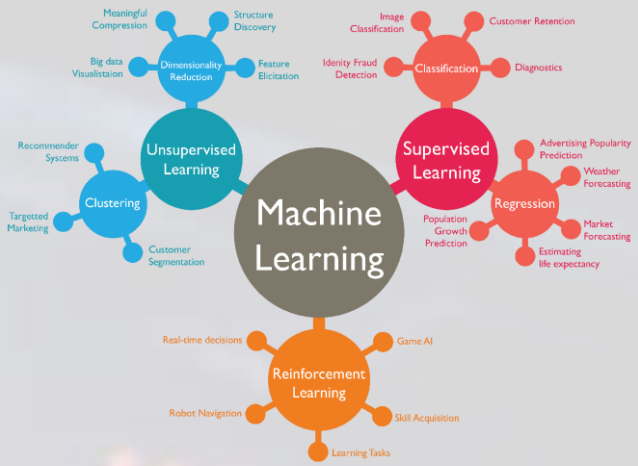


บทที่ 8

การจำแนกข้อมูลด้วยวิธี SVM (Data Classification with Support Vector Machine)

วิชา การวิเคราะห์ข้อมูลขั้นสูง (6608202) (Advanced Data Analytics)

Asst. Prof. Dr. Nattapong Songneam

xnattapong@hotmail.com

http://www.siam2dev.com

วิชาเหมืองข้อมูล Data Mining (4124305)

บทที่ 8

2/2567

การจำแนกข้อมูลด้วยวิธี SVM: Support Vector Machine (Data Classification with Support Vector Machine)

Asst. Prof. Dr. Nattapong Songneam

**Department of Computer Science,
Faculty of Science and Technology,
Phranakhon Rajabhat University**

แก้ไขล่าสุด : 11.02.2568

เทคนิคในการทำเหมืองข้อมูล

1. การวิเคราะห์ข้อมูลเชิงสถิติในงานเหมืองข้อมูล	บทที่ 4 การวิเคราะห์ข้อมูลเชิงสถิติในงานเหมืองข้อมูล
2. กฎความสัมพันธ์ (Association rule)	บทที่ 5 กฎความสัมพันธ์ (Association rule)
3. การจำแนกข้อมูล (Data classification)	บทที่ 7-9 Decision Tree, SVM , KNN , NN , Naive Bays
4. การแบ่งกลุ่มข้อมูล (Data clustering)	บทที่ 10 K-Mean, Hierarchical Clustering
5. การสร้างมโนภาพ (Visualization)	บทที่ 10 K-Mean, Hierarchical Clustering

8

Agenda

0 : ทบทวน

8.1 ความหมายของ SVM

8.2 คุณสมบัติหลักของ SVM

8.3 กระบวนการทำงานของ SVM

8.4 การจำแนกข้อมูล (Data Classification) ด้วยวิธี SVM

8.5 ขั้นตอนในการจำแนกข้อมูลด้วย SVM

8.6 ข้อดีของการใช้ SVM สำหรับการจำแนกข้อมูล

8.7 ตัวอย่างการประยุกต์การจำแนกข้อมูลด้วย SVM

8.8 สรุป

8

ประเภทของการเรียนรู้ของเครื่อง

Reviews

ประเภทของการเรียนรู้ของเครื่อง

1. การเรียนรู้แบบมีผู้สอน (Supervised Learning): แบบจำลองที่ฝึกฝนจากข้อมูลที่มีป้ายกำกับ (labeled data) เช่น การทำนายราคาบ้านจากข้อมูลราคาที่มีอยู่
2. การเรียนรู้แบบไม่มีผู้สอน (Unsupervised Learning) : แบบจำลองที่ฝึกฝนจากข้อมูลที่ไม่มีป้ายกำกับ เช่น การจัดกลุ่มลูกค้าตามพฤติกรรมการซื้อ
3. การเรียนรู้แบบกึ่งผู้สอน (Semi-Supervised Learning): ผสมผสานระหว่างข้อมูลที่มีป้ายกำกับและไม่มีป้ายกำกับ
4. การเรียนรู้แบบเสริมกำลัