Advanced Application of Artificial Intelligence and Digital Transformation

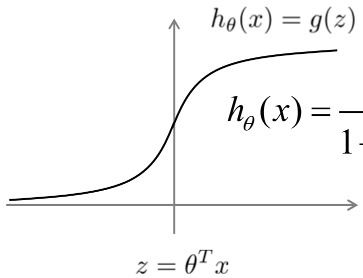
K.N. Toosi University of Technology

Support Vector Machines

رگرسیون

در رگرسیون سعی بر این بود تا بهترین خطی که به داده ها برازش می‌شود پیدا کنیم

$$h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_n x_n \qquad h_\theta = \theta^T X = \sum_0^n \theta_n x_n$$



$$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$$

در رگرسیون لجستیک سعی بر این بود تا داده های را دسته بندی کنیم

آیا این دسته بندی را می‌توان به نحوی دیگر توسعه داد؟

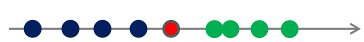
If $y = 1$, we want $h_\theta(x) \approx 1$, $\theta^T x \gg 0$
If $y = 0$, we want $h_\theta(x) \approx 0$, $\theta^T x \ll 0$

این توسعه به نام بردار پشتیبان در سال ۱۹۶۴ توسط آقای Vladimir Vapnik برای طبقه بندی پیشنهاد شد

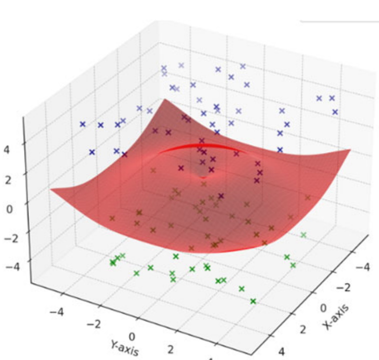
Hyperplane ابر صفحه

در فضا n بعدی یک ابر صفحه دارای n-1 بعد است و فضا را به دو نیم‌فضا تقسیم می کند

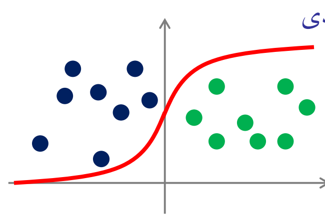
$$f(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n = 0 \qquad f(x) = \beta_0 + \vec{\beta} x = 0$$



فضای یک بعدی



فضای سه بعدی

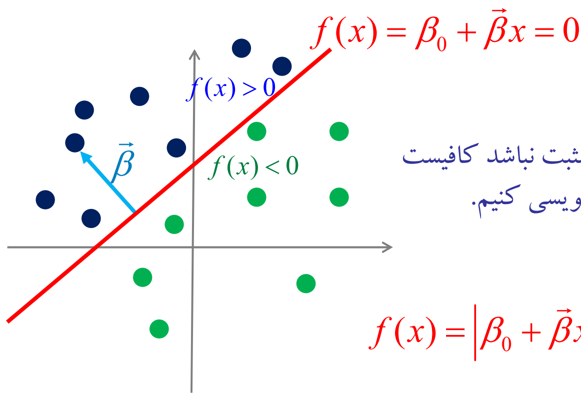


فضای دو بعدی

Hyperplane ابر صفحه

علامت تابع f(x) برای نقاط در دو طرف ابر صفحه (دو نیم فضا) متفاوت است

$$f(x) = \beta_0 + \vec{\beta}x = 0$$



اگر بالای ابرصفحه علامت تابع مثبت نباشد کافیست تابع را با ضرب در یک منفی بازنویسی کنیم.

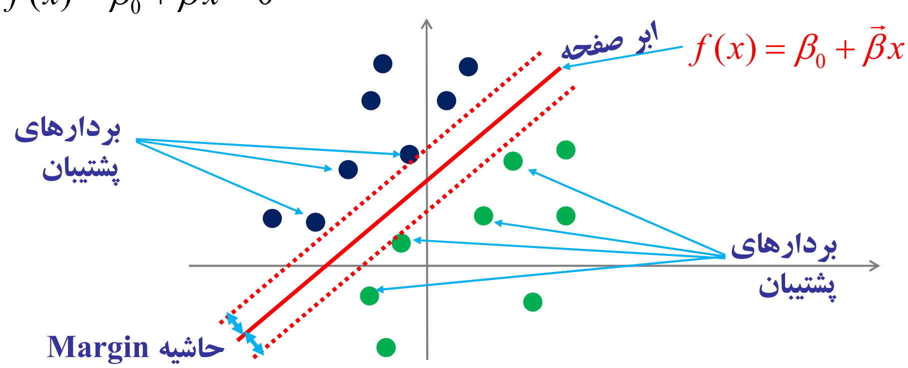
نقاط کلاسها:

$$f(x) = \left|\beta_0 + \vec{\beta}x\right| > 1$$

ماشین بردار پشتیبان SVM

در یادگیری بردار پشتیبان به دنبال یافتن یک ابرصفحه جداکننده هستیم که فاصله بین نزدیکترین داده کلاسها را تا ابرصفحه حداکثر کند

$$f(x) = \beta_0 + \vec{\beta}x = 0$$



بردارهای پشتیبان، نقاط دادهای هستند که نزدیک‌ترین به ابرصفحه جداکننده قرار دارند
الگوریتم SVM از بردارهای پشتیبان برای ساختن ابرصفحه استفاده می کند.

فاصله تا ابر صفحه

اگر دو کلاس قابل تفکیک باشند بینهایت ابرصفحه برای تفکیک وجود دارد.

مقادیر برچسب $y^{(i)} \in \{-1,1\}$

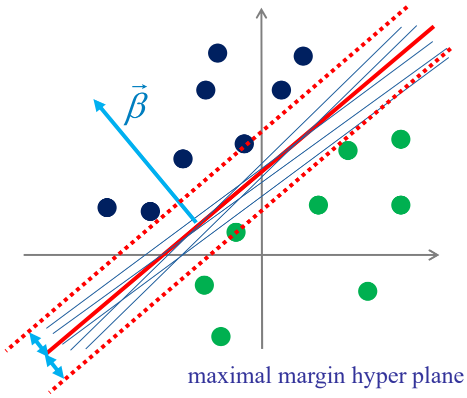
$$f(x) = \beta_0 + \vec{\beta} x = 0$$

با ضرب در مقدار برچسب، کلاسها جدا می شوند و داریم

$$y^{(i)}(\beta^T x^{(i)} + \beta_0) > 0$$

حاشیه حداقل فاصله داده‌های کلاسها تا ابرصفحه است

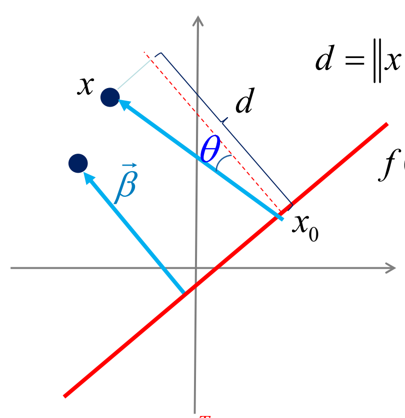
بدنبال ابرصفحه با حداکثر حاشیه هستیم



ابرصفحه با حداکثر حاشیه دارای یک داده از هرکلاس روی حاشیه خود است

فاصله تا ابر صفحه

علامت تابع f(x) برای نقاط در دو طرف ابر صفحه (دو نیم فضا) متفاوت است



$$d = \|x - x_0\| |Cos\theta| = \|x - x_0\| \frac{|\beta^T (x - x_0)|}{\|\beta\| \|x - x_0\|}$$

$$f(x_0) = \beta_0 + \beta^T x_0 = 0 \Rightarrow \beta_0 = -\beta^T x_0$$

$$\Rightarrow d = \frac{|\beta^T x + \beta_0|}{\|\beta\|} = \frac{|f(x)|}{\|\beta\|}$$

$f(x) = \beta_0 + \beta^T x = 0$

بهینه نمودن فاصله تا ابر صفحه

در بردار پشتیبان به دنبال اکسترمم کردن تابع زیر هستیم

$$\max_{\beta,\beta_0}\left\{\min_i\left\{\frac{\left|\beta^T x^{(i)}+\beta_0\right|}{\|\beta\|}\right\}\right\} \rightarrow M \quad \text{حاشیه} \quad M=\min_i\left\{\frac{\left|\beta^T x^{(i)}+\beta_0\right|}{\|\beta\|}\right\}$$

$$subject\ to\quad y^{(i)}(\beta^T x^{(i)}+\beta_0)>0 \xrightarrow{\frac{y^{(i)}(\beta^T x^{(i)}+\beta_0)}{\|\beta\|}>\min\frac{y^{(i)}(\beta^T x^{(i)}+\beta_0)}{\|\beta\|}} \frac{y^{(i)}(\beta^T x^{(i)}+\beta_0)}{\|\beta\|}>M$$

مساله بهینه‌سازی بردار پشتیبان :

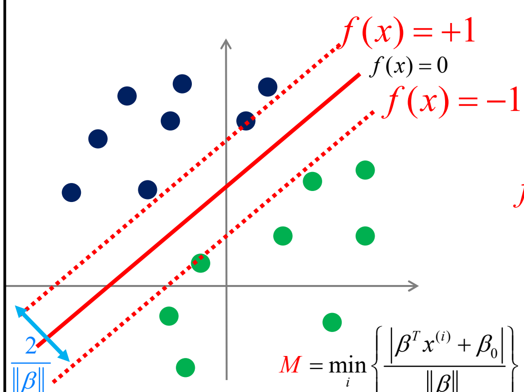
$$\max_{\beta,\beta_0} M \quad s.t.\quad \frac{y^{(i)}(\beta^T x^{(i)}+\beta_0)}{\|\beta\|}>M$$

$x^{(i)}$: The feature vector for the i-th training sample.
$y^{(i)}$: The label for the i-th training sample (+1 or −1).
β: The normal vector to the hyperplane.
β_0: The bias term (intercept) of the hyperplane.
$\|\beta\|$: The norm of the vector β, ensuring normalization.

بردار پشتیبان به دنبال بهترین خط جداکننده با بیشترین حاشیه امن بین کلاس‌ها است.

حاشیه سخت

حاشیه سخت: درون حاشیه نقاط داده هیچ کلاسی وجود ندارد



ابر صفحه: $f(x)=\beta_0+\vec{\beta}x=0$

مرزهای حاشیه مثبت و منفی: مقدار بتا را می توان نرمالایز کرد

$$f(x)=\beta_0+\vec{\beta}x=\pm 1$$

یا: $y^{(i)}(\beta^T x^{(i)}+\beta_0)=1$

بنابراین:

$$M=\min_i\left\{\frac{\left|\beta^T x^{(i)}+\beta_0\right|}{\|\beta\|}\right\}=\frac{1}{\|\beta\|}\rightarrow max M=min\|\beta\|=min\frac{1}{2}\beta^T\beta$$

مساله بردار پشتیبان:

$$\min_{\beta,\beta_0}\frac{1}{2}\beta^T\beta \quad s.t.\quad y^{(i)}(\beta^T x^{(i)}+\beta_0)\geq 1$$

مساله دوگان

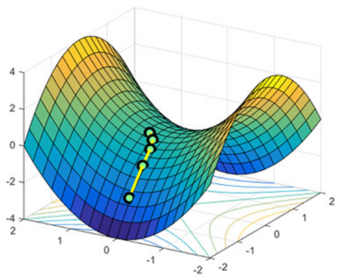
در بهینه سازی ریاضی به جای مساله اصلی می توان مساله دوگان را بهینه نمود که در بهینه سازی مسائل محدب و مقید کاربرد زیادی دارد:

مساله اصلی (Primal problem): $\min_{x} f(x) \qquad s.t.\;\; g(x)=0$

تشکیل تابع لاگرانژ و ضرایب لاگرانژ

$$L(x,\alpha)=f(x)+\alpha g(x) \qquad \alpha \ge 0$$

$$\frac{\partial L(x,\alpha)}{\partial x}=0 \qquad \frac{\partial L(x,\alpha)}{\partial \alpha}=0$$



مساله دوگان معادل (Dual problem):

$$\max_{x,\alpha} L(x,\alpha) \qquad s.t.\; \alpha \ge 0$$

مساله دوگان حاشیه سخت

در بهینه سازی ریاضی به جای مساله اصلی می توان مساله دوگان را حل نمود که در بهینه سازی مسائل محدب و مقید کاربرد زیادی دارد:

مساله اصلی (Primal problem):

$$\min_{\beta,\beta_0} \frac{1}{2}\beta^T \beta \; s.t. \quad y^{(i)}(\beta^T x^{(i)}+\beta_0)-1 \ge 0$$

تشکیل تابع لاگرانژ
و مشتق گیری و ساده سازی

$$L(\beta,\beta_0,\alpha)=\frac{1}{2}\beta^T\beta-\sum_{i=1}^{N}\alpha_i\left[y^{(i)}(\beta^T x^{(i)}+\beta_0)-1\right]$$

