

Supervised Learning Bagian - II

18 April 2024



Materi Hari Ini



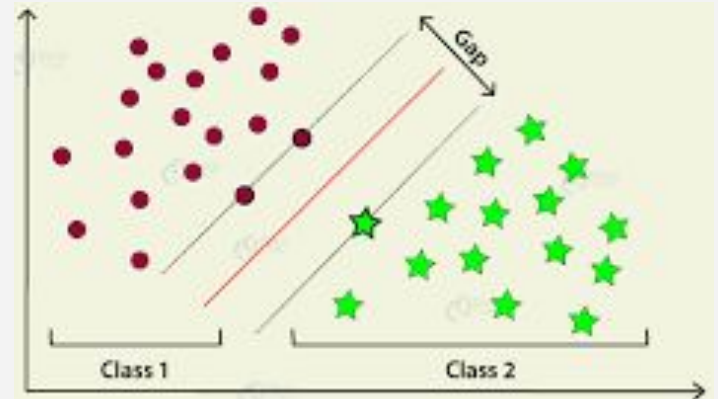
Kita akan mempelajari antara lain:

1. Pengantar SVM (Support Vector Machine)
2. Multiclass Classifier

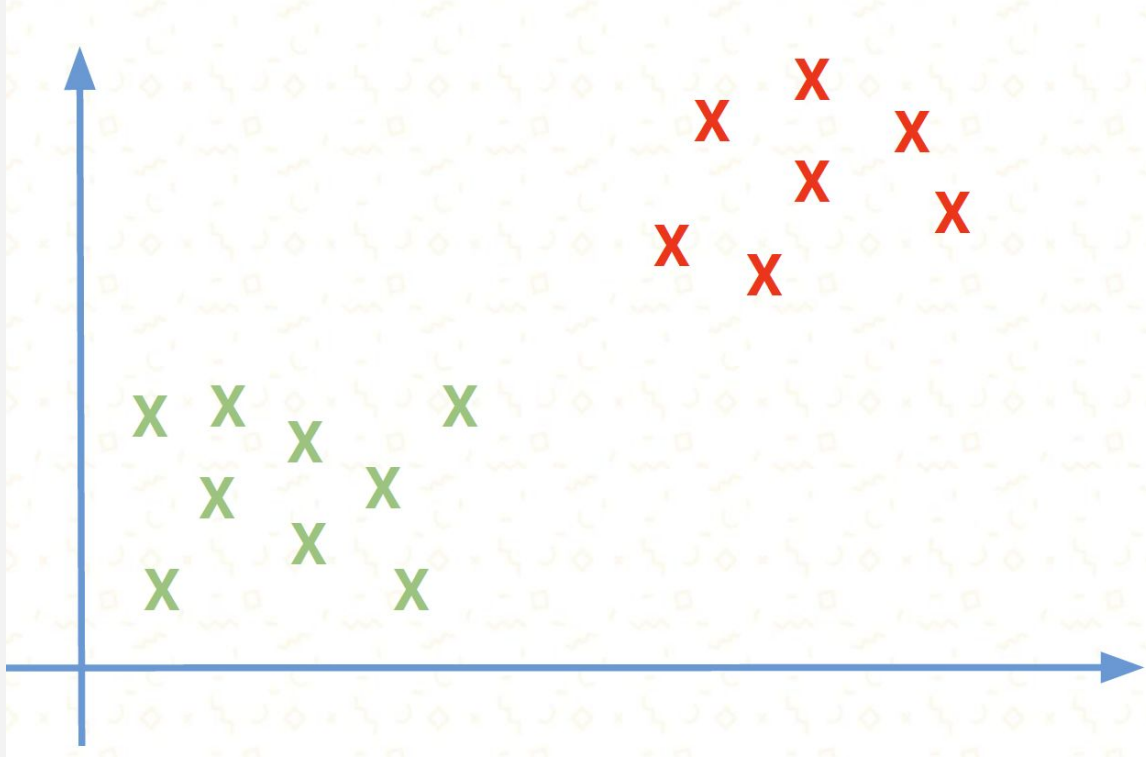


Pengantar SVM (Support Vector Machine)

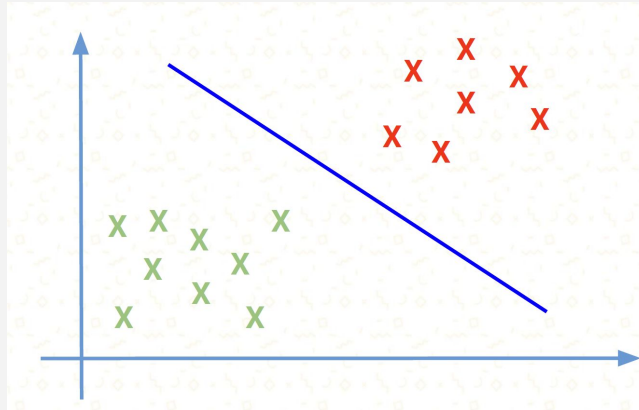
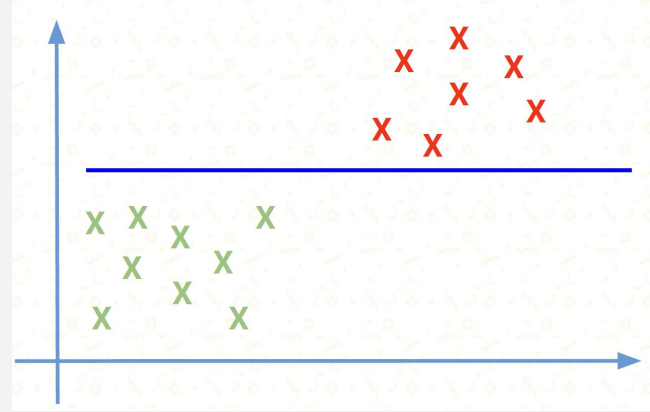
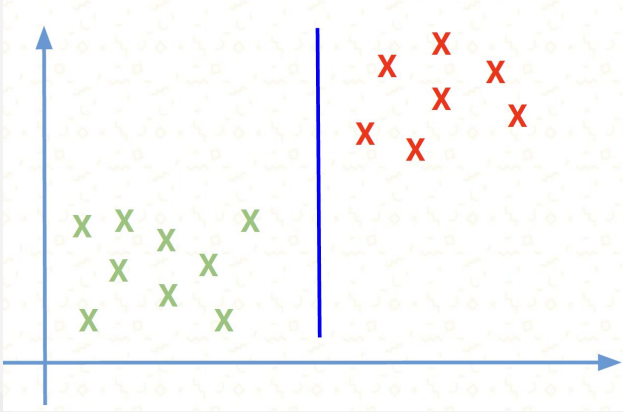
Apa dan Bagaimana



Permasalahan



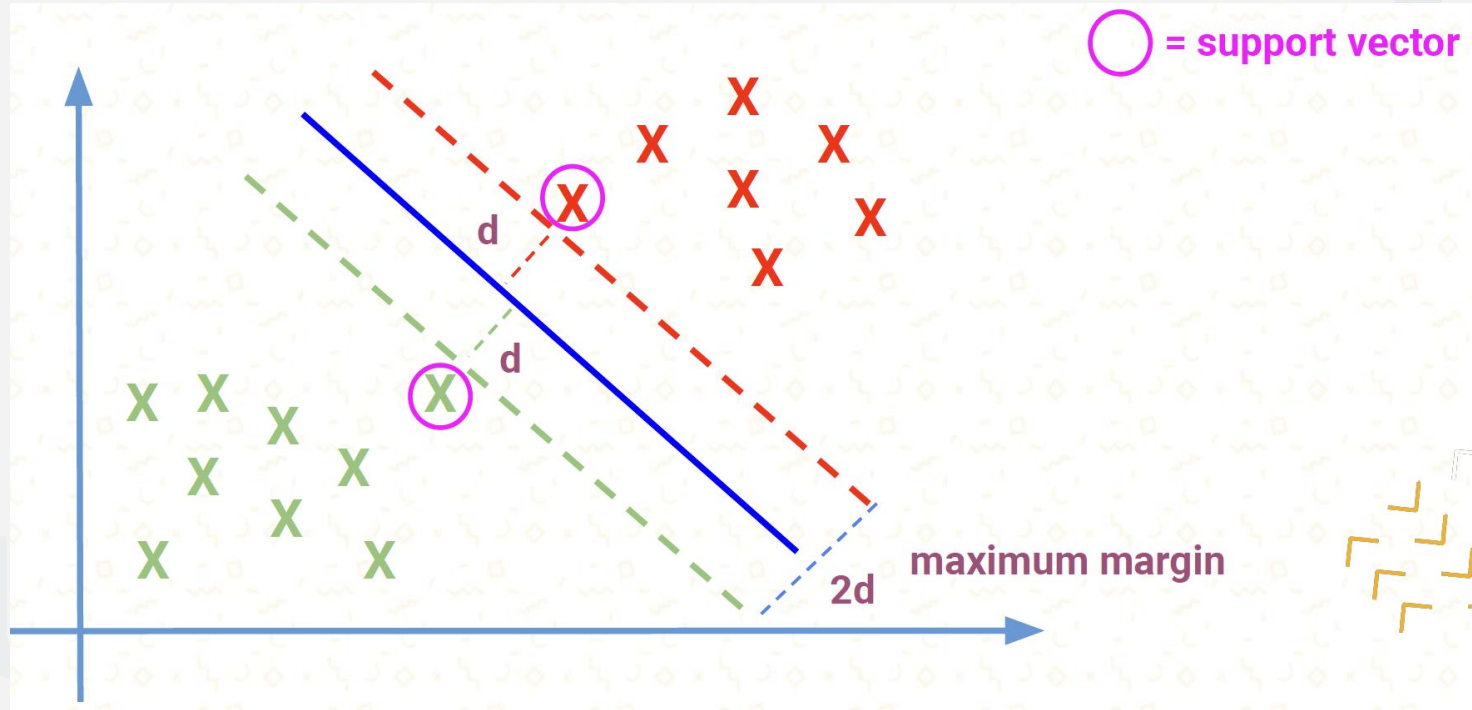
Permasalahan



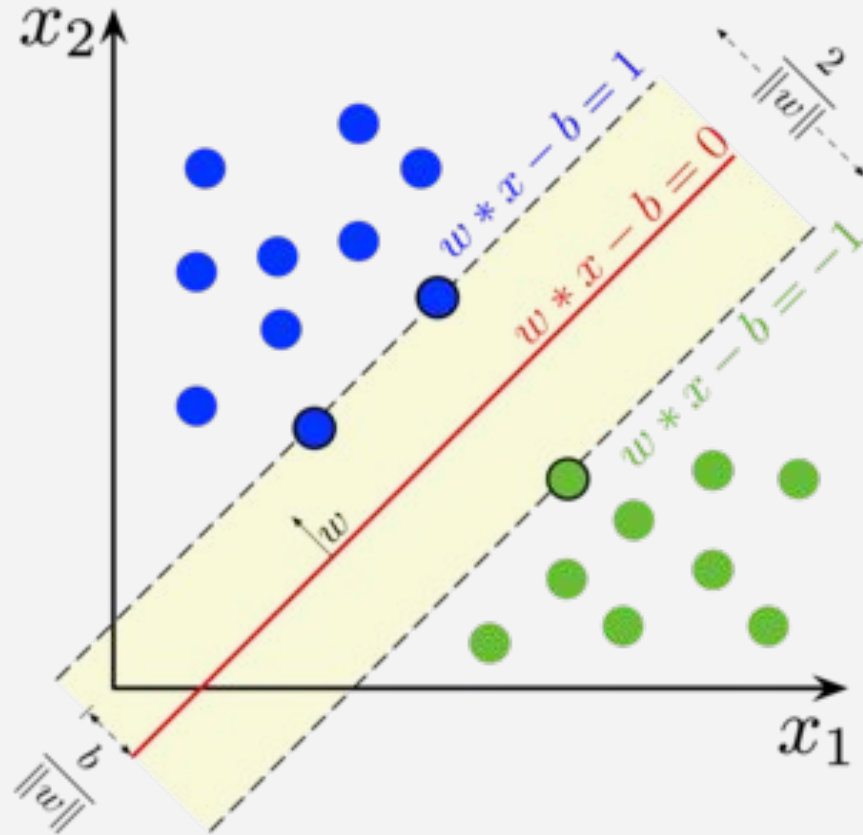
Garis Separasi mana yang terbaik?



