frac{1}{1 + e^{-(b+mx)}} \tag{5.2}$$

Define

$$V(b, m) = \sum_{x,y} P(x, y)|y - \hat{y}(x; b, m)|^2 \ . \tag{5.3}$$

We want to minimize $V(b, m)$ (called a cost or loss function) wrt b and m.

Node transition probabilities of B net of Fig.5.1 given next in blue.

$$P(b, m) = \text{given} \tag{5.4}$$

On the first time, $P(b, m)$ is arbitrary. After the first time, it is determined by previous stage.

Let

$$P_{\underline{x},\underline{y}}(x, y) = \frac{1}{nsam(\vec{x})} \sum_i \hat{1}(x = x[i], y = y[i]) \ . \tag{5.5}$$

$$P(\vec{x}) = \prod_i P(x[i]) \tag{5.6}$$

$$P(\vec{y}|\vec{x}) = \prod_i P(y[i] \mid x[i]) \tag{5.7}$$

$$P(\vec{\hat{y}}|\vec{x}, b, m) = \prod_i \delta(\hat{y}[i], \hat{y}(x[i], b, m)) \tag{5.8}$$

$$P(V|\vec{\hat{y}}, \vec{y}) = \delta(V, \frac{1}{nsam(\vec{x})} \sum_i |y[i] - \hat{y}[i]|^2) \tag{5.9}$$

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".

$$P(b'|V, b) = \delta(b', b - \eta_b \partial_b V) \tag{5.10}$$

$$P(m'|V, m) = \delta(m', m - \eta_m \partial_m V) \tag{5.11}$$

Generalization to x with multiple components(features)

Suppose that for each sample i, instead of $x[i]$ being a scalar, it has n components called features:

$$x[i] = (x_0[i], x_1[i], x_2[i], \ldots x_{n-1}[i]) \ . \tag{5.12}$$

Slope m is replaced by weights

$$w = (w_0, w_1, w_3, , \ldots, w_{n-1}) \ , \tag{5.13}$$

and the product of 2 scalars $mx[i]$ is replaced by the inner vector product $w^T x[i]$.

Alternative $V(b, m)$ for logistic regression

For logistic regression, since $y[i] \in \{0, 1\}$ and $\hat{y}[i] \in [0, 1]$ are both in the interval $[0, 1]$, they can be interpreted as probabilities. Define probability distributions $p[i](x)$ and $\hat{p}[i](x)$ for $x \in \{0, 1\}$ by

$$p[i](1) = y[i], \quad p[i](0) = 1 - y[i] \tag{5.14}$$

$$\hat{p}[i](1) = \hat{y}[i], \quad \hat{p}[i](0) = 1 - \hat{y}[i] \tag{5.15}$$

Then for logistic regression, the following 2 cost functions $V(b, m)$ can be used as alternatives to the cost function Eq.(5.3) previously given.

$$V(b, m) = \frac{1}{nsam(\vec{x})} \sum_i D_{KL}(p[i] \parallel \hat{p}[i]) \tag{5.16}$$

and

$$\begin{aligned} V(b, m) &= \frac{1}{nsam(\vec{x})} \sum_i CE(p[i] \to \hat{p}[i]) && (5.17)\\ &= \frac{-1}{nsam(\vec{x})} \sum_i \{y[i] \log \hat{y}[i] + (1 - y[i]) \log(1 - \hat{y}[i])\} && (5.18)\\ &= \frac{-1}{nsam(\vec{x})} \sum_i \log \left\{\hat{y}[i]^{y[i]} (1 - \hat{y}[i])^{(1-y[i])}\right\} && (5.19)\\ &= \frac{-1}{nsam(\vec{x})} \sum_i \log P(\underline{Y} = y[i] \mid \hat{y} = \hat{y}[i]) && (5.20)\\ &= -\sum_{x,y} P(x, y) \log P(\underline{Y} = y | \hat{y} = \hat{y}(x, b, m)) && (5.21) \end{aligned}$$

Above, we used

$$P(\underline{Y} = Y | \hat{y}) = \hat{y}^Y [1 - \hat{y}]^{1-Y} \tag{5.22}$$

for $Y \in S_{\underline{Y}} = \{0, 1\}$. (Bernoulli distribution).