مساله دوگان (Dual problem):

$$\max_{\alpha} \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y^{(i)}y^{(j)}(x^{(i)}x^{(j)})$$

$$\sum_{i=1}^{N}\alpha_i y^{(i)}=0 \quad and \quad \alpha_i \ge 0$$

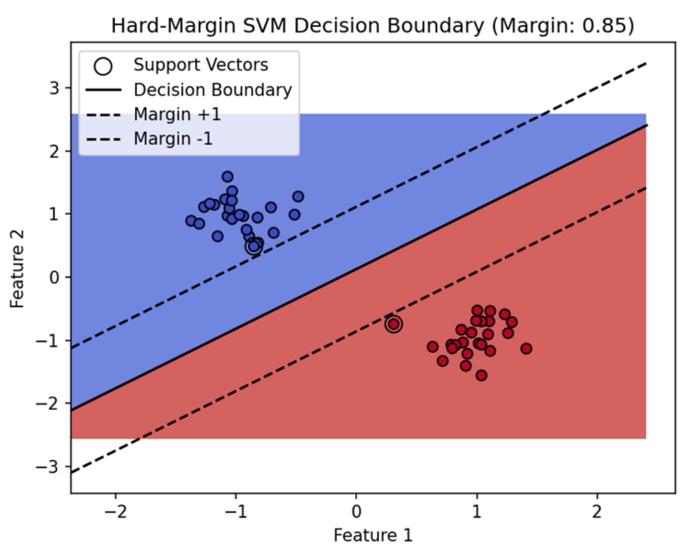
حل مساله دوگان به دلیل وجود صرفا ضرب داخلی ویژگیها و وابستگی به تعداد ویژگیها نه ابعاد آنها راحتتر است و در مسایل غیر خطی و ابعاد بالا قابل استفاده است

مثال حاشیه سخت

فایل: 11 SVM Hard.py

```python
from sklearn.svm import SVC
from sklearn.datasets import make_blobs
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report

# Generate a synthetic dataset
X, y = make_blobs(n_samples=50, centers=2, random_state=42)
y = 2 * y - 1  # Convert labels to {-1, 1}

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=42)

# Create and train the hard-margin SVM
hard_margin_svm = SVC(C=1e6, kernel='linear')
hard_margin_svm.fit(X_train, y_train)

# Evaluate the model
y_pred = hard_margin_svm.predict(X_test)
print(classification_report(y_test, y_pred))
```

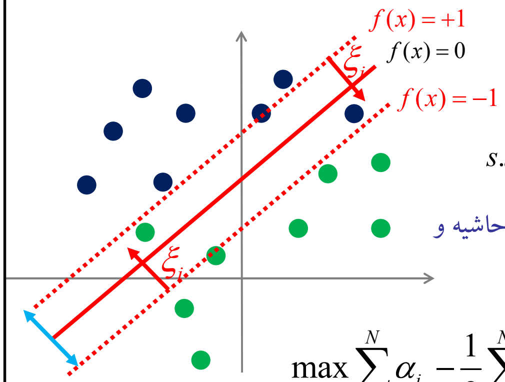
	precision	recall	f1-score	support
-1	1.00	1.00	1.00	6
1	1.00	1.00	1.00	9
accuracy			1.00	15
macro avg	1.00	1.00	1.00	15
weighted avg	1.00	1.00	1.00	15



Formula of the decision boundary: 0.81 * x1 + -0.86 * x2 + 0.11 = 0

حاشیه نرم

حاشیه نرم: اگر با کنترل اجازه داده شود بعضی نقاط داده از حاشیه عبور کنند بردار پشتیبان بصورت پایدارتری برای داده های نوفه دار و غیرخطی قابل حل است



مساله اصلی (Primal problem):

$$\min_{\beta,\beta_0,\xi} \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\xi_i$$

$$s.t.\quad y^{(i)}(\beta^T x^{(i)}+\beta_0) \geq 1-\xi_i \qquad \xi_i \geq 0$$

$C>0$ پارامتر جریمه برای حداکثر کردن حاشیه و حداقل کردن خطای کلاس بندی

مساله دوگان (Dual problem):

$$\max_{\alpha} \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y^{(i)}y^{(j)}(x^{(i)}x^{(j)})$$

$$\sum_{i=1}^{N}\alpha_i y^{(i)} = 0 \quad and \quad 0\leq\alpha_i\leq C$$

حاشیه نرم

تاثیر پارامتر تنظیم C

مقادیر کوچک -تنظیم زیاد:

۱. بزرگ‌تر کردن حاشیه بین کلاس‌ها.
۲. تعداد بیشتری از داده‌ها به اشتباه طبقه‌بندی شوند
۳. منجر به مدلی ساده‌تر می‌شود که ممکن است داده‌ها را کم‌برازش کند.