Support Vector



Pendekatan Geometris



SVM Sebagai Convex Optimization Problem

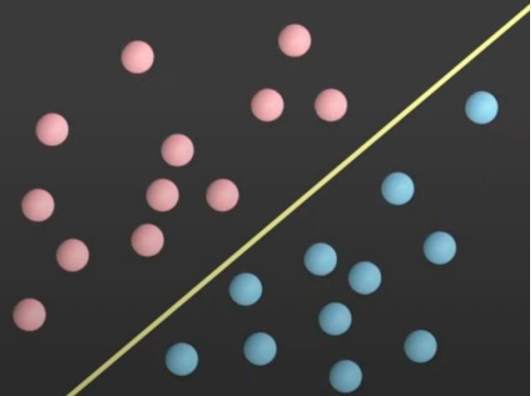


Convex Optimization Problem

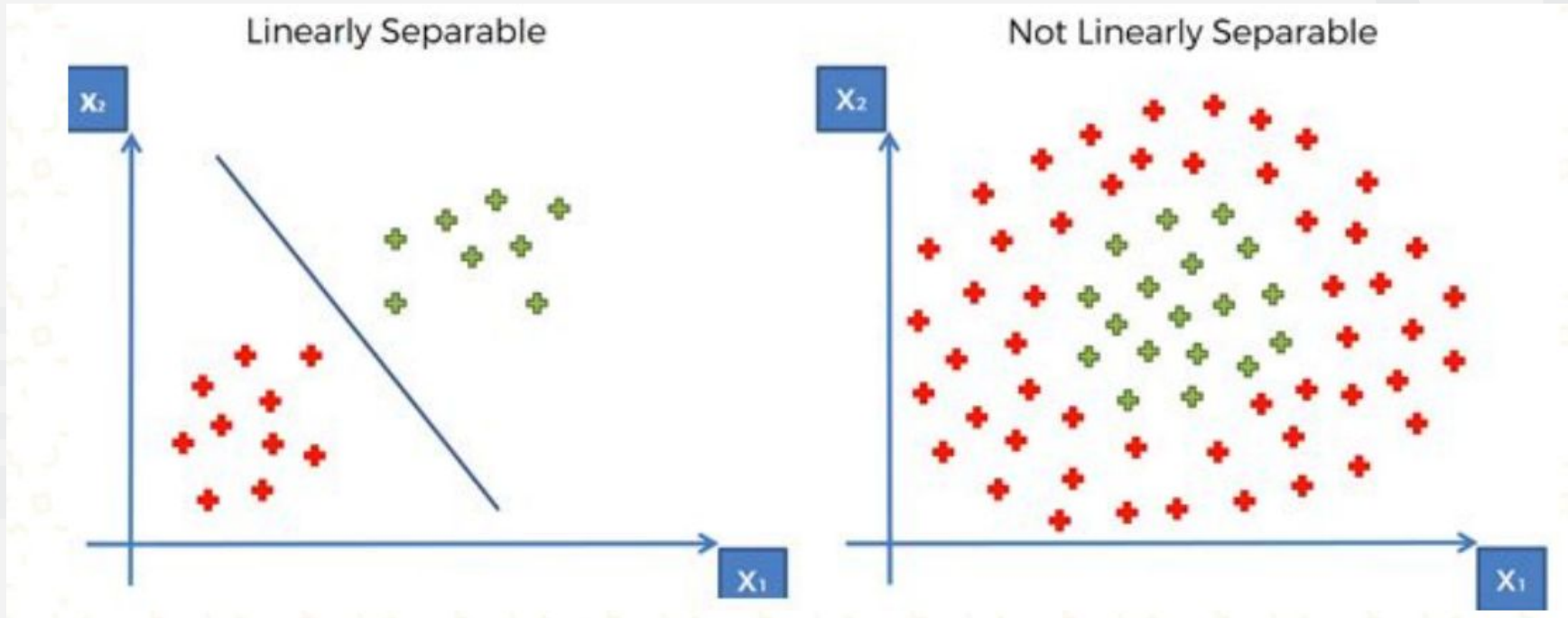
$$\max_{w, b} ||w||^{-2}$$

$$w^T x + b \geq 1 \quad \forall x \in C_1$$

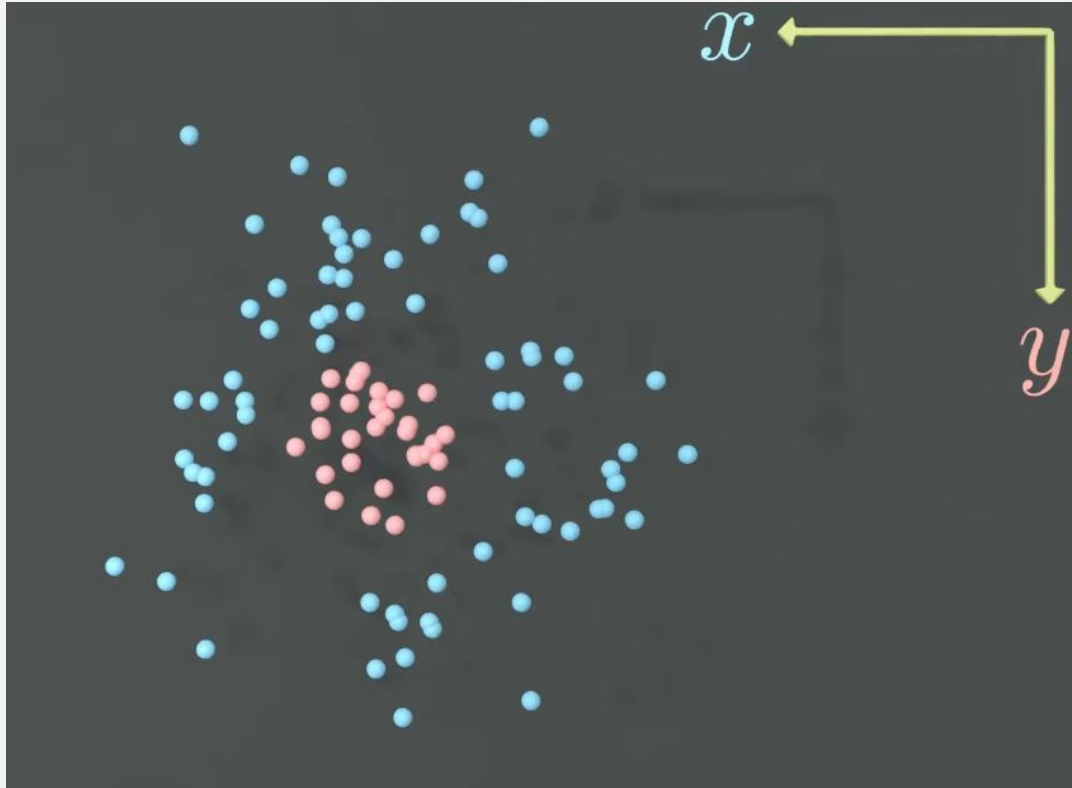
$$w^T x + b \leq -1 \quad \forall x \in C_2$$



