01:XXX:XXX - Homework n

Pranav Tikkawar

May 12, 2025

Note Sheet

Inner Product: $u^TAv = \langle u, v\rangle$. A is symmetric and positive definite. Dot Product is an inner product with $A = I$. Notice that a symmetric positive definite matrix has all positive eigenvalues and positive determinant.

Orthogonal Projection: We have the projecton $\pi(x)$ onto some subspace W. We want to minimize $||x - \pi(x)||^2$. By the orthogonality condition, we have $\langle x - \pi(x), b\rangle = 0$. Since π is a linear map, we can write $\pi(x) = P_\pi x$ thus $\langle x - P_\pi x, b\rangle = 0 \implies \langle x - \lambda b, b\rangle = 0$. This results in $\lambda = \frac{\langle x,b\rangle}{\langle b,b\rangle}$. $\lambda = (B^TB)^{-1}B^Tx$, $\pi(x) = B(B^TB)^{-1}B^Tx$. and $P = B(B^TB)^{-1}B^T$.

SVD: Decompose $A = U\Sigma V^T$. Notice $A^TA = V\Sigma U^TU\Sigma V^T = V\Sigma^2V^T$. And $AA^T = U\Sigma V^TV\Sigma U^T = U\Sigma^2U^T$. U = eigenvectors of AA^T. V = eigenvectors of A^TA. Σ = square root of eigenvalues of A^TA. Notice that if A is $m \times n$ then U is $m \times m$, Σ is $m \times n$, and V is $n \times n$. A geometric intuition for SVD is that U is the ON basis of the column space of A, V is the ON basis of the row space of A, and Σ is the scaling factor. You can also Write $A = \sum_{i=1}^r \sigma_i u_i v_i^T$ where r is the rank of A. Thus the SVD is a chnage of basis in the domain, then an independent scaling, and then a change of basis in the codomain.

Best Fit Line: With n points (x_i, y_i), minimize $\sum_{i=1}^n (y_i - (\theta_1 x_i + \theta_0))^2$. We can see that the best fit line is the orthogonal projection of $(y_1, y_2, \ldots, y_n)$ onto the subspace spanned by $(x_1, x_2, \ldots, x_n)$.

Reaching Line of best fit: for a 2 dimensional space, we can take the IPM of x_i as $\begin{bmatrix} n & \sum x_i \\ \sum x_i & \sum x_i^2 \end{bmatrix}$. ie B^TB where $B = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}$. Solve $B^TB\Theta = B^TY$ where $Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$. Using Cramer's rule for matrix inverses, we can find Θ as: $\theta_0 = \frac{(\sum y_i)(\sum x_i^2) - (\sum x_i)(\sum x_i y_i)}{n\sum x_i^2 - (\sum x_i)^2}$ and $\theta_1 = \frac{n\sum x_i y_i - (\sum x_i)(\sum y_i)}{n\sum x_i^2 - (\sum x_i)^2}$. In terms of IP it is $\theta_0 = \frac{\langle x,x\rangle\langle 1,y\rangle - \langle 1,x\rangle\langle x,y\rangle}{\langle 1,1\rangle\langle x,x\rangle - \langle 1,x\rangle^2}$ and $\theta_1 = \frac{\langle 1,1\rangle\langle x,y\rangle - \langle 1,x\rangle\langle 1,y\rangle}{\langle 1,1\rangle\langle x,x\rangle - \langle 1,x\rangle^2}$. Note this abstracted version works with any inner product.

Differentiation Diff is a linear map. The gradient of a function is the vector of partial derivatives. $\Delta_x f = \begin{bmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} & \cdots & \frac{\partial f}{\partial x_n} \end{bmatrix}$. The directional derivative is $\Delta_x f \cdot v$ where v is the direction. The Jacobian is the matrix of partial derivatives. $J_f = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$. When we consider the line of best fit the derivate of our function $||Ax - b||^2$ is $2(Ax - b)^TA$. Setting this to 0 gives us the equation $A^TAx = A^Tb$. Note that a jacobian of $f(x) = Ax$ is A.

Identities for Gradients: Short hand: $\frac{\partial}{\partial X} f = f_X$:

$$\begin{aligned}
(f(X)^T)_X &= (f(X)_X)^T \\
(\text{tr} f(X))_X &= \text{tr}(f(X)_X) \\
(\det f(X))_X &= \det(f(X))\text{tr}(f(X)^{-1} f(X)_X) \\
(f(X)^{-1})_X &= -f(X)^{-1} f(X)_X f(X)^{-1} \\
(a^T X^{-1} b)_X &= -(X^{-1})^T a b^T (X^{-1})^T \\
(x^T a)_x &= a^T \\
(a^T x)_x &= a^T \\
(a^T X b)_X &= a b^T \\
(x^t B x)_x &= x^T (B + B^T) \\
((x - As)^T W (x - As))_s &= -2(x - As)^T W A
\end{aligned}$$

Chain Rule: $f(g(x))' = f'(g(x))g'(x)$. We can use this for multivariable functions as well. $\Delta_x f(g(x)) = \Delta_{g(x)} f \cdot \Delta_x g$.

