Vector Algebra

Constants => Diag. Matrices

$XX^T = (XX^T)^T = X^{T^T}X^T = XX^T$

$\frac{\partial}{\partial x}(x^T a) = \frac{\partial}{\partial x}(a^T x) = a$; $\frac{d}{dw}(w^T w) = 2w$; $(AB)^{-1} = B^{-1}A^{-1}$

$\frac{d}{dx} x^T A x = Ax + A^T x = (A + A^T)x = (A^T + A)x$

$(X^T)^{-1} = (X^{-1})^T$. Remember: If hard, derive elementwise.

Trick: derive w.r.t. every vector elem => many summands become 0

$\frac{d}{dx} \|f(x)\|_{L2} = \frac{f(x)}{\|f(x)\|_{L2}} \cdot \partial f(x)$

Convex

A twice diff.able function is convex iff its Hessian $H \in \mathbb{R}^{d \times d}$ satisfies $w^T H w \geq 0 \;\forall w \in \mathbb{R}^d$.

The function would be concave if $-f(x)$ is strictly convex.

Alternative criterium: convex if $\forall u, v, \lambda \in [0,1]$ $f(\lambda u + (1-\lambda)v) \leq \lambda f(u) + (1-\lambda) f(v)$

i.e. if the function lies below the connecting line | convex + convex = convex

Random Formula Knowledge

$\int_{\mathbb{R}} \exp\left\{-\frac{1}{2}\frac{(x+\frac{b}{a})^2}{1/2a}\right\} dx = \sqrt{\pi / a}$

Standard $\mathcal{N} = \varphi(x) = \frac{1}{\sqrt{2\pi}} e^{\ldots}$ => $\varphi'(x) = -x\varphi(x)$

$\frac{\partial}{\partial x} \log_b(x) = \frac{1}{x \ln(b)}$

$\int_{\mathbb{R}} e^{-(ax^2 + 2bx + c)} = \int_{\mathbb{R}} \exp\left\{-a\left[\left(x + \frac{b}{a}\right)^2 - \frac{b^2 - ac}{a^2}\right]\right\} dx = e^{\frac{b^2 - ac}{a}} \sqrt{\frac{\pi}{a}}$

$\log(a^b) = b \cdot \log(a)$

Covariance Matrix, Bivariate normal random Variable

$\mathcal{N}(\Sigma, \mu) => f_x(x) = \frac{1}{\sqrt{(2\pi)^2 \det(\Sigma)}} \exp\left\{-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)\right\}$

$\Sigma = Cov(\vec{X}) = E[(\vec{X} - \mu)(\vec{X} - \mu)^T]$

$= \begin{bmatrix} (X_1 - \mu_1)^2 & \cdots & (X_1 - \mu_1)(X_N - \mu_N) \\ (X_2 - \mu_2)(X_1 - \mu_1) & & \\ \vdots & & \\ \vdots & \cdots & (X_N - \mu_N)^2 \end{bmatrix}$

PCA & Empirical Covariance Matrix

Center Data by subtracting the mean. Compute $\Sigma = \frac{1}{n}\sum_{i=1}^{n} x_i x_i^T$ and its eigenvalues $\det(\Sigma - \mathbb{I}\lambda) \stackrel{!}{=} 0$ and eigenvectors $(\Sigma - \mathbb{I}\lambda)x = 0$.

For solving: Gaussian Elimination (same change on both sides)

Determinants are computed by taking $a_{ij} \cdot \det$(Matrix without row$_i$ and col$_j$) $- a_{i(j+1)} \cdots + a_{i(j+2)} \cdots$ etc.

And then make the result unit vectors for PCA.

Inverse of a matrix

Write down $M | \mathbb{I}$. Apply row-wise operations to both sides until LHS is $\mathbb{I}$.

Confusion Matrix

rec: $\frac{TP}{TP+FN}$ prec: $\frac{TP}{TP+FP}$

Table with numbers of actual vs predictions

Recall = P(isDog | predDog), Precision = P(predDog | isDog)

Accuracy: $\frac{TP+TN}{TP+TN+FP+FN}$ F1: $\frac{2TP}{2TP+FP+FN}$

Linear Regression and Regularizers

$\arg\min_w \sum_{i=1}^{n} (y_i - w^T x_i)^2 + \lambda_{Lasso}\|w\|_{L1} + \lambda_{Ridge}\|w\|_{L2}^2$ (or $\lambda_{Ridge} w^T w$)

Lasso, Ridge — elastic net

L2 drives params down quickly but doesn't zero them out

Gradient Descent

$x_{n+1} = x_n - \gamma_n \nabla F(x_n)$, Stepsize $\gamma_n = 1/n$

$\gamma_n = \frac{(x_n - x_{n-1})^T (\nabla F(x_n) - \nabla F(x_{n-1}))}{\|\nabla F(x_n) - \nabla F(x_{n-1})\|^2}$

Logistic Regression

$\ln\left(\frac{P(Y=1)}{1 - P(Y=1)}\right)$ limits the result to at most 1 and at least 0

$\log = X\beta = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p$

odds that $Y = 1 = e^{\beta_0} e^{\beta_1 x_1} e^{\beta_2 x_2} \cdots e^{\beta_p x_p}$

Generative & Discriminative

y = labels, x = datapoints

Gen: $P(x|y)$ and $P(y)$ model how data was created => $P(x,y)$ example: Naive Bayes

Dis: $P(y|x)$ models decision Boundary between classes e.g. SVM, K-Means Clustering, Perceptron

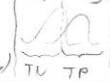
ROC Curve

True Positives as ratio of all de facto positives on y Axis = Sensitivity: How sensitive is the test to sick people?

True Negatives ... on x Axis = Specificity. Increase in one, decrease in the other. (reversed → False Positives)

Test (Classifier) determines those depending on some threshold that can be moved.

ROC Curve is sensitivity on x-Axis and (1 − Specificity) on y-Axis, i.e. the False positives. Better classifiers have ⌐ and not ⁄.



Naive Bayes

assumes independent feature dimensions given y (conditionally):

$P(\vec{X}|y) \stackrel{!}{=} P(X_1|y)P(X_2|y)P(X_3|y)$ ← i.e. poisson for spam mails

and computes $P(y|X) = P(X|y)P(y) / P(X)$, $P(X) = \sum_y p(x, y)$

Maximum A Posteriori (MAP)

$P(\theta|X)$ maximized => $\max_\theta (P(X|\theta)P(\theta))$

Maximum Likelihood Estimation (MLE)

$\max_\theta P(X|\theta)$

Perceptron Algorithm

Gradient descent on perceptron loss function ℓ_p:

$\nabla \ell_p(w; x, y) = \begin{cases} 0 & , \text{if } y = \text{sign}(w^T x) \\ -yx & , \text{if } y \neq \text{sign}(w^T x) \end{cases}$

Laplacian smoother: Counting words... one is not in Dataset => probability = 0 => problem.

Solution: Add 1 to every word's occurrence count in the data, and add #words to the divisor. $\frac{2}{31}$ becomes $\frac{2+1}{31+74}$

Perceptron and SVM

Perceptron: $\min \sum \max\{0, -y_i w^T x_i\}$

SVM: $\min \sum \max\{0, 1 - y_i w^T x_i\} + \lambda \|w\|_{L2}^2$

Gradient $\nabla G(w) = \frac{1}{n}\sum_i^B \nabla g_i(x_i; w)$

$\nabla g_i(x_i; w) = \nabla \max(0, 1 - y_i w^T x_i) + \nabla \lambda \|w\|_{L2}^2$

$(\nabla \lambda \|w\|_{L2}^2)_j = 2\|w\|_{L2} \frac{\lambda w_j}{\|w\|_{L2}} = 2w_j \Rightarrow \nabla(\lambda\|w\|_{L2}^2) = \begin{pmatrix} 2w_1 \\ \vdots \\ 2w_j \end{pmatrix}$

$A^T A$ is always pos. semidef.

$\nabla \max(0, 1 - y_i w^T x_i) = \begin{cases} 0, & \text{if } y_i w^T x_i \geq 1 \\ -y_i x_i, & \text{otherwise} \end{cases} = -\sum_{i: y_i \neq \text{sign}(w^T x_i)} y_i x_i$

SVM: To get 0 Loss, you have to classify not only correctly, but correctly by some margin. This is hinge Loss: just shifted perceptron Loss by 1. If it were some other nonzero number, we would get the same solution just by scaling w. So we also want to keep the weights small to keep the margin larger, which scales with $\frac{2}{\|w\|}$.

SVM Gradient Update Derivation with SGD

From above Gradient, we get

$w = \begin{cases} w + \eta(-2\lambda w), & \text{if } y_i w^T x_i \geq 1 \\ w + \eta(y_i x_i - 2\lambda w), & \text{otherwise} \end{cases}$

Stochastic Gradient Descent (SGD) with batchsize 1

Initialize somehow, then: for t = 1, 2, ... do {
pick $i_t \sim Unif(1, n)$.
if$(y_{i_t} \neq \text{sign}(w_t^T x_{i_t}))${
$w_{t+1} = w_t + \eta_t y_{i_t} x_{i_t}$
}else{ $w_{t+1} = w_t$ }}

e.g. $w_0 = 0$

Kernel Trick

want: artificially high dimensionality to linearly separate data that isn't linearly separable.

$\varphi\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} x_1^2 \\ x_2^2 \\ x_1 x_2 \sqrt{2} \end{pmatrix} => \varphi(\vec{x_a})^T \varphi(\vec{x_b}) = (x_{a1}x_{b1} + x_{a2}x_{b2})^2 = (x_a^T x_b)^2 \stackrel{!}{=} k$

k is easier to compute than first transforming x_a and x_b with $\varphi(x)$. Generalization with d input dimension instead of 2: $x_a = [x_{a1}, x_{a2}, \ldots, x_d]^T$, same for x_b, both are inputs.

$\varphi(x) = [x_1^2, x_2^2, x_1 x_2 \sqrt{2}] \in \Theta(d^2)$ is way too costly to compute.

But $(x_a^T x_b)^2 \in \Theta(d)$.

For monomials of degree m instead of 2:

$\varphi(x) = [x_1^m, x_2^m, \ldots, x_d^m, x_1^{m-1} x_2, \ldots]^T$ => kernel $k(x_a, x_b) = (x_a^T x_b)^m$

All monomials up to degree m: $k(x_a, x_b) = (1 + x_a^T x_b)^m$

Explicit computation would be exponential in m.

Kernelized perceptron

$\min_{\alpha_{1:n}} \sum_{i=1}^{n} \max\left\{0, -\sum_{j=1}^{n} \alpha_j y_i y_j x_i^T x_j\right\}$

K-Means Clustering

1. Assign every Datapoint a cluster whose mean is closest.
2. Recompute cluster means (sum over all, divide by num)

Repeat until not changing

Kernel Requirements

Note: smth is also a kernel if it can be replaced by scalar product.

$K_{ij} = k(x_i, x_j)$ Gram Matrix K must be pos. semi-definite. $k(x_a, x_b) \stackrel{e.g.}{=} (1 + x_a^T x_b)^2 = 1 + x_{a1}^2 x_{b1}^2 + x_{a2}^2 x_{b2}^2 + 2x_{a1}x_{b1} + 2x_{a2}x_{b2} + 2x_{a1}x_{a2}x_{b1}x_{b2}$

$= \varphi(x_a) \cdot \varphi(x_b) => \varphi = [1, x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1 x_2]^T$

must also be symmetric and pos. semidefinite.