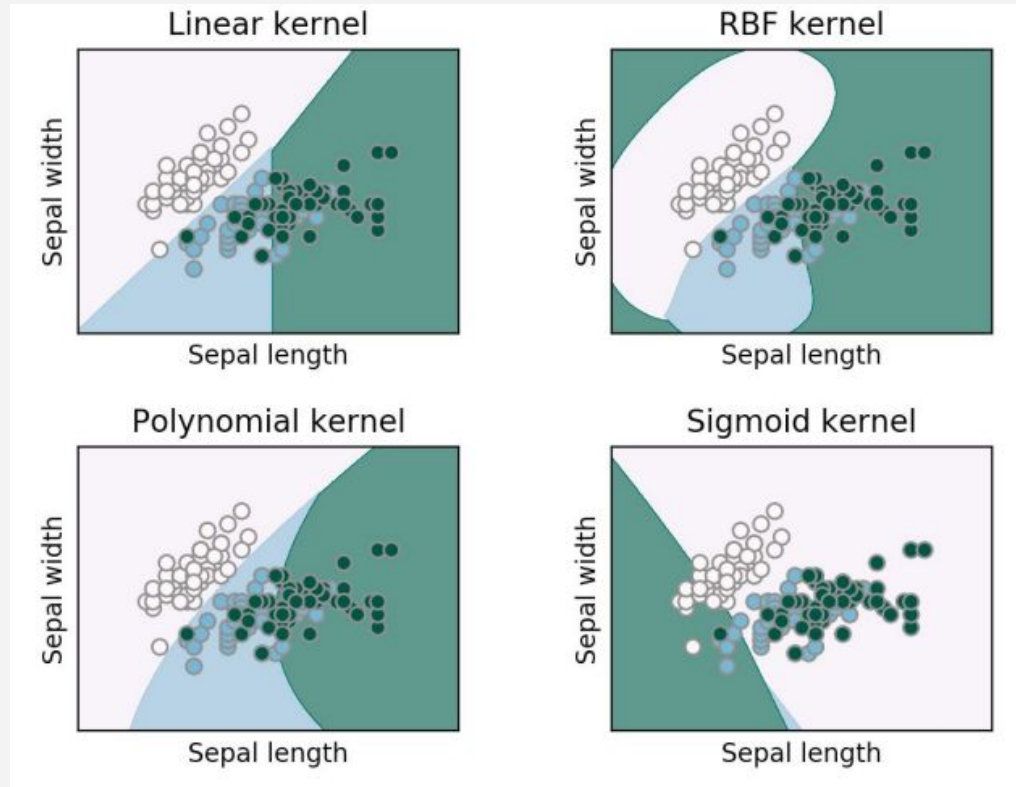
Kasus Lain



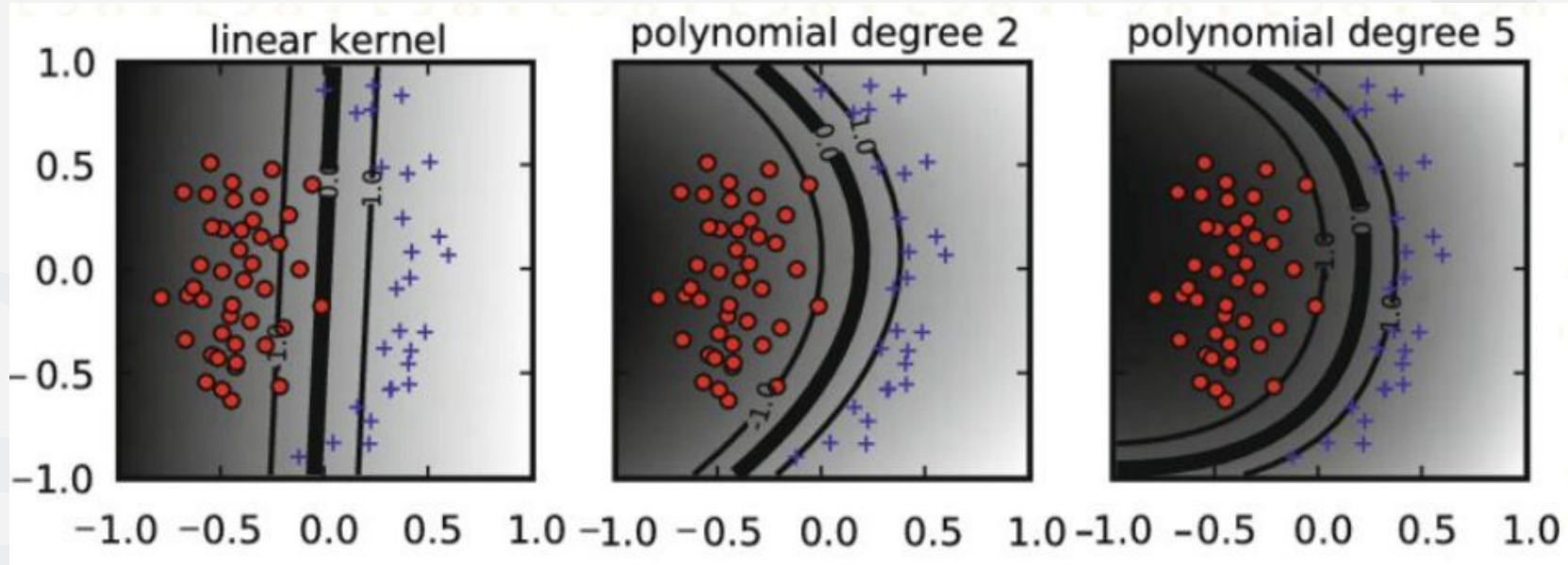
Kasus Lain



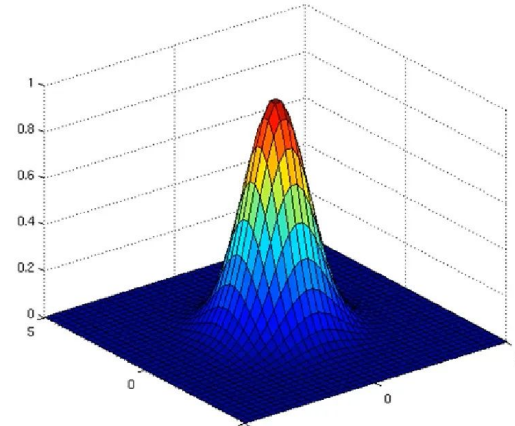
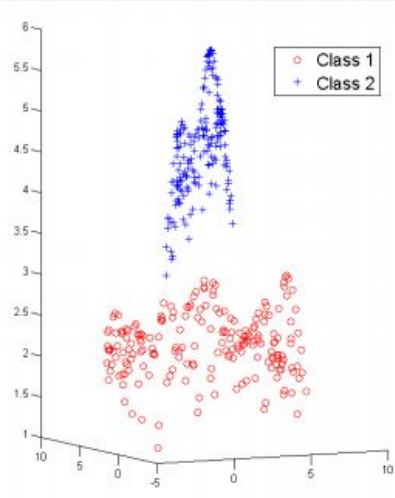
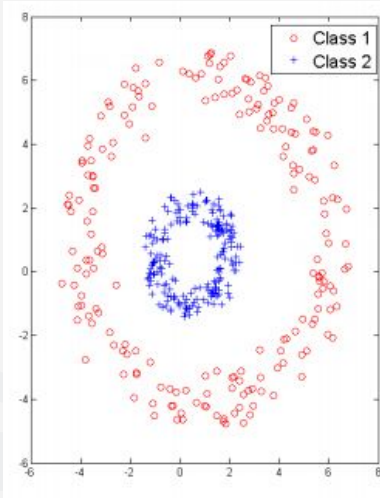
Fungsi-Fungsi Kernel Pada SVM