مقادیر بزرگ - تنظیم کم:

۱. طبقه‌بندی اشتباه را بیشتر جریمه می‌کند. و عبور از حاشیه کمتری ایجاد می‌شود
۲. ممکن است مدل بسیار پیچیده شود و به بیش‌برازش دچار شود.

تعیین پارامتر تنظیم C

- **پارامتر پیش‌فرض:** از پیش‌فرض C=1 شروع کنید
- **مقیاس لگاریتمی :** مقادیر C را روی مقیاس لگاریتمی بررسی کنید. $C \in \{10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 100, 1000\}$.
- **بررسی نتایج :** دقت مدل را روی مجموعه آموزش و اعتبارسنجی بررسی کنید (کنترل صحت و دقت یا F1).

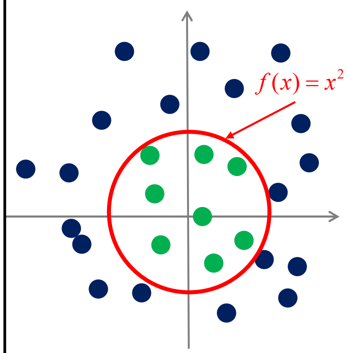
اعتبار سنجی ضربدری: برای تعیین C بهینه می توان با تقسیم مجموعه داده های آموزش به k دسته (مثلا ۵) و آموزش k-1 دسته و اعتبارسنجی با دسته آخر برای مقادیر مختلف C انجام داد. برای هر C میانگین دقت k-1 دسته تعیین شده و بر اساس بهترین دقت C تعیین می شود.

```
from sklearn.model_selection import cross_val_score
scores = cross_val_score(estimator, X, y, cv=5, scoring='accuracy')
```

بردار پشتیبان برای مسایل غیرخطی

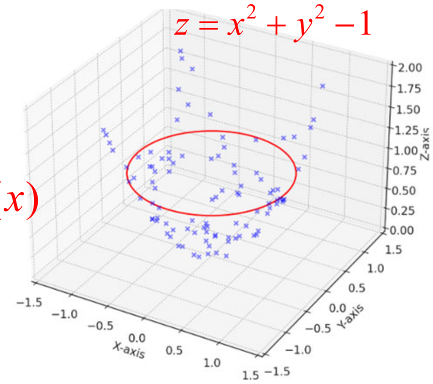
اگر با خط امکان جدا کردن داده‌ها نباشد از ابعاد بالاتر می‌توان استفاده کرد

$$f(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n \qquad f(x) = \beta_0 + \vec{\beta} x$$



$$T : x \rightarrow \phi(x)$$

نگاشت ویژگی
به ابعاد بالاتر



کرنل ماشین بردار پشتیبان

اگر با خط امکان جدا کردن داده‌ها نباشد از ابعاد بالاتر می‌توان استفاده کرد

مساله اصلی (Primal problem):

$$\min_{\beta,\beta_0,\xi} \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\xi_i$$

$$s.t. \quad y^{(i)}(\phi(x^{(i)})^T \beta + \beta_0) \geq 1 - \xi_i \qquad \xi_i \geq 0$$

$\phi(x^{(i)})$ تابع نگاشت ویژگی به ابعاد بالاتر

مساله دوگان (Dual problem):

$$\max_{\alpha} \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y^{(i)} y^{(j)} K(x^{(i)}, x^{(j)})$$