So $k_1 - k_2$ is not necessarily a kernel again.

Funfact: set of vectors is linearly independent iff $\det(K) \neq 0$.

EM Algorithm

Have Datapoints but don't know their color or distribution. Assume Normal Distribution.

1. guess some parameters
2. Evaluate those PDFs at every datapoint to get likelihood of red/blue for datapoint given the guessed params
3. create weights $red_w = \frac{Pred}{Pred + Pblue}$ and same for blue
4. Compute mean and Variance twice. Once for red and for blue. => new params for red and blue PDF.

estimVariance = sum(weights * (data − old Mean)^2) / sum(weights)

estimMean = sum(data * weights) / sum(weights)

Neural Network

Given $\varphi(z) = \frac{1}{1 + e^{-z}}$, value$_j^{layer} = \varphi(w_0 + \sum_{k \in layer-1} w_{jk} v_k^{\ell-1})$

Treat final outputs as 1 if greater than 0.5, otherwise 0.

Give weights for: (cus φ output is real number)

OR: $A \vee B$ = round($\varphi(w_0 + w_A A + w_B B)$) => $w = [0.5, 1, 1]^T$

AND: $A \wedge B$ = ... => $w = [-1.5, 1, 1]^T$ (for this φ)

Task: Create a NNet with only one hidden layer and an arbitrary number of nodes that implements <expr>. Possible approach: turn expr into ORs of ANDs and try putting all ORs into second layer. Output of that 2nd layer is $\varphi(h_1 + h_2 + h_3 + h_4)$ because one 1 is enough.

1st layer: $(A \wedge B \wedge C) => h_1 = \varphi(A + B + C - 2.5)$

$(B \wedge C \wedge \neg D) => h_2 = \varphi(B + C - D - 1.4)$

$(\neg A \wedge B \wedge \neg C) => \varphi(-A + B - C - 0.4)$

To get close enough to the changing bounds 0.5 that 0 has an influence.

Task: XOR with as few units as possible.

A —1→ ◯ —(−1)→ out; B —2→ ◯ —0.84

$=> \varphi(-\varphi(A + B) + 0.54\varphi(2A + 2B))$

Taylor Expansion

$f(x) \approx f(a) + \frac{f'(a)}{1!}(x - a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \ldots$

$\frac{1}{1-x} = \sum_{n=0}^{\infty} x^n$, $e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!}$, $\cos(x) = \sum_{n=0}^{\infty} \frac{(-1)^n x^{2n}}{(2n)!}$

$\sin(x) = \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n+1}}{(2n+1)!} = \sum_{n=1}^{\infty} \frac{(-1)^{n-1} x^{2n-1}}{(2n-1)!}$

Lagrange Multiplier for optimizing

Constraint $g(x) \stackrel{!}{=} 0$. $\mathcal{L}(x, \lambda) = f(x) - \lambda g(x)$

derivative must equal 0 => critical points. Check 2nd derivative or compare values to determine if maximal.

Example HW6: Make sure that $\sum_{j=1}^{m} \theta_j = 1$:

$\mathcal{L}(\theta, \lambda) = \sum_{i=1}^{n}\sum_{j=1}^{m} \gamma_j(x_i)\left[\log \theta_j + \sum_{\ell=1}^{k} \log \theta_{\ell j}^{x_{i\ell}} (1 - \theta_{\ell j})^{1 - x_{i\ell}}\right] + \lambda\left(\sum_{j=1}^{m} \theta_j - 1\right)$

From the constraint g = 0, we can now find λ.