Kernel Polynomial



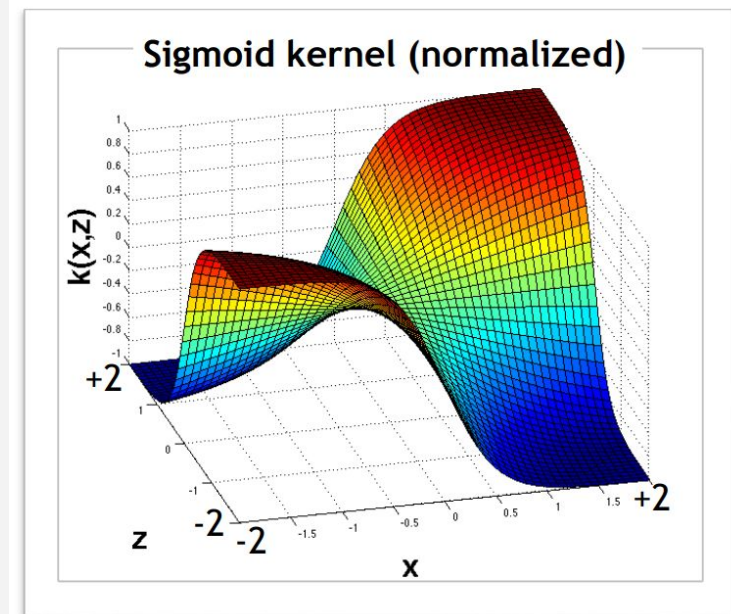
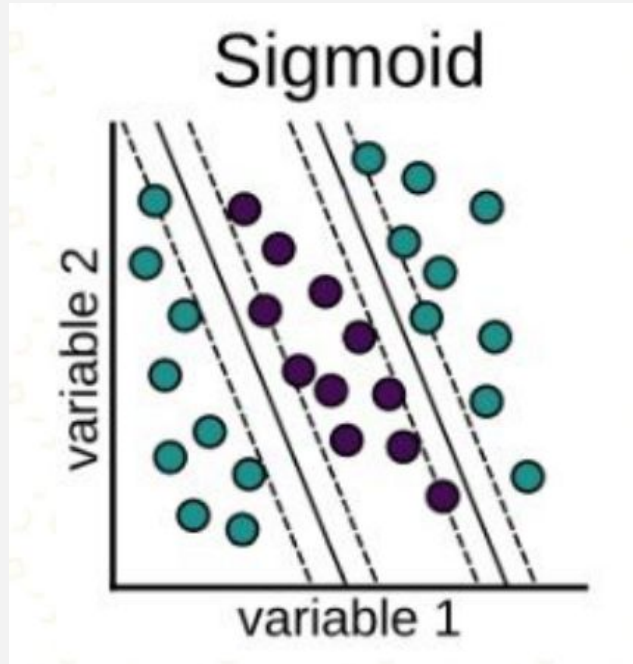
Kernel Radial Basis Function (RBF) (Gaussian Kernel)