Backpropgation: $f_0 = x$ and $f_i = \sigma(A_{i-1} f_{i-1} + b_{i-1})$. We also have a loss function $L(\theta) = ||y - f_K(\theta, x)||$ which we want to minimize. We can use the chain rule to find the gradient of L with respect to θ. We have:

$$\begin{aligned}
\Delta_\theta L &= \Delta_{f_K} L \cdot \Delta_\theta f_K \\
&= \Delta_{f_K} L \cdot \Delta_{f_{K-1}} f_K \cdot \Delta_\theta f_{K-1} \\
&= \Delta_{f_K} L \cdot A_{K-1}^T \sigma'(A_{K-1} f_{K-2} + b_{K-1}) \Delta_\theta f_{K-1}
\end{aligned}$$

Example (Steps of tuning a weight)**.**

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial n} \frac{\partial n}{\partial w}$$

where L is loss, $\hat{y}$ is the prediction, and n is the neuron output.
Once you calculate the gradient, you can update the weights using the gradient descent algorithm. The update rule is given by:

$$w_{new} = w_{old} - \eta \frac{\partial L}{\partial w}$$

where η is the learning rate.

Maximum Likelihood Estimation: We want to find the parameters θ that maximize the likelihood function $L_{MLE}(\Theta) = \arg\max \Pi_{i=1}^{N} P(y_i|\theta, x_i)$. Ie max the joint probability of the data given the parameters. We can use the log likelihood function to convert the product to a sum while maintaining the same maximum.

MLE EX: The step is to find the Likelihood function $L(\theta) = \prod_{i=1}^{N} P(y_i|\theta, x_i)$. For example, if we have $y_i \sim N(\mu, \sigma^2)$, then the likelihood function is given by:

$$L(\theta) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_i-\mu)^2}{2\sigma^2}\right)$$

Taking the log of the likelihood function, then taking the derivative with respect to θ and setting it to 0 gives us the MLE estimates.

Regression with features: Consider that $y \sim N(\theta_0 + \theta_1 x + \theta_2 x^2, \sigma^2)$. We can write this as $y = \theta_0 + \theta_1 x + \theta_2 x^2 + \epsilon$. We can use MLE while considering the pdf of the normal distribution. This boils down to minimizing the function $L(\theta) = ||X\Theta - Y||^2$ where $X = \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ \vdots & \vdots & \vdots \\ 1 & x_n & x_n^2 \end{bmatrix}$, $\Theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{bmatrix}$, and $Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$. With the equations $X^TX\Theta = X^TY$ we can find Θ as $\Theta = (X^TX)^{-1}X^TY$.

Learning/Training: When we have an instance of parameters θ, we want to know how to do better. Use an "evaluation" function $J(\theta)$ to measure how well the model fits the data. We can do better by moving in the direction where J inc most rapidly. For J being the least squares error, We can see that $\frac{d}{dt}\Theta = -2X^TX\theta + 2X^TY$. This is linear in θ and can be solved using the normal equations.

PCA: The goal of PCA is to find the directions of maximum variance in the data. We can do this by finding the eigenvectors of the covariance matrix $C = \frac{1}{N}XX^T$. The eigenvectors of C are the principal components, and the eigenvalues are the variances along those components. We can then project the data onto the principal components to reduce the dimensionality.
This is due to the fact that we can first center the data, then consider variance as the sum of squares. which is just $||X||^2$. We can then consider the covariance matrix $C = \frac{1}{N}XX^T$. The eigenvectors of C are the directions of maximum variance. Choosing the largest eigenvalues gives us the directions of maximum variance.

PCA EX: Given a set of points $X = \begin{bmatrix} x_1 & x_2 & \ldots & x_n \end{bmatrix}$, we can compute the covariance matrix $C = \frac{1}{N}XX^T$. We can then find the eigenvectors and eigenvalues of C. The eigenvectors with the largest eigenvalues are the principal components. We can then project the data onto these components to reduce the dimensionality.