$\frac{\partial \mathcal{L}}{\partial \theta_{\ell j}} \stackrel{!}{=} 0 => \theta_{\ell j} = \ldots$

$\frac{\partial \mathcal{L}}{\partial \lambda} \stackrel{!}{=} 0 => \theta_j = \ldots$

Generalization Error

e.g. $(w^T x - y)^2$ e.g. $w^T x$

Expected Error of prediction function $f(x)$ is
$I[f] = \int_{X \times Y} L(f(x), y)\, p(x, y)\, dx\, dy$ where L is a loss function and p is the unknown joint probability distribution for x and y. We don't know p so we cannot compute $I[f]$.
Instead, we compute the empirical error $I_S[f] = \frac{1}{n} \sum_{i=1}^{n} L(f(x_i), y_i)$. The generalization error is $I[f] - I_S[f]$.
An Algorithm is said to generalize if this goes towards 0 wrt. n.

Regularization

(see also page 1)

Lasso: $\|w\|_{\ell_1} = \sum_i w_i$
Ridge: $\|w\|_{\ell_2}^2 = \sum_i w_i^2$
Euclidean: $\|w\|_{\ell_2} = \sqrt{\sum_i w_i^2}$
Ridge vs Euclidean:
=> Euclidean drives quickly towards 0, so does ridge. But Ridge scales with the euclidean distance squared, so further away from 0 is punished harder => more stable near 0.

Proving positive semi-definiteness

- all Eigenvalues are ≥ 0   ($X^T X$ => all singular values are squares: $U\Sigma V^T$ ... $= V\Sigma^T\Sigma V \Rightarrow D = \Sigma^2$)
- $\forall z \neq \vec{0}$ $z^T M z$
- Gram Matrix was made from lin. indep. vectors x_i
- Sylvesters Criterion: leading principal minors positive (≥)
  k-th leading principal minor = M's upper left k×k submatrix's determinant. Pos. definite iff all those are positive.

=> Test by reducing to upper triangular like in gaussian Elimination. (Careful when pivoting to keep sign of det correct) Now check that all diagonal elements are > 0. (bzw ≥ 0)

Cumulative Distribution Function

CDF = $F(x \leq a)$ input
PDF = $f(x = a)$

Marginal Density

$f_Y(x) = \int f_{X,Y}(x, y)\, dx$
$= \int f_{Y|X}(y|X = x) f_X(x)$

Kernels Named

$k_{linear}(x, x') = x^T x'$
$k_{poly}(x, x') = (x^T x' + 1)^2$
φ_{poly} s.t. $k_{poly} = \varphi(x)\varphi(x')$
$= 1, \sqrt{2}x_1, \dots, \sqrt{2}x_n, \sqrt{2}x_2x_1, \sqrt{2}x_{n-1}x_{n-2}, \sqrt{2}x_{n-1}x_{n-3}, \sqrt{2}x_n x_{n-1}, x_1^2, \dots, x_n^2$

Probability

$P(a, b | c) = P(a | b, c) P(b | c)$
$P(a \wedge b \wedge c) = P(a) P(c | a, b) = P(a) P(b|a) \cdot P(c|a, b)$
$P(x | y, z) = \frac{P(x, y | z)}{P(y | z)}$
$P(a, b | c) = \frac{P(a, b, c)}{P(c)} = \frac{P(a | b, c) P(b, c)}{P(c)}$
$= \frac{P(a | b, c) P(b | c) P(c)}{P(c)}$

EM-Algo Cancer

2015 7.i)

Given: $P(X = x | \lambda) = \begin{cases} \frac{\lambda^x}{x!} e^{\lambda} & \text{if } \lambda \in \mathbb{N} \\ & \text{otherwise} \end{cases}$
Two distributions (implicit) Z_1, Z_2. Probability that a point is sampled from $Z_1 = \pi$. $\lambda_1, \lambda_2 > 0$.
Given "posterior probabilities in the E step" after some EM-iterations as a table:

	$Q(Z = 1 \| X = x_i)$	$Q(Z = 2 \| X = x_i)$
$x_1 = 2$	0.3	0.7
$x_2 = 4$	0.8	0.2
$x_3 = 8$	0.3	0.7