$$K(\vec{x}, \vec{l}^i) = e^{-\frac{\|\vec{x} - \vec{l}^i\|^2}{2\sigma^2}}$$



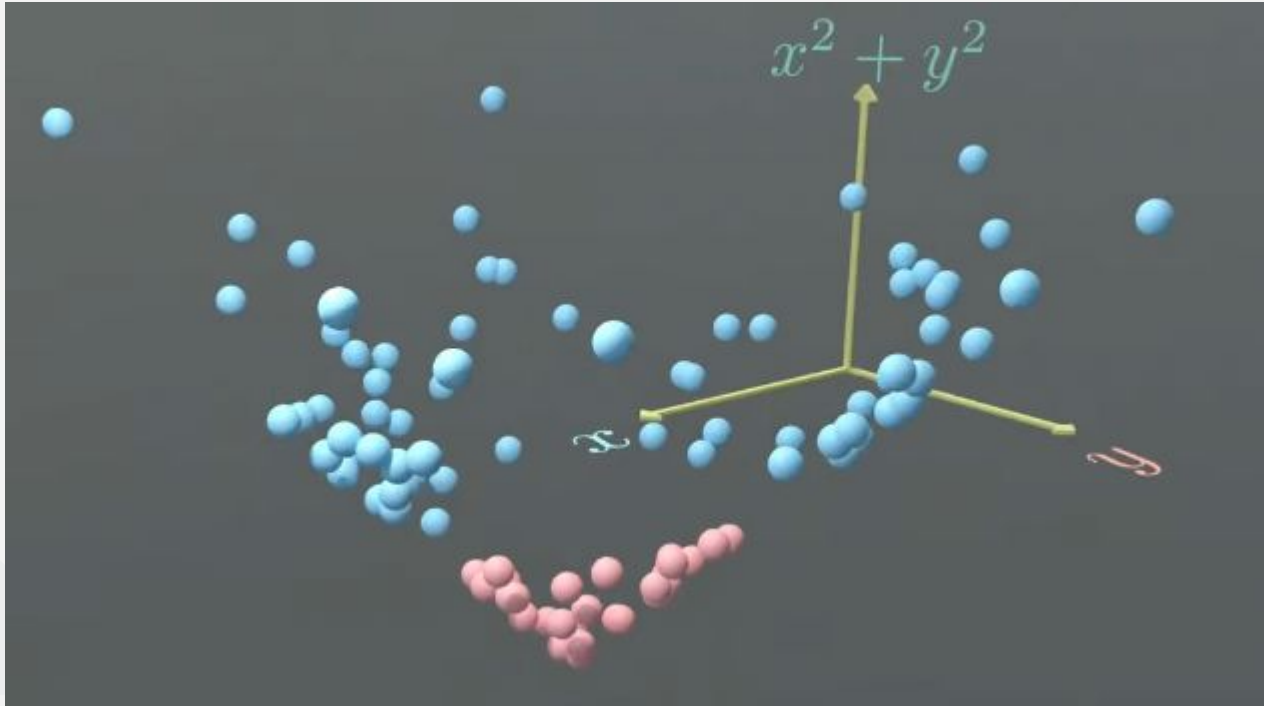
Kernel Sigmoid



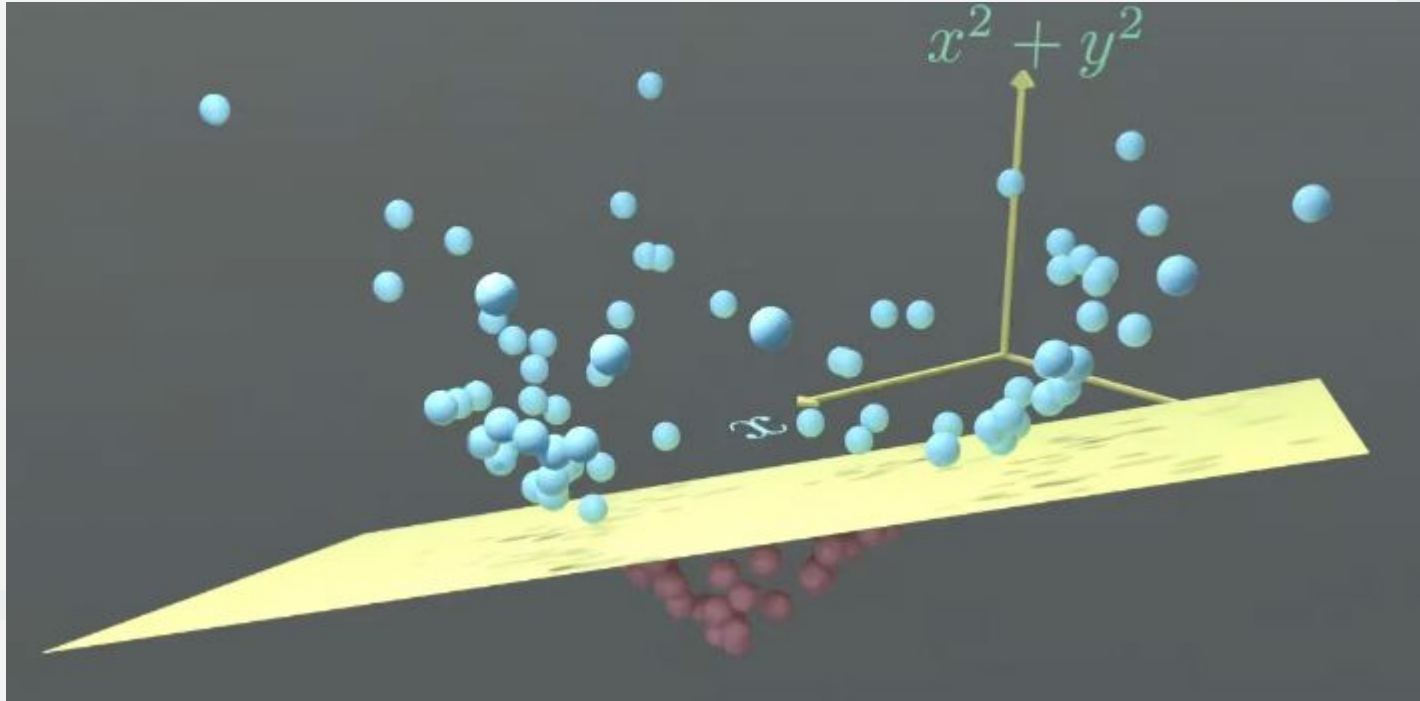
$$k_S \left(\frac{\mathbf{x}}{\sqrt{p}}, \frac{\mathbf{z}}{\sqrt{p}} \right) = \tanh \left(\frac{a}{p} \cdot \mathbf{x}^T \mathbf{z} + r \right)$$



