Solución

1-)

- Clasificador Naive bayes

Basado en el modelo de bayes asume que las características son independientes entre si y que siguen una distribución normal (gaussiana)

$$p(x_i|A) = p(x|A)p(A) = \prod_{j=1}^{p} p(x_j|A)p(A)$$

Se asume una distribución gaussiana para

$$p(x_k|A_i) = \frac{1}{\sqrt{2\pi\sigma_{k_i}^2}} \exp\left(-\frac{(x_i - \mu_{k_{i,i}})}{2\sigma_{k_i i}^2}\right)$$

La función de predicción se define como:

$$\hat{y} = f(\bar{X}) = \text{argmax} \prod_{j=1}^{p} p(x_j|A_c)p(A_c)$$

Naive bayes no posee un problema de optimización específico sin embargo se basa en estimar medias y varianzas

- Parametros

$$\mu_{ki} = \frac{1}{|N_A|} \sum_{i=1}^{NA} x_i$$

$$\sigma_{A,j}^2 = \frac{1}{N_A} \sum_{j=1}^{N_A} (x_j^{(i)} - \mu_{A,j})^2$$

- SGD (gradient descent) classifier

considerando la optimización estocástica donde se minimiza el valor esperado

$$L(\theta) = E_{q(\xi)}[L(\theta, \xi)]$$

en cada iteración observamos $L_t(\theta) = L(\theta, \xi_t)$

$\xi_t \sim q$, estimamos la estimación del gradiente de L; $q(\xi)$ es independiente de los parámetros como $g_t = \nabla_\theta L_t(\theta_t)$

el algoritmo para actualizar es

$$\theta_{t+1} = \theta_t - \eta_t \nabla_\theta L(\theta_t, \xi_t) = \theta_t - \eta_t g_t$$

Para regresión lineal Modelo $(x_n^T \theta - y_n)$

la función objetivo (costo) toma la forma

$$L(\theta) = \frac{1}{2N} \sum_{n=1}^{N} (x_n^T \theta - y_n)^2 = \frac{1}{2N} \|X\theta - y\|_2^2$$

el gradiente

$$g_t = \frac{1}{N} \sum_{n=1}^{N} (\theta_t^T x_n - y_n) x_n$$

la actualización de parámetros θ basada en el gradiente (Así resolver el problema de minimización de la función de pérdida SGD)

$$\theta_{t+1} = \theta_t - \eta_t (\theta_t^T x_n - y_n) x_n$$

- Regresión logística

Utilizamos función lineal para tomar decisiones

$$f(x) = w^T x + b$$

La regresión logística transforma la salida de la función lineal $f(x)$ en una probabilidad utilizando la función sigmoide

$$\sigma(f(x)) = \frac{1}{1+e^{-f(x)}}$$

La función de pérdida logística

$$L(w,b) = \frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\sigma(f(x_i))) + (1-y_i)\log(1-\sigma(f(x_i)))\right]$$

por lo que el problema de optimización consiste en minimizar la función de pérdida logística

$$\min_{w,b}\left(L(w,b) + \gamma R(w)\right)$$

y utiliza los algoritmos de optimización SGD gradiente descendente o métodos como newton para minimizar w y b de la función logística

$$w \leftarrow w - \eta \nabla_w L(w,b)$$

$$b \leftarrow b - \eta \nabla_b L(w,b)$$

- Linear Discriminant Analysis (LDA)

busca una proyección lineal de las características originales que maximiza la separabilidad entre clases

$$P(y=k|x) = \frac{\exp\left(-1/2\,(x-\mu_k)^T \Sigma^{-1}(x-\mu_k)\right)}{(2\pi)^{d/2}|\Sigma|^{1/2}}$$

μ_k es la media de la clase k, Σ es la matriz de covariancia común y d la dimensión del espacio

El objetivo es encontrar un hiperplano que maximiza la separación entre clases

$$w^* = \arg\max_w \frac{w^T S_B w}{w^T S_W w}$$

y se soluciona

$$S_w^{-1} S_B w = \lambda w$$

- K-Neighbors Classifier

Dado un conjunto de datos $\{(x_1, y_1), (x_2, y_2) \dots (x_n, y_n)\}$ se calcula una distancia $d(x_{new}, x_i)$, por ejemplo usando métrica euclidiana

$$d(x_{new}, x_i) = \left(\sum_{j=1}^{p} |x_{new,j} - x_{ij}|^q\right)^{1/q}$$

Solución de vecinos cercanos x_{new}

$$N_k(x_{new}) = \{x_i \mid x_i \text{ están entre los puntos más cercanos } x_{new}\}$$

la clase y_{new} se asigna como la clase ~~del~~ más frecuente entre los k vecinos

$$y_{new} = \underset{c \in \{1,2,\dots C\}}{\operatorname{argmax}} \sum_{i \in N_k(x_{new})} \mathbb{I}(y_i = c)$$

$\mathbb{I}$ → función indicador $\begin{cases} 1 \text{ si } y_i = c \\ 0 \text{ en otro caso} \end{cases}$

el conjunto de error para un valor dado de k

$$error(k) = \frac{1}{m} \sum_{i=1}^{m} \mathbb{I}(y_i \neq \hat{y}_i(k))$$

donde $\hat{y}_i(k)$ es la clase predicha

el valor optimo de k es

$$k^* = \underset{k \in N}{\operatorname{arg\,min}} \, error(k)$$

- SVC

se debe encontrar un hiper-plano definido por

$$w^T x + b = 0 \quad y \; y_n \in \{-1, +1\}$$

es un problema de optimización cuadrática con restricciones

el objetivo es minimizar la norma del vector w

para maximizar el margen

$$\min_{w,b} \frac{1}{2} \|w\|_2^2 \quad s.t. \; y_n(f(x_n) - 1) \geq 1 \; \forall n \in \{1,2,\dots N\}$$

La mayoría de problemas presentan un traslape entre clases no es posible encontrar un margen duro entonces

$$\min_{w,b} = \frac{1}{2} \|w\|_2^2 + C \sum_{n=1}^{N} \xi_n$$

$$\text{s.t. } y_n (f(x_n) - 1) \geq 1 - \xi_n$$

$$\xi_n \geq 0$$

el problema dual se obtiene introduciendo multiplicadores de lagrange

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j$$

- α_i multiplicadores de lagrange
- $x_i^T x_j$ es el producto interno

el problema no lineal para SVC se utiliza kernel trick con función de decisión

$$f(x_{new}) = \text{sign}\left(\sum_{i=1} \alpha_i y_i k(x_i, x_{new}) + b \right)$$

- Random forest classifier

Su principio se centra en la generación de fronteras mediante umbralización sobre las características

- consiste en un conjunto árboles de decisión $\{h_1, h_2, \dots h_T\}$

el entrenamiento de h_t se realiza seleccionando un subconjunto de n' muestras $(n' \leq n)$ del conjunto D, en cada división del árbol, la predicción final se obtiene por votación mayoritaria

$$y_{new} = \arg\max_{c \in \{1,2,\dots C\}} \sum_{t=1}^{T} \mathbb{I}(h_t(X_{new}) = c)$$

los criterios de división basan su aprendizaje en la medida Gini impurity

$$G_i = 1 - \sum_{k=1}^{n} p_{ik}^2$$

- Gaussian process classifier

Función latente que sigue el proceso gaussiano

$$f(x) \sim GP(m(x), k(x, x'))$$

$m(x)$ función de media $m(x) = 0$
$k(x, x')$ kernel

la probabilidad pertenece a la clase 1, se utiliza función de enlace link function

$$P(y=1 \mid f(x)) = \sigma(f(x)) = \sigma) = \frac{1}{1 + e^{-f(x)}}$$

el objetivo es inferir la distribución posterior de $f(x)$
en texto se maximiza la verosimilitud

$$p(y|x) = \int p(y|f) p(f|x) df$$

los hiper parámetros del kernel $k(x, x')$ se optimizan maximizando la verosimilitud marginal

$$\theta^* = \arg\max p[y|x, \theta]$$

Post data

la IA fue utilizada para explicar de manera general los modelos y adicionalmente para traducir código y plantear los modelos como Gaussian process y K-NN código de Random forest regressor código de complementarios con hiperparámetros teóricos

IAs usadas Deepseek, Chat GPT