Exercício 2: Verossimilhança, Entropia Cruzada e Regressão Logística

(a) Equivalência entre Maximizar a Verossimilhança e Minimizar a Entropia Cruzada

A função de verossimilhança (likelihood) para um modelo de Regressão Logística com alvos binários $t_n \in \{0, 1\}$, dado um conjunto de pesos $\mathbf{w}$ e features $\boldsymbol{\phi}_n$, é o produto das probabilidades de Bernoulli para cada ponto de dado:

$$p(\mathbf{t}|\mathbf{w}) = \prod_{n=1}^{N} p(t_n|\boldsymbol{\phi}_n, \mathbf{w}) = \prod_{n=1}^{N} y_n^{t_n}(1 - y_n)^{1-t_n},$$

onde $y_n = p(t_n = 1|\boldsymbol{\phi}_n, \mathbf{w}) = \sigma(\mathbf{w}^T\boldsymbol{\phi}_n)$ é a probabilidade prevista de que o alvo seja 1.

É mais conveniente trabalhar com o logaritmo natural da função de verossimilhança, conhecido como log-verossimilhança:

$$\ln p(\mathbf{t}|\mathbf{w}) = \ln\left(\prod_{n=1}^{N} y_n^{t_n}(1 - y_n)^{1-t_n}\right) = \sum_{n=1}^{N} \ln\left(y_n^{t_n}(1 - y_n)^{1-t_n}\right).$$

Usando as propriedades do logaritmo, podemos reescrever a log-verossimilhança como:

$$\ln p(\mathbf{t}|\mathbf{w}) = \sum_{n=1}^{N} \left[t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\right].$$

Maximizar a função de verossimilhança é equivalente a maximizar a sua log-verossimilhança, pois o logaritmo natural é uma função monotonicamente crescente.

A função de erro de entropia cruzada é definida como:

$$E(\mathbf{w}) = -\sum_{n=1}^{N} \left[t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\right].$$

Comparando a log-verossimilhança com a função de erro de entropia cruzada, vemos que:

$$E(\mathbf{w}) = -\ln p(\mathbf{t}|\mathbf{w}).$$

Portanto, **maximizar a função de verossimilhança $\ln p(\mathbf{t}|\mathbf{w})$ é equivalente a minimizar a função de erro de entropia cruzada $E(\mathbf{w})$**.

(b) Derivação do Gradiente $\nabla E(\mathbf{w})$

A função de erro de entropia cruzada é:

$$E(\mathbf{w}) = -\sum_{n=1}^{N} \left[t_n \ln \sigma(\mathbf{w}^T\boldsymbol{\phi}_n) + (1 - t_n) \ln(1 - \sigma(\mathbf{w}^T\boldsymbol{\phi}_n))\right],$$

onde $\sigma(a) = \frac{1}{1+e^{-a}}$ é a função sigmoide logística e $a_n = \mathbf{w}^T\boldsymbol{\phi}_n$.

Primeiro, encontramos a derivada da função sigmoide em relação a seu argumento:

$$\frac{d\sigma(a)}{da} = \sigma(a)(1-\sigma(a)) = y_n(1-y_n) \quad .$$

Agora, derivamos a função de erro $E(\mathbf{w})$ em relação a um componente w_j do vetor de pesos $\mathbf{w}$:

$$\begin{aligned}
\frac{\partial E(\mathbf{w})}{\partial w_j} &= -\sum_{n=1}^{N}\left[t_n \frac{1}{\sigma(\mathbf{w}^T\boldsymbol{\phi}_n)}\frac{\partial \sigma(\mathbf{w}^T\boldsymbol{\phi}_n)}{\partial w_j} + (1-t_n)\frac{1}{1-\sigma(\mathbf{w}^T\boldsymbol{\phi}_n)}\frac{\partial(1-\sigma(\mathbf{w}^T\boldsymbol{\phi}_n))}{\partial w_j}\right] \\
&= -\sum_{n=1}^{N}\left[t_n \frac{1}{y_n} y_n(1-y_n)\frac{\partial(\mathbf{w}^T\boldsymbol{\phi}_n)}{\partial w_j} + (1-t_n)\frac{1}{1-y_n}(-y_n(1-y_n))\frac{\partial(\mathbf{w}^T\boldsymbol{\phi}_n)}{\partial w_j}\right] \\
&= -\sum_{n=1}^{N}[t_n(1-y_n)\phi_{nj} - (1-t_n)y_n\phi_{nj}] \\
&= -\sum_{n=1}^{N}[t_n\phi_{nj} - t_n y_n\phi_{nj} - y_n\phi_{nj} + t_n y_n\phi_{nj}] \\
&= -\sum_{n=1}^{N}[t_n\phi_{nj} - y_n\phi_{nj}] \\
&= \sum_{n=1}^{N}(y_n - t_n)\phi_{nj}.
\end{aligned}$$

O gradiente $\nabla E(\mathbf{w})$ é um vetor cujos componentes são $\frac{\partial E(\mathbf{w})}{\partial w_j}$. Portanto, podemos escrever o gradiente em forma vetorial como:

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N}(y_n - t_n)\boldsymbol{\phi}_n \quad .$$

Conveniência para Otimização Baseada em Gradiente: Esta forma do gradiente é **extremamente conveniente para otimização baseada em gradiente**.