There is no node corresponding to $\underline{Y}$ in the B net of Fig.5.1. Fig.5.2 shows a new B net that has a new node called $\underline{\vec{Y}}$ compared to the B net of Fig.5.1. One defines the transition probabilities for all nodes of Fig.5.2 except $\underline{\vec{Y}}$ and $\underline{V}$ the same as for Fig.5.1. For $\underline{\vec{Y}}$ and $\underline{V}$, one defines

$$P(Y[i] \mid \vec{\hat{y}}) = P(\underline{Y} = Y[i] \mid \hat{y}[i]) \tag{5.23}$$

$$P(V|\vec{Y}, \vec{y}) = \delta(V, \frac{-1}{nsam(\vec{x})} \log \mathcal{L}) \ , \tag{5.24}$$

where $\mathcal{L} = \prod_i P(\underline{Y} = y[i] \mid \hat{y}[i])$=likelihood.

Chapter 6

Monty Hall Problem

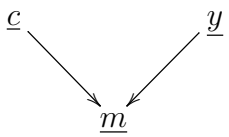

Figure 6.1: Monty Hall Problem.

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.6.1, 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:

$$P(c) = \frac{1}{3} \text{ for all } c \tag{6.1}$$

$$P(y) = \frac{1}{3} \text{ for all } y \tag{6.2}$$

$$P(m|c, y) = \hat{1}(m \neq c) \left[\frac{1}{2} \hat{1}(y = c) + \hat{1}(y \neq c) \hat{1}(m \neq y) \right] \tag{6.3}$$

It's easy to show that the above node probabilities imply that

$$P(c = 1|m = 2, y = 1) = \frac{1}{3} \tag{6.4}$$

$$P(c = 3|m = 2, y = 1) = \frac{2}{3} \tag{6.5}$$

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 7

Naive Bayes

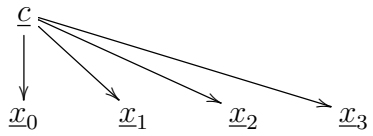

Figure 7.1: bnet for Naive Bayes with 4 features

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

$$x. = [x_0, x_1, \ldots, x_{F-1}] \; . \tag{7.1}$$

For the bnet of Fig.7.1,

$$P(c, x.) = P(c) \prod_{i=0}^{F-1} P(x_i|c) \; . \tag{7.2}$$

Given $x.$ values, find most likely class $c \in S_{\underline{c}}$.
Maximum a Posteriori (MAP) estimate:

$$\begin{aligned} c^* &= \text{argmax}_c P(c|x.) && (7.3) \\ &= \text{argmax}_c \frac{P(c, x.)}{P(x.)} && (7.4) \\ &= \text{argmax}_c P(c, x.) \; . && (7.5) \end{aligned}$$

Chapter 8

Reinforcement Learning (RL)

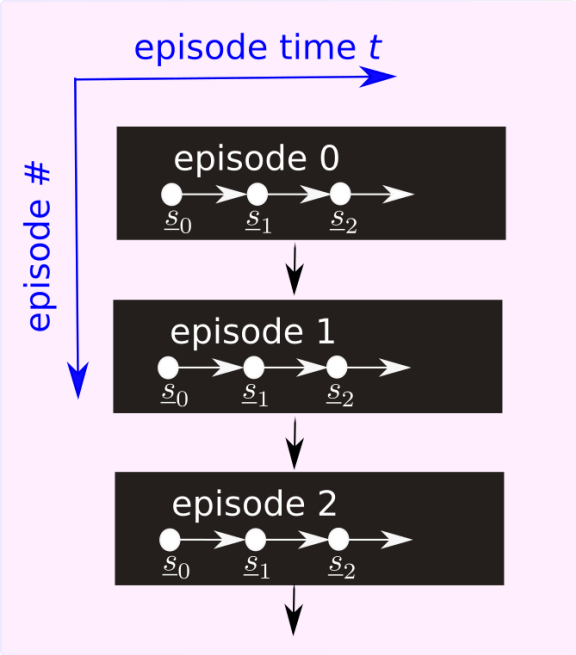

Figure 8.1: Axes for episode time and episode number.

