Descipción General del código

El código proporcionado realiza las siguientes operciones:

1. **Generación de Datos:** Crea 40 puntos bidimensionales que son linealmente separables. Los primeros 20 puntos están centrado en $[-2, -2]$ y los siguientes 20 en $[2, 2]$, asigándoles etiquetas 0 y 1 respectivamente.
2. **Entrenamiento del modelo SVM**: Utiliza las clases SVC con un núcleo lineal para entrenar dos modelos SVM con diferentes penalizaciones (`C=1` para "no regularizado" y `C=0.05` para "regularizado").
3. **Cálculo del Hiperplano de Separación:** Obtiene los coeficientes del modelo para determinar el hiperplano que separa las dos clases.
4. **Cálculo de los márgenes:** Calcula las líneas paralelas al hiperplano de separación que representan los márgenes de la SVM.
5. **Visualización:** El código nos genera una gráfica del hiperplano de separación, los márgenes, los vectores soporte y los puntos de datos.

Diferencias entre SVC y LinearSVC

Ambas clases, `SVC` y `LinearSVC`, implementan máquinas de vectores de soporte para clasificación lineal, pero difieren en varios aspectos:

1. **Kernel:**
   - `SVC` soporta diferentes tipos de núcleos mediante el parámetro `kernel`. En el caso del código proporcionado se utiliza un núcleo lineal.
   - `LinearSVC` está optimizado exclusivamente para un núcleo lineal y no soporte ninguno de los otros tipos que sí soporta `SVC`.
2. **Optimización del Primal vs el Dual:**
   - `SVC` resuelve el problema dual, lo que lo hace muy eficiente para conjuntos de datos pequeños.
   - `LinearSVC` resuelve el problema primal, lo que lo hace muy eficiente para conjuntos de datos grandes.
3. **Regularización:**
   - `SVC` utiliza la regularización mediante el parámetro `C` que controla la penalización de los errores de clasificación.
   - `LinearSVC` también utiliza el parámetro `C` para la regularización, pero, a diferencia de `SVC`, implementa diferentes estrategias de optimización. En resumen, mientras que `SVC` es más versatil en términos de tipos de kernels y también es mucho más útil para conjuntos de datos pequeños, `LinearSVC` es preferible para grandes conjuntos de datos lineales debido a su eficiencia.

Cálculo del Hiperplano de Separación

El hiperplano de separación es la "línea" que divide el espacio de carácterísticas de manera que las dos clases queden separaadas con el mayor margen posible.

El hiperplano se define con la siguiente ecuación:

$$\omega^{\mathsf{T}} x + b = 0$$

Donde: - ω es el vector de coeficientes del modelo - b es el término independiente. - x es el vector de características.

En el código anterior, una vez habíamos entrenado el modelo `clf`, extraíamos los coeficientes y el término independiente:

```
w = clf.coef_[0]
a = -w[0] / w[1]
yy = a * xx - (clf.intercept_[0] / w[1])
```

Aquí, la línea **`yy = a * xx - (clf.intercept_[0] / w[1]`** define el hiperplano de separación en función de los valores de **`xx`**.

Cálculo de los márgenes

La distancia del margen se calcula utilizando la norma del vector de coeficientes ω:

$$\text{Margen} = \frac{1}{\|\omega\|} = \sqrt{\frac{1}{\omega_1^2 + \omega_2^2}}$$

En el código:

```
margin = 1 / np.sqrt(np.sum(clf.coef_**2))
yy_down = yy - np.sqrt(1 + a**2) * margin
yy_up = yy + np.sqrt(1 + a**2) * margin
```

Las líneas **`yy_down`** y **`yy_up`** representan los márgenes inferiores y superiores, respectivamente, desplazados verticalmente desde el hiperplano de separación.

Vectores soporte

Los vectores soporte son los puntos de datos más cercanos al hiperplano de separación y son cruciales para definir la posición y orientación del hiperplano. Estos puntos "soportan" el hiperplano, ya que cualquier cambio en ellos afectaría directamente la posición del hiperplano.

En el código, los vectores soporte se obtienen mediante:

```
clf.support_vectors_
```

Y se representan gráficamente como puntos con bordes destacados:

```
plt.scatter(
    clf.support_vectors_[:, 0],
    clf.support_vectors_[:, 1],
    s=80,
    facecolors="none",
    zorder=10,
    edgecolors="k",
    cmap=plt.cm.get_cmap("RdBu"),
)
```

Formulación matemática

Problema Primal

$$(Primal)\begin{cases} \text{Minimizar} & \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{n}\xi_i \\ \text{Sujeto a} & y_i\left(\omega^\intercal x_i + b\right) \geq 1 - \xi_i \\ & \xi \geq 0 \quad \forall i \end{cases}$$

Donde:

- ξ_i son las variables de holgura que permiten la violación de las restricciones de margen.
- C es el parámetro de regularización que controla la penalización de los errores de clasificación.

Problema Dual

$$(Dual)\begin{cases} \text{Maximizar} & \sum_{i=1}^{n}\alpha_i + \dfrac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^{\intercal} x_j \\ \text{Sujeto a} & 0 \leq \alpha_i \leq C \quad \forall i \\ & \sum_{i=1}^{n}\alpha_i y_i = 0 \end{cases}$$

Donde:

- α_i son los multiplicadores de Lagrange asociados a las restricciones.