- **Interpretação Clara do Erro:** O termo $(y_n - t_n)$ representa o erro entre a previsão do modelo (y_n) e o valor alvo verdadeiro (t_n) para o n-ésimo exemplo. O gradiente é, portanto, uma soma ponderada dos vetores de features, onde os pesos são os erros de previsão.
- **Algoritmos de Otimização:** Esta forma do gradiente é diretamente utilizada em algoritmos de otimização de primeira ordem, como o **gradiente descendente (gradient descent)**, onde os pesos são atualizados iterativamente na direção oposta ao gradiente: $\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \eta\nabla E(\mathbf{w}^{(k)})$, com η sendo a taxa de aprendizado.
- **Variantes do Gradiente Descendente:** A forma do gradiente também se adapta bem a variantes como o **gradiente descendente estocástico (stochastic gradient descent - SGD)**, onde o gradiente é aproximado usando apenas um subconjunto (ou um único) ponto de dado em cada iteração, e o **gradiente descendente em mini-batch**, que usa pequenos lotes de dados. A contribuição de cada ponto de dado para o gradiente é isolada, facilitando essas implementações.

A simplicidade e a relação direta com o erro de previsão tornam esta forma do gradiente fundamental para o treinamento eficiente de modelos de regressão logística.

(c) Overfitting em Dados Linearmente Separáveis e Regularização L_2

Problema de Overfitting: Em problemas de classificação binária onde os dados são **linearmente separáveis** no espaço de features $\boldsymbol{\phi}$, a Regressão Logística treinada via Máxima Verossimilhança pode sofrer de **overfitting**. Isso ocorre porque a solução de máxima verossimilhança busca ajustar perfeitamente os dados de treinamento. Em dados linearmente separáveis, isso significa que o algoritmo tentará encontrar um hiperplano de decisão que separa completamente as duas classes com uma margem cada vez maior, fazendo com que a magnitude do vetor de pesos $\mathbf{w}$ tenda ao infinito.

Quando a magnitude de $\mathbf{w}$ se torna muito grande, a função sigmoide $\sigma(\mathbf{w}^T\boldsymbol{\phi}_n)$ se torna extremamente íngreme em torno da fronteira de decisão ($\mathbf{w}^T\boldsymbol{\phi}_n = 0$). Isso faz com que as probabilidades previstas (y_n)

para os pontos de treinamento se aproximem de 0 ou 1 com alta confiança, mesmo para pontos que estão muito próximos da fronteira de decisão. Embora isso leve a um erro de classificação muito baixo (ou zero) nos dados de treinamento, o modelo se torna **excessivamente adaptado ao ruído e às particularidades dos dados de treinamento**, generalizando mal para novos dados não vistos. Além disso, pode haver um **contínuo de hiperplanos separadores**, e a máxima verossimilhança não fornece um critério para escolher um em detrimento de outro.

Como a Regularização L_2 (Ridge) Ajuda: A regularização L_2, também conhecida como **Ridge regression** ou **weight decay** no contexto de redes neurais, adiciona um termo de penalidade à função de erro original. A função de erro regularizada com L_2 é:

$$E_{regularizado}(\mathbf{w}) = -\sum_{n=1}^{N} [t_n \ln y_n + (1 - t_n) \ln(1 - y_n)] + \frac{\lambda}{2}\|\mathbf{w}\|^2,$$

onde $\lambda > 0$ é o **parâmetro de regularização** que controla a força da penalidade, e $\|\mathbf{w}\|^2 = \mathbf{w}^T\mathbf{w} = \sum_j w_j^2$ é o quadrado da norma L_2 do vetor de pesos.

A regularização L_2 ajuda a mitigar o problema de overfitting da seguinte maneira:

- **Penaliza Pesos Grandes:** O termo de penalidade $\frac{\lambda}{2}\|\mathbf{w}\|^2$ penaliza valores grandes para os pesos. Durante o processo de minimização da função de erro regularizada, o otimizador busca um equilíbrio entre ajustar os dados de treinamento (minimizando o primeiro termo, a entropia cruzada) e manter os pesos pequenos (minimizando o segundo termo, a penalidade L_2).

- **Previne o Crescimento Descontrolado dos Pesos:** Em dados linearmente separáveis, a penalidade L_2 impede que os pesos cresçam indefinidamente, o que, por sua vez, evita que a função sigmoide se torne excessivamente íngreme. Isso resulta em uma **fronteira de decisão mais suave** e menos sensível a pontos individuais de treinamento.

- **Melhora a Generalização:** Ao restringir a magnitude dos pesos, a regularização L_2 reduz a complexidade do modelo, tornando-o menos propenso a se ajustar ao ruído nos dados de treinamento e, assim, **melhorando o desempenho de generalização** para dados não vistos.

- **Escolha de Soluções:** A regularização L_2 introduz uma preferência por soluções com pesos menores, ajudando a selecionar uma solução mais estável e generalizável entre o possível contínuo de soluções de máxima verossimilhança em dados linearmente separáveis.

A magnitude do parâmetro de regularização λ determina o quão forte é a penalidade sobre os pesos. Um λ maior leva a pesos menores e a um modelo mais regularizado. A escolha de um valor apropriado para λ é crucial e geralmente é feita usando técnicas de validação cruzada.