I based this chapter on the following references. Refs.[4][5]

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.8.1.

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.

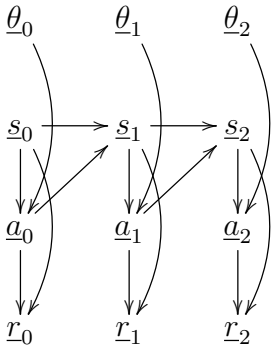

Figure 8.2: State-Action-Reward dynamical bnet

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

$$\underline{X}_\cdot = [\underline{X}_0, \underline{X}_1, \ldots, \underline{X}_{T-1}] \; . \tag{8.1}$$

Also let

$$\underline{X}_{\geq t} = [\underline{X}_t, \underline{X}_{t+1}, \ldots, \underline{X}_{T-1}] \; . \tag{8.2}$$

Fig.8.2 shows the basic State-Action-Reward bnet for an agent that is learning. The transition probabilities for the nodes of Fig.8.2 are given in blue below:

$$P(a_t|s_t, \theta_t) = \text{given.} \tag{8.3}$$

$P(a_t|s_t, \theta_t)$ is called a **policy with parameter θ_t.**

$$P(s_t|s_{t-1}, a_{t-1}) = \text{given.} \tag{8.4}$$

$P(s_t|s_{t-1}, a_{t-1})$ is called the **transition matrix of the model**. $P(s_t|s_{t-1}, a_{t-1})$ reduces to $P(s_0)$ when $t = 0$.

$$P(r_t|s_t, a_t) = \delta(r_t, r(s_t, a_t))) \; . \tag{8.5}$$

$r : S_{\underline{s}} \times S_{\underline{a}} \to \mathbb{R}$ is a given **one-time reward function**.

Note that

$$P(s_\cdot, a_\cdot|\theta_\cdot) = \prod_{t=0}^{T-1} \{P(s_t|s_{t-1}, a_{t-1}) P(a_t|s_t, \theta_t)\} \; . \tag{8.6}$$

Define the **all times reward** Σ by

$$\Sigma(s_\cdot, a_\cdot) = \sum_{t=0}^{T-1} \gamma^t r(s_t, a_t) \; . \tag{8.7}$$

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

$$E\Sigma(\theta.) = E_{\underline{s}.,\underline{a}.|\theta.}\Sigma(\underline{s}., \underline{a}.) = \sum_{s.,a.} P(s., a.|\theta.)\Sigma(s., a.) \tag{8.8}$$

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:

$$\theta.^* = \text{argmax}_{\theta.} E\Sigma(\theta.) \tag{8.9}$$

Define a **future reward** for times $\geq t$ as:

$$\Sigma_{\geq t}((s_{t'}, a_{t'})_{t' \geq t}) = \sum_{t'=t}^{T-1} \gamma^{t'-t} r(s_{t'}, a_{t'}) \tag{8.10}$$

Define the following **conditional future rewards** (rewards for times $\geq t$, conditioned on certain quantities having given values):

$$\begin{aligned} v_t &= v(s_t, a_t; \theta.) = E_{\underline{s}.,\underline{a}.|s_t,a_t,\theta.}[\Sigma_{\geq t}] & (8.11)\\ V_t &= V(s_t; \theta.) = E_{\underline{s}.,\underline{a}.|s_t,\theta.}[\Sigma_{\geq t}] = E_{\underline{a}_t|s_t,\theta.}[v(s_t, \underline{a}_t; \theta.)] & (8.12) \end{aligned}$$

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:

$$E\Sigma(\theta.) = E_{\underline{s}_0,\underline{a}_0|\theta_0} v(\underline{s}_0, \underline{a}_0; \theta.) \tag{8.13}$$

Define a **one-time reward** and a **conditional one-time reward** as:

$$\begin{aligned} r_t &= r(s_t, a_t) & (8.14)\\ R_t &= R(s_t; \theta_t) = E_{\underline{a}_t|s_t,\theta_t}[r(s_t, \underline{a}_t)] \ . & (8.15) \end{aligned}$$