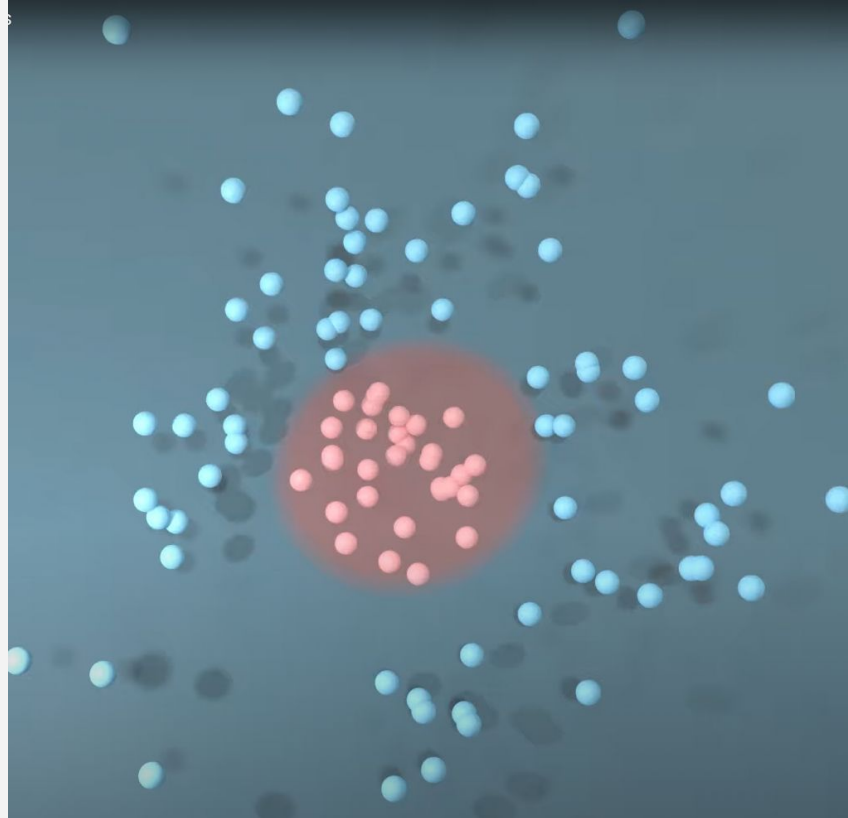
Fungsi Kernel: Data Augmentation



Fungsi Kernel: Untuk Data NOT Linearly Separable



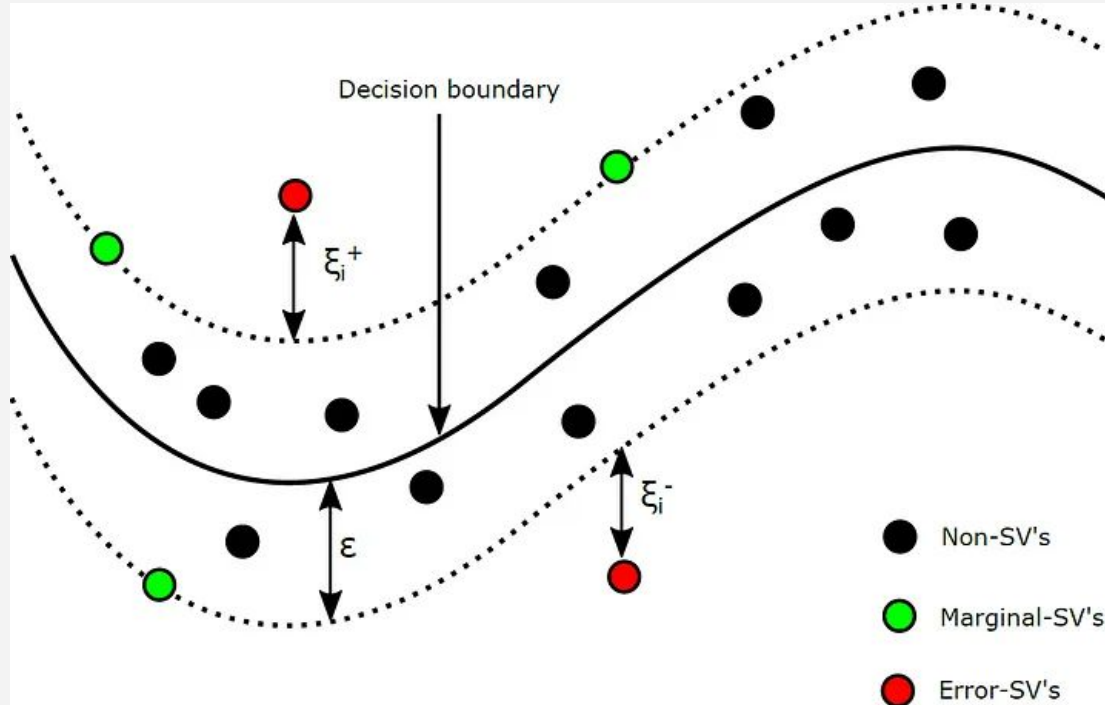
Fungsi Kernel: Untuk Data NOT Linearly Separable



KERNEL TRICK



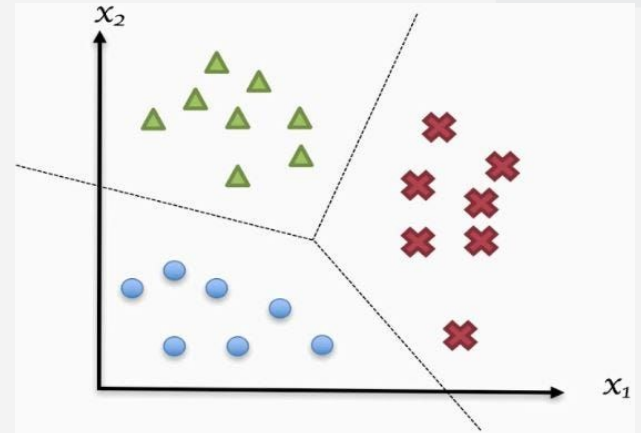
SVM Untuk Regresi: SVR



Source: [Link](#)

Multiclass Classifier

Apa dan Bagaimana



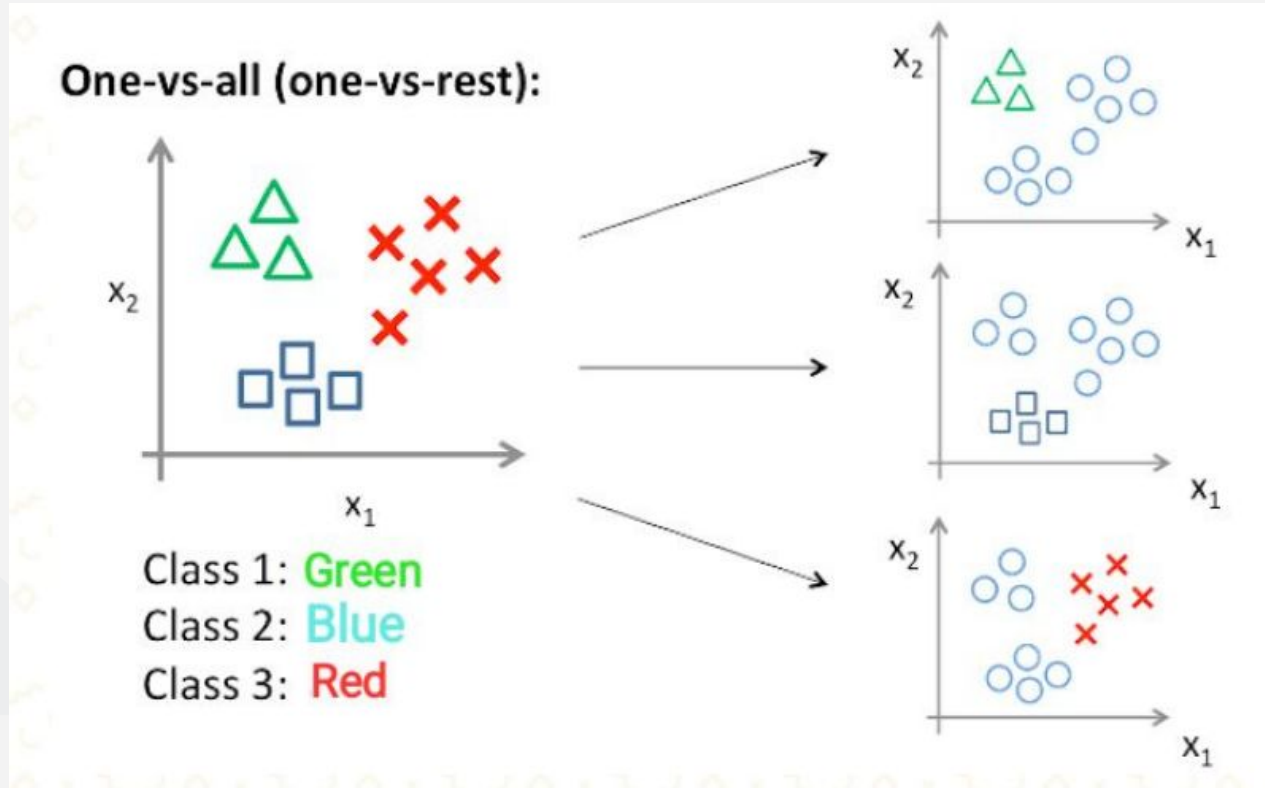
Multiclass Classifier

Konsep multiclass classifier bisa dilakukan pendekatan seperti binary classifier dengan metode:

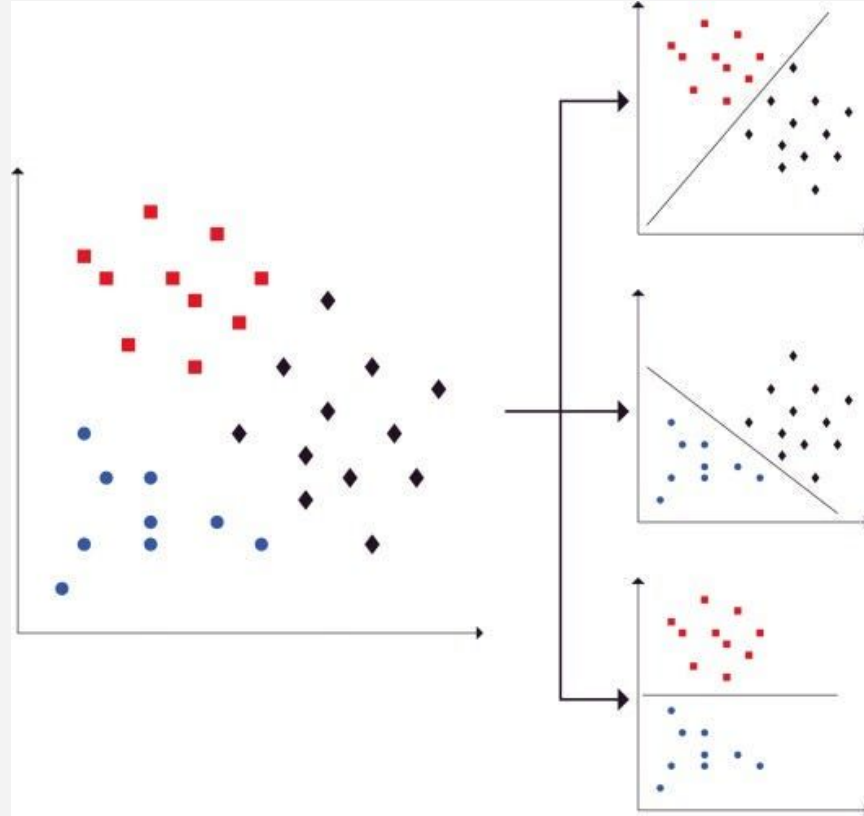
1. One vs Rest (OvR)
2. One vs One (OvO)



One vs. Rest (OvR)



One vs. One (OvO)



Memilih Antara OvR dan OvO



1. **Ukuran Dataset:** Untuk dataset berukuran kecil hingga menengah, kedua strategi dapat dipertimbangkan. Untuk dataset berukuran besar, OvR sering lebih praktis karena efisiensinya dalam komputasi.
2. **Jumlah Kelas:** OvO mungkin lebih baik ketika jumlah kelas sedikit, dan sumber daya komputasi bukanlah batasan. OvR umumnya lebih dapat diukur untuk jumlah kelas yang besar.
3. **Kelas yang Tidak Seimbang:** Jika kelas-kelas tidak seimbang, OvO mungkin lebih baik karena setiap klasifikasi biner dilatih pada subset yang seimbang. Namun, OvR dapat lebih efisien secara komputasi.
4. **Sumber Daya Komputasi:** Pertimbangkan sumber daya komputasi yang tersedia. OvO menjadi tidak praktis untuk jumlah kelas yang besar, sedangkan OvR lebih scalable.

Multiclass Confusion Matrix

		Predicted			
		A	B	C	
True labels	A	2	2	0	4
	B	1	2	0	3
	C	0	0	3	3
		3	4	3	Total



Evaluation Metric



$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$




$$\text{F1-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$




	precision	recall	f1-score	support
Cat	0.308	0.667	0.421	6
Fish	0.667	0.200	0.308	10
Hen	0.667	0.667	0.667	9
accuracy			0.480	25
macro avg	0.547	0.511	0.465	25
weighted avg	0.581	0.480	0.464	25




Evaluation Metric



Label	True Positive (TP)	False Positive (FP)	False Negative (FN)	Precision	Recall	F1 Score
 Airplane	2	1	1	0.67	0.67	$2 * (0.67 * 0.67) / (0.67 + 0.67)$ = 0.67
 Boat	1	3	0	0.25	1.00	$2 * (0.25 * 1.00) / (0.25 + 1.00)$ = 0.40
 Car	3	0	3	1.00	0.50	$2 * (1.00 * 0.50) / (1.00 + 0.50)$ = 0.67

Label	Per-Class F1 Score	Macro-Averaged F1 Score
 Airplane	0.67	$\frac{0.67 + 0.40 + 0.67}{3}$ = 0.58
 Boat	0.40	
 Car	0.67	

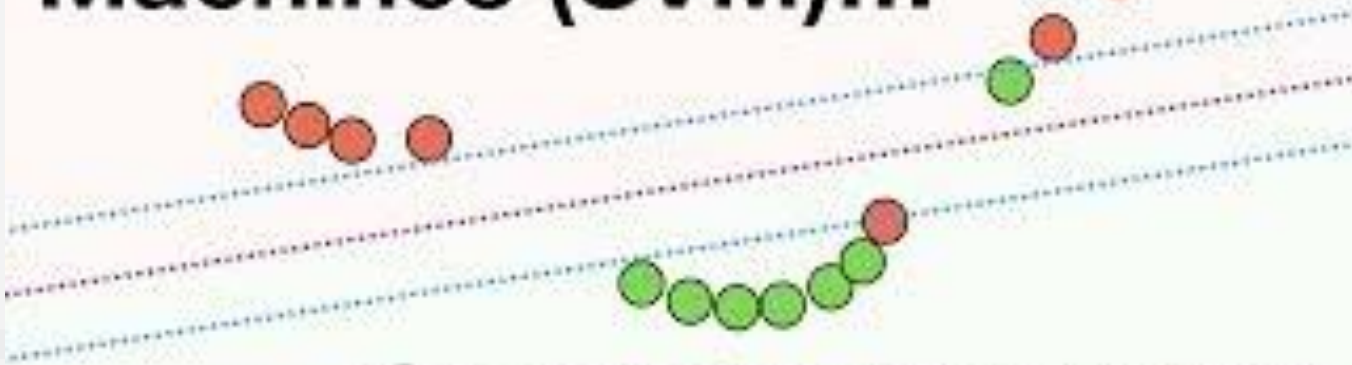
Label	Per-Class F1 Score	Support	Support Proportion	Weighted Average F1 Score
 Airplane	0.67	3	0.3	$(0.67 * 0.3) + (0.40 * 0.1) + (0.67 * 0.6)$ = 0.64
 Boat	0.40	1	0.1	
 Car	0.67	6	0.6	
Total	-	10	1.0	



Materi Tambahan



Support Vector Machines (SVM)...



...Clearly Explained!!!



Implementasi

- SVM for Classification & Regression
- Hands-on Coding in Google Colab

```
31 def __init__(self, settings):
32     self.file = None
33     self.fingerprints = set()
34     self.logdupes = True
35     self.debug = debug
36     self.logger = logging.getLogger(__name__)
37     if path:
38         self.file = open(os.path.join(path, "requests.log"),
39                         "a")
40         self.fingerprints.update(self.fingerprints)
41
42 @classmethod
43 def from_settings(cls, settings):
44     debug = settings.getbool("DEBUG", False)
45     return cls(job_dir(settings), debug)
46
47 def request_seen(self, request):
48     fp = self.request_fingerprint(request)
49     if fp in self.fingerprints:
50         return True
51     self.fingerprints.add(fp)
52     if self.file:
53         self.file.write(fp + os.linesep)
54
55 def request_fingerprint(self, request):
56     return request_fingerprint(request)
```





Thank You