SVM: A support vector machine splits the data into two classes by finding the hyperplane that maximizes the margin between the two classes. We essentially want a hyperplane $f(x) = sign(w^T x + b)$ for some w and b. The values where $f(x)$ is undefined are $w^T x + b = 0$. and the set of these values H are our hyperplane: $b + \sum w_i x_i = 0$. Notice that w is orthogonal to the hyperplane. Given a set of points, we may not even be able to find a hyperplane that separates the two classes perfectly.
Given P, N such that such a hyperplane f exists, the best f maximizes the margin between the two classes, but it minimizes the distance to the hyperplane. We can see that $\max r$ st $y_n(w^T x_n + b) \geq r$ for all n is equivalent to $\min \frac{1}{2}||w||^2$ st $y_n(w^T x_n + b) \geq 1$.

SVM EX: Given the points $P = \{(-1,0),(0,1),(-3,4)\}$, $N = \{(0,-1),(1,0),(2,0),(3,-5)\}$ We can consider the constraints either the system of equations $y_i(w^T x_i + b) \geq r$. Then solving the system boils down to $0 \leq rleq \min(y_i(w^T x_i + b))$ some examples of the equations are

$$\begin{aligned}
1(w^T(-1,0) + b) &\geq r \\
1(w^T(0,1) + b) &\geq r \\
-1(w^T(0,-1) + b) &\geq r \\
-1(w^T(1,0) + b) &\geq r \\
-1(w^T(2,0) + b) &\geq r \\
-1(w^T(3,-5) + b) &\geq r
\end{aligned}$$

where $w = \begin{bmatrix} u \\ v \end{bmatrix}$

SVM cont. When we want to consider the same problem as beforehand we can write the Gram Matrix $K(x_i, x_j) = \langle x_i, x_j \rangle$. We can also consider the Lagrangian dual problem: $\min \frac{1}{2}||w||^2 + C\sum_{i=1}^N \xi_i$ st $y_i(w^T x_i + b) \geq 1 - \xi_i$ for all i. The Lagrangian dual is definied as $\min_\alpha \sum_{i=1}^N \sum^N y_i y_j \alpha_i \alpha_j \langle x_i, x_j \rangle - \sum^N \alpha_i$ subject to $\sum^N y_i \alpha_i = 0$ and $0 \leq \alpha_i \leq C$ for all i. We get this by considering the Lagranian $L(w, b, \alpha) = \frac{1}{2}||w||^2 - \sum_{i=1}^N \alpha_i(y_i(w^T x_i + b) - 1)$ and the dual problem is given by $\max_\alpha \min_{w,b} L(w, b, \alpha)$ and this simplifies $\min_\alpha \sum_{i=1}^N \sum^N y_i y_j \alpha_i \alpha_j \langle x_i, x_j \rangle - \sum^N \alpha_i$. We know by partials that $\frac{\partial L}{\partial w} = 0 \implies w = \sum^N \alpha_i y_i x_i$. Ie when x_i is not a support vector $\alpha_i = 0$ and $\frac{\partial L}{\partial b} = \alpha^T y = 0$.

GMM: A Gaussian Mixture Model is a probabilistic model that assumes that the data is generated from a mixture of several Gaussian distributions.
Suppose we have N data points $x_1, x_2, \ldots, x_N$ and we want to fit a GMM with K components. We can write the GMM as: $p(x) = \sum^K w_i p_i$ with $p_i(x) = \frac{1}{\sqrt{2\pi v_i}} \exp(-\frac{(x-\mu_i)^2}{2v_i})$ where w_i is the weight of the ith component, μ_i is the mean of the ith component, and v_i is the variance of the ith component. with $w_i \geq 0$ and $\sum_{i=1}^K w_i = 1$. We can solve this using MLE. Considering the entire likelihood function, we have $L(\theta) = \prod_{i=1}^N p(x_i|\theta) = \prod_{i=1}^N \sum_{j=1}^K w_j p_j(x_i)$. For notation, let $L_n = \prod_{i=1}^K L_{n,k}$ Then $ln(L)_{\mu_m} = \sum_{i=1}^N \frac{L_{n\mu_m}}{L_n}$ We can see that this is minimized (ie derivative = 0) when $\mu_m = \frac{\sum_n x_n \frac{L_{n,m}}{L_n}}{\sum_n \frac{L_{n,m}}{L_n}}$ Simlarly $v_m = \frac{\sum_n (x_n - \mu_n)^2 \frac{L_{n,m}}{L_n}}{\sum_n \frac{L_{n,m}}{L_n}}$ for w_i we cannot just set the derivative to 0 since the $\sum w_i = 1$. We can use the Lagrange multiplier method to solve this. thus we have $w_m = \sum \frac{L_{n,m}}{NL_n}$. This is not the full pcker as we need to solve for each term induvidually, but afyter doing the ieration of starting with parameter values, then calculating and repeating we get the best solution. This is the EM (Expectation Maximization) algorithm.