Note that

$$\begin{aligned} \Sigma_{\geq t} &= r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots + \gamma^{T-1-t} r_{t+(T-1-t)} & (8.16)\\ &= r_t + \gamma \Sigma_{\geq t+1}; . & (8.17) \end{aligned}$$

If we take $E_{\underline{s}.,\underline{a}.|s_t,a_t,\theta.}[\cdot]$ of both sides of Eq.(8.17), we get

$$v_t = r_t + \gamma E_{\underline{s}_{t+1},\underline{a}_{t+1}|\theta.}[v_{t+1}] \ . \tag{8.18}$$

If we take $E_{\underline{s}_\cdot,\underline{a}_\cdot|s_t,\theta_\cdot}[\cdot]$ of both sides of Eq.(8.17), we get

$$V_t = R_t + \gamma E_{\underline{s}_{t+1}|\theta_\cdot}[V_{t+1}] \ . \tag{8.19}$$

Note that

$$\begin{aligned} \Delta r_t &= r_t - R_t && (8.20)\\ &= r_t - (V_t - \gamma E_{\underline{s}_{t+1}|\theta_\cdot}[V_{t+1}]) && (8.21)\\ &= r_t + \gamma E_{\underline{s}_{t+1}|\theta_\cdot}[V_{t+1}] - V_t \ . && (8.22) \end{aligned}$$

Define

$$\Delta v_t = v_t - V_t \ . \tag{8.23}$$

Note that

$$\Delta v_t = \Delta r_t \ . \tag{8.24}$$

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)

Exact RL bnet

An exact RL bnet is given by Fig.8.3.

Fig.8.3 is the same as Fig.8.2 but with more nodes added in order to optimize the policy parameters. Here are the transition matrices, in blue, for the nodes not already discussed in connection to Fig.8.2.

$$P(\theta_t|\theta_\cdot) = \delta(\theta_t, (\theta_\cdot)_t) \tag{8.25}$$

$$\forall (s_t, a_t): \quad P(v_t(s_t,a_t)|r_t, v_{t+1}(\cdot), \theta_\cdot) = \delta(v_t(s_t,a_t), r_t + \gamma E_{\underline{s}_{t+1},\underline{a}_{t+1}|\theta_\cdot}[v_{t+1}]) \tag{8.26}$$

$$P(\theta_\cdot'|\theta_\cdot, v_0(\cdot)) = \delta(\theta_\cdot', \theta_\cdot + \alpha \partial_{\theta_\cdot} \underbrace{E_{\underline{s}_0,\underline{a}_0|\theta_0} v(\underline{s}_0, \underline{a}_0; \theta_\cdot)}_{E\Sigma(\theta_\cdot)}) \tag{8.27}$$

$\alpha > 0$ is called the **learning rate**. This method of improving $\theta_\cdot$ is called gradient ascent.

Concerning the gradient of the objective function, note that

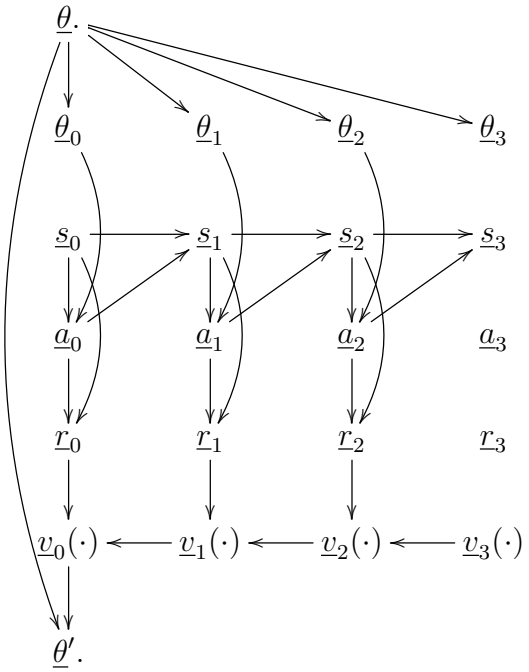

Figure 8.3: Exact RL bnet. $v_t(\cdot)$ means the array $[v_t(s_t, a_t)]_{\forall s_t, a_t}$ The following arrows are implicit: for all t, arrow from $\underline{\theta}. \to \underline{v}_t(\cdot)$. We did not draw those arrows so as not to clutter the diagram.