"Write down the objective that the M-step is optimizing wrt. the params $\pi, \lambda_1, \lambda_2$. Remember that this is the expected log-likelihood of the complete Data under the posterior distribution $Q(Z|X)$. Formally:
$\mathbb{E}_{Q(Z|X)}[\log(P(x_1, x_2, x_3, z_1, z_2, z_3))]$"

Solution:
individual datapoints are independent:
$\Rightarrow \mathbb{E}_{Q(Z|X)}[\log(\prod_{i=1}^{3} P(x_i, z_i))]$
log trick:
$\Rightarrow \mathbb{E}_Q[\sum_{i=1}^{3} \log(P(x_i, z_i))]$ $=: C_{ij}$
Sum of E:
$\Rightarrow \sum_{i=1}^{3} \mathbb{E}_Q[\log(P(x_i, z_i))]$
Usually, $\mathbb{E}_{P(x)}[f(x)] = \sum_{j=1}^{\#x} f(x_j) P(x_j)$,
=> here we have a distribution $Q(Z|X)$, so it's about Z!
$\Rightarrow \sum_{i=1}^{3} \sum_{j=1}^{2} C_{ij} Q(z = z_j | X = x_i)$
Probability rules on $P(x_i, z_i)$:
$\Rightarrow C_{ij} = \log(P(X = x_i | Z = z_j) P(Z = z_j))$
(given as a poisson distribution; π or $(1-\pi)$)

Case distinction over z_j to use that:
$\sum_{i=1}^{3} [Q_1(\log(P(Z=1) P(X = x_i | Z = 1))) +$ (π; poisson λ_1)
$Q_2(\log(P(Z=2) P(X = x_i | Z = 2)))]$ ($(1-\pi)$; poisson λ_2)

And use the logarithm again to turn the product into a sum.
$\sum_{i=1}^{3} Q_1(\log(\pi) + x_i \log(\lambda_1) - \lambda_1 - \log(x_i!)) + Q_2(\log(1-\pi) + x_i \log(\lambda_2) - \lambda_2 - \log(x_i!))$

Jensen's inequality

$f(E[X]) \leq E[f(x)]$
For a convex function: $f(t \cdot u + (1-t)v) \leq t \cdot f(u) + (1-t) f(v)$. $\forall u, v$ $\forall t \in [0, 1]$

HW1

a) Assume invertible $(X^T X)$. Show that lost $w^* = (X^T X)^{-1} X^T y$.
solution: $\|Xw - y\| = (Xw - y)^T (Xw - y) = w^T X^T X w - 2 w^T X^T y + y^T y$
$\nabla R(w) = 2X^T X w - 2X^T y \stackrel{!}{=} 0$
$\Rightarrow w^* = (X^T X)^{-1} X^T y$
$y^T X w = w^T X^T y$ because same dimensionality

b) Show that if $X \in \mathbb{R}^{n \times d}$ with $n < d$, this does not admit a unique solution.
Intuition: X (n×d) w (d) => underdetermined when solving for w.
Formally: $X \stackrel{SVD}{=} U \Sigma V^T$ (unitary n×n, diagonal n×d, unitary d×d)
$z := V^T w$ (ok because V unitary and thus bijection)
=> ... $\min_z (z^T \Sigma^2 z - 2 y^T U \Sigma z)$ (diagonal)
=> independent problems: $\min_{z_i} = z_i^2 \sigma_i^2 - 2(U^T y)_i z_i \sigma_i$
[illegible] convex => solution at derivative = 0: $z_i \sigma_i^2 - (U^T y)_i \sigma_i = 0$
So if any $\sigma_i = 0$, we have an underdetermined system.
This happens if $n < d$.
[illegible] $X^T X \in \mathbb{R}^{d \times d}$ was not full rank and $(X^T X)^{-1}$ doesn't exist

EM Slides (soft)