بجای ضرب ویژگیها کرنل قرار می گیرد

$$\sum_{i=1}^{N}\alpha_i y^{(i)} = 0 \quad and \quad 0 \leq \alpha_i \leq C$$

تابع کرنل

$$K(x^{(i)}, x^{(j)}) = \phi(x^{(i)})^T \phi(x^{(j)})$$

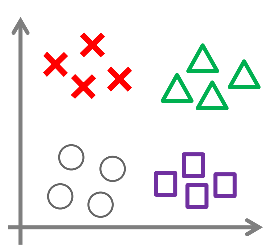
کرنل ماشین بردار پشتیبان

انواع کرنل

Kernel	Equation	Use Case	Pros	Cons
Linear	$K = x.x'$	Linearly separable data	Simple, fast, interpretable	Limited to linear relationships
RBF: Radial Basis Function	$K = e^{-\gamma\|x-x'\|^2}$	Non-linearly separable data	Highly flexible	Can overfit with large γ
Polynomial	$K = (\gamma x.x' + r)^d$	Polynomial feature relationships	Flexible, interpretable	High-degree polynomials may overfit
Sigmoid	$K = \tanh(\gamma(x.x') + r)$	Neural network-like transformations	Flexible	Less robust than RBF

بردار پشتیبان چندکلاسه

در حالتیکه بیش از یک کلاس داشته باشیم:



$$y \in \{1, 2, 3, \ldots, K\}$$

Many SVM packages already have built-in multi-class classification functionality.

بردار پشتیبان چندکلاسه

در حالتیکه بیش از یک کلاس داشته باشیم:

One-vs-Rest (OvR): For K classes, train K binary classifiers.

Advantages: Simple to implement and Efficient when K is small.
Disadvantages: Overlaps in decision boundaries can occur and less effective for imbalanced classes.

One-vs-One (OvO): For K classes, train K(K−1)/2 binary classifiers.

Advantages: Often performs better than OvR and More granular decision boundaries.
Disadvantages:Computationally expensive for large K, due to many classifiers.

Direct Multi-Class SVM (All-vs-All):

- A single optimization problem is formulated to classify multiple classes.
- This approach solves a large optimization problem for all classes simultaneously.
- Less commonly used due to computational complexity.

مثال حاشیه نرم

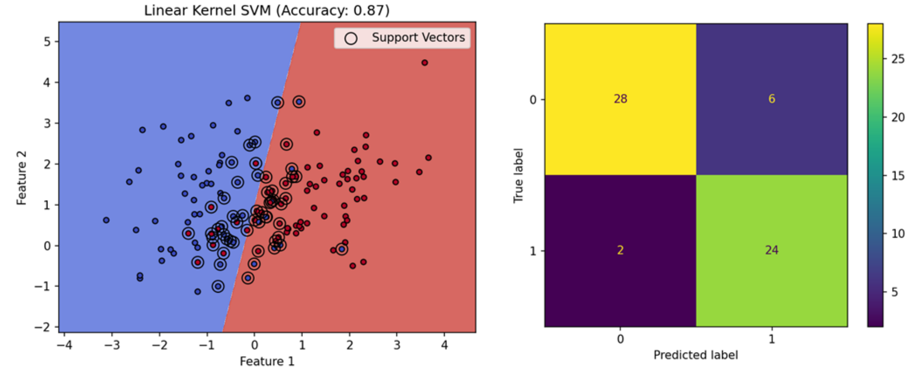
فایل:11 SVM Soft.py

مساله بهینه‌سازی بردار پشتیبان :

```
# Function to train, evaluate, and plot SVM for different kernels
def train_evaluate_plot(kernel, C=1.0, degree=3):
    # Train the SVM model with a specific kernel
    svm = SVC(C=C, kernel=kernel, gamma='scale', degree=degree)
    svm.fit(X_train, y_train)

    # Predict and calculate accuracy
    y_pred = svm.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)

    # Plot decision boundary
    plot_svm_with_margin(X_train, y_train, svm, kernel.capitalize(),
accuracy)

# Test the function with different kernels
for kernel in ['linear', 'rbf', 'poly']:
    train_evaluate_plot(kernel, C=1.0, degree=3)
```