$$\partial_{\theta_t} E\Sigma(\theta.) = \sum_{s.,a.} \partial_{\theta_t} P(s.,a.|\theta.)\Sigma(s.,a.) \tag{8.28}$$

$$= \sum_{s.,a.} P(s.,a.|\theta.)\partial_{\theta_t} \log P(s.,a.|\theta.)\Sigma(s.,a.) \tag{8.29}$$

$$= E_{\underline{s}.,\underline{a}.|\theta.} \left\{\partial_{\theta_t} \log P(a_t|s_t,\theta_t)\Sigma(s.,a.)\right\} \ . \tag{8.30}$$

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:

$$\partial_{\theta_t} E\Sigma(\theta.) \approx \frac{1}{nsam(\vec{s}_t)} \sum_i \sum_{t=0}^{T-1} \partial_{\theta_t} \log P(a_t[i] \mid s_t[i],\theta_t) r(s_t[i],a_t[i]) \ . \tag{8.31}$$

The exact RL bnet Fig.8.3 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.

Actor-Critic RL bnet

For the actor-critic RL bnet, we approximate Eq.(8.31) by

$$\partial_{\theta_t} E\Sigma(\theta.) \approx \frac{1}{nsam(\vec{s})} \sum_i \sum_{t=0}^{T-1} \underbrace{\partial_{\theta_t} \log P(a_t[i] \mid s_t[i],\theta_t)}_{Actor} \underbrace{\Delta r_t(s_t[i],a_t[i])}_{Critic} \tag{8.32}$$

The actor-critic RL bnet is given by Fig.8.4. This bnet is approximate and assumes that the policy is known. The transition matrices for its nodes are given in blue below.

$$P(\theta_t) = \text{given} \tag{8.33}$$

$$P(s_t[i] \mid s_{t-1}[i], a_{t-1}[i]) = \text{ given} \tag{8.34}$$

$$P(a_t[i] \mid s_t[i],\theta_t) = \text{ given} \tag{8.35}$$

$$P(r_t[i] \mid s_t[i], a_t[i]) = \delta(r_t[i], r(s_t[i], a_t[i])) \tag{8.36}$$

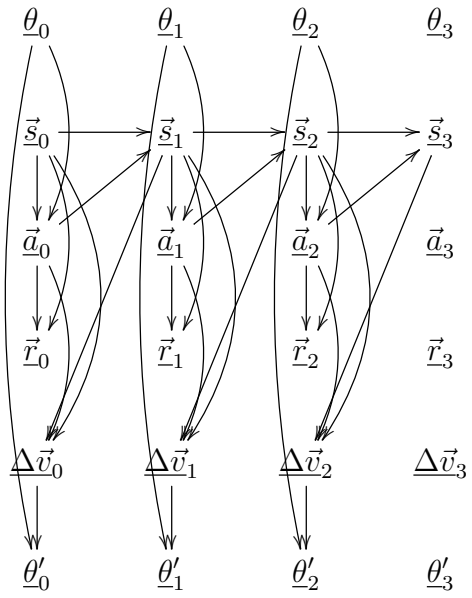

Figure 8.4: Actor-Critic RL bnet.

$r : S_{\underline{s}} \times S_{\underline{a}} \to \mathbb{R}$ is given.

$$P(\Delta v_t[i] \mid s_t[i], a_t[i], s_{t+1}[i]) = \delta(\Delta v_t[i], r(s_t[i], a_t[i]) + \gamma \hat{V}(s_{t+1}[i]; \phi') - \hat{V}(s_t[i]); \phi) \ . \tag{8.37}$$

$$P(\theta'.) = \delta(\theta'., \theta_t + \alpha \partial_{\theta_t} \sum_i \log P(a_t[i] \mid s_t[i], \theta_t) \Delta v_t[i]) \tag{8.38}$$

$\hat{V}(s_t[i]); \phi)$ is obtained by curve fitting (see Chapter 1) using samples $(s_t[i], a_t[i])\ \forall t, i$ with

$$y[i] = \sum_{t'=t}^{T} r(s_{t'}[i], a_{t'}[i]) \tag{8.39}$$

and

$$\hat{y}[i] = \hat{V}(s_t[i]; \phi) \ . \tag{8.40}$$

Eq.(8.39) 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'.