$\forall \gamma_i\ \gamma_i = 1$ would be hard

E: Compute cluster membership probabilities $\gamma_j(x_i)$
for each i, j given μ_j, Σ_j, w from the previous step.
$\gamma_j(x_i) = P(Z = j | x_i, \Sigma, \mu, w) = \frac{w_j P(x_i | \Sigma_j, \mu_j)}{\sum_\ell w_\ell P(x_i | \Sigma_\ell, \mu_\ell)}$ (sum; used again next step)

M: Fit clusters (z-distributions) to weighted data.
$w_j \leftarrow \frac{1}{n} \sum_{i=1}^{n} \gamma_j(x_i)$
$\mu_j \leftarrow \frac{\sum_{i=1}^{n} \gamma_j(x_i) \cdot x_i}{\sum_{i=1}^{n} \gamma_j(x_i)}$
$\Sigma_j \leftarrow \left(\sum_{i=1}^{n} \gamma_j(x_i)(x_i - \mu_j)(x_i - \mu_j)^T\right) / \sum_{i=1}^{n} \gamma_j(x_i)$

also called MLE Rules

Semi-Supervised Learning

like soft-EM above but with $\gamma_j(x_i) = [j = y_i]$ for known y_i

Normal Distribution

$\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{(x - \mu)^2}{2\sigma^2}\right\}$

Expected Value Math

$\frac{d}{da} E[\dots] = E[\frac{d}{da} \dots]$
$E[X + Y] = E[X] + E[Y]$
$E[aX + b] = aE[X] + b$
$E[XY] = E[X]E[Y] + Cov(X, Y)$ (i.i.d. => 0)
$E[f(x)] = \int f(x) P(x)\, dx$

Hard EM Formulas

M-step: $\theta^{new} = \arg\max_\theta \sum_i^n \log \sum_j^k w_j P(x_i; \theta)$
which results in:
$w_j = \frac{1}{n} \sum_{i: z_i = j}^{k} x_i$, $\Sigma_j = \frac{1}{n} \sum_{i: z_i = j}^{k} (x_i - \mu_j)(x_i - \mu_j)^T$
$\mu_j =$

EM

$P_{red} = 0.5$, $P_{Green} = 0.5 - \theta$, $P_g = ?$
Anzahl $X_r + X_g = \alpha$, $X_b = n - \alpha =: \beta$
$L^{(i)}(\theta) = E_Q[\log(P(\bar{X} | \theta))]$
$\dots \Rightarrow \approx E_Q[\log(c)] + E_Q[X_r] \log(0.5) + E_Q[X_g] \log(\theta) + \beta \log(0.5 - \theta)$
maximize (M-step): $\frac{\partial L}{\partial \theta} = \frac{E[X_g]}{\theta} - \frac{\beta}{0.5 - \theta} \stackrel{!}{=} 0 \Rightarrow \theta = \dots$

E-step: Compute $E[X_g] = E[X_g | \alpha, \beta, \theta]$
$= \sum_n P(g | \alpha, \beta, \theta, r \vee g \vee b) \cdot 1$
$= \sum_n P(g | \alpha, \beta, \theta, r \vee g) P(r \vee g | \alpha, \beta, \theta) + P(g | \alpha, \beta, \theta | b) P(b | \alpha, \beta, \theta)$ (0)
$= \sum_n \frac{\alpha}{n} P(g | r \vee g, \alpha, \beta, \theta) = \sum_n \frac{\alpha}{n} \frac{\theta}{0.5 + \theta} = \frac{\alpha\theta}{0.5 + \theta}$

And also: $P(g | r \vee g \vee b) \underset{\text{since}}{=} \theta$
$= P(g | r \vee g) \frac{\alpha}{n} \Rightarrow E[X_g] = \frac{\alpha n \theta}{\alpha} = n\theta$

More Probability

$P(w | x, y) = \frac{P(w, x, y)}{P(x, y)} = \frac{P(y | w, x) P(w, x)}{P(y | x) P(x)}$
$= \frac{P(y | w, x) P(w | x) P(x)}{P(y | x) P(x)} = \frac{P(y | w, x) P(w | x)}{P(y | x)}$