مثال حاشیه نرم

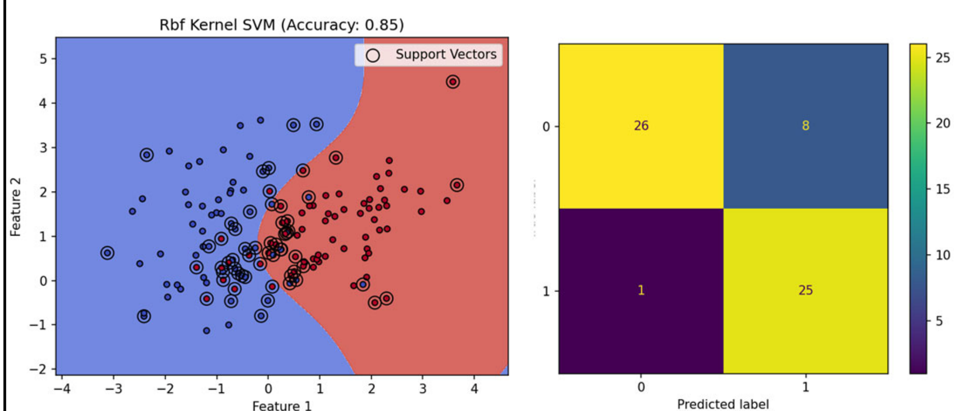
$K = x.x'$



`Kernel: linear, C: 1.0, Degree: 3, Gamma: 0.5`

مثال حاشیه نرم

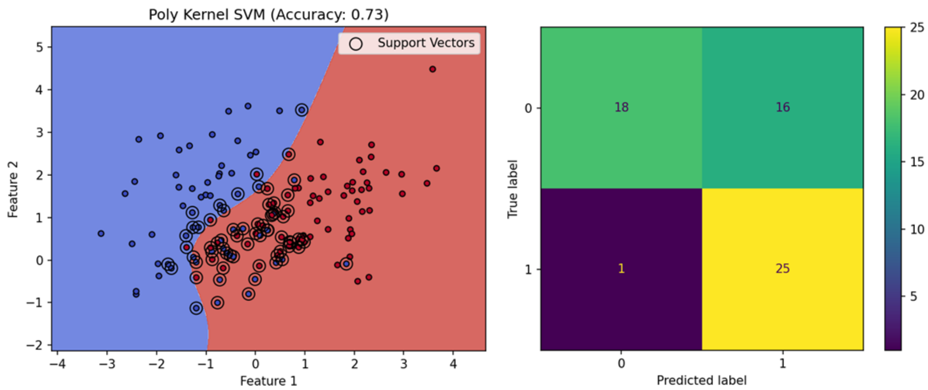
$K = e^{-\gamma\|x-x'\|^2}$



`Kernel: rbf, C: 1.0, Degree: 3, Gamma: 0.5`

مثال حاشیه نرم

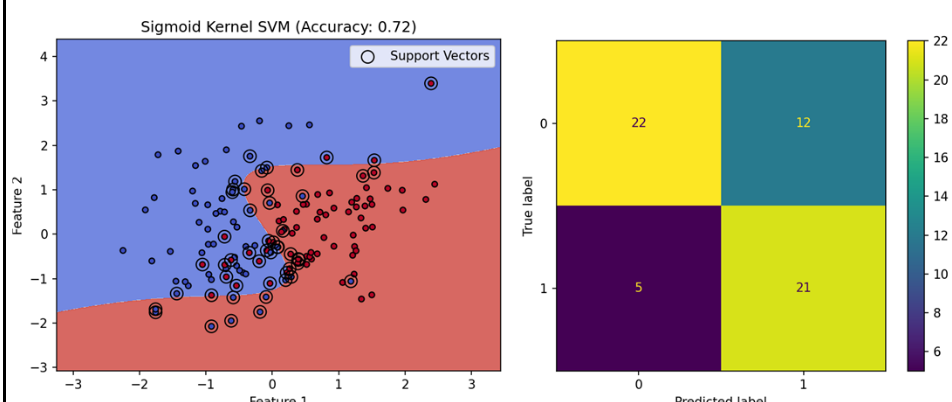
$K = (\gamma x.x' + r)^d$



`Kernel: poly, C: 1.0, Degree: 3, Gamma: scale, Coef0 (r): 0.0`

مثال حاشیه نرم

$K = \tanh(\gamma(x.x') + r)$



`sigmoid, C: 1.0, Degree: 3, Gamma: 0.5`

مقایسه روشها

تمرین برنامه نویسی

تمرین ششم : یک برنامه به زبان پایتون بنویسید که یک فایل داده را خوانده و به روش بردار پشتیبان با حاشیه نرم و کرنل تابع پایه شعاعی و چند جمله‌ای کلاس بندی انجام دهد.

۱- خواندن فایل داده

۲- طبقه بندی با بردار پشتیبان با حاشیه نرم

۳- بررسی هایپر پارامترهای کرنل تابع پایه شعاعی و چند جمله ای

۴- تاثیر هایپر پارامترهای مدل با ترسیم آنها

۵- پیشنهاد بهترین دسته بندی با هایپرپارامترهای مربوطه