Q function learning RL bnet

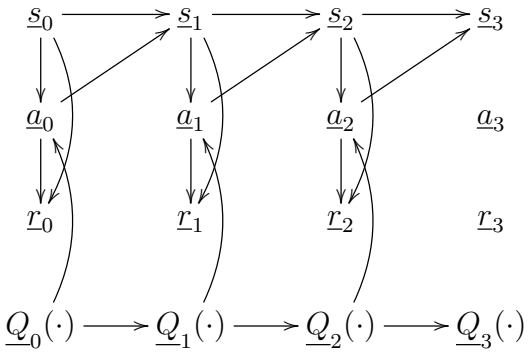

Figure 8.5: Q function learning RL bnet.

The Q-function learning RL bnet is given by Fig.8.5. This bnet is approximate and assumes that the policy is NOT known. The transition matrices for its nodes are given in blue below. (Remember that $Q = v$).

$$P(s_t|s_{t-1}, a_{t-1}) = \text{ given} \tag{8.41}$$

$$P(a_t|s_t, v_t(\cdot)) = \delta(a_t, \text{argmax}_a v_t(s_t, a)) \tag{8.42}$$

$$P(r_t|s_t, a_t) = \delta(r_t, r(s_t, a_t)) \tag{8.43}$$

$r : S_{\underline{s}} \times S_{\underline{a}} \to \mathbb{R}$ is given.

$$\begin{aligned} \forall (s_t, a_t): \quad & P(v_t(s_t, a_t)|v_{t-1}(\cdot)) = \\ & = \delta(v_t(s_t, a_t), r(s_t, a_t) + \gamma \text{max}_a E_{\underline{s}_{t+1}|s_t, a_t} v_{t-1}(\underline{s}_{t+1}, a)) \end{aligned} \tag{8.44}$$

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.8.6) instead of Fig.8.5 and replace Eq.(8.44) by

$$\begin{aligned} \forall (s_t, a_t): \quad & P(v_t(s_t, a_t)|s_{t+1}, v_{t-1}(\cdot)) = \\ & = \delta(v_t(s_t, a_t), r(s_t, a_t) + \gamma \text{max}_a v_{t-1}(s_{t+1}, a)) \ . \end{aligned} \tag{8.45}$$

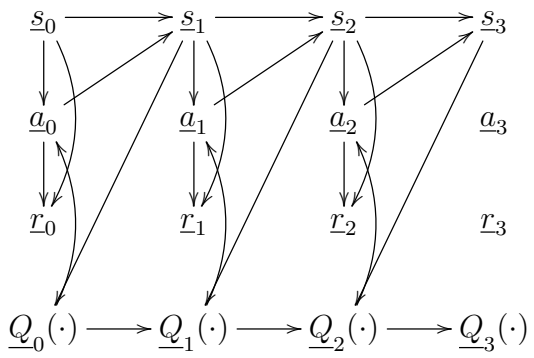

Figure 8.6: Q function learning RL bnet. Same as Fig.8.5 but with new arrow passing s_t to Q_{t-1}.

Chapter 9

Simpson's Paradox

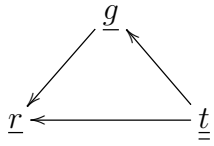

Figure 9.1: bnet for a simple case of Simpson's paradox.

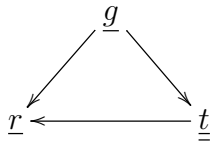

Figure 9.2: Equivalent to Fig.9.1

I wrote a post about this in 2020 for my blog "Quantum Bayesian Networks". See Ref.[6].

Bibliography

[1] Robert R. Tucci. Bell's inequalities for Bayesian statisticians. blog post in blog "Quantum Bayesian Networks.

[2] 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.

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

[4] 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".

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

[6] Robert R. Tucci. Simpson's paradox, the bane of clinical trials. blog post in blog Quantum Bayesian Networks, https://qbnets.wordpress.com/2020/07/09/simpsons-paradox-the-bane-of-clinical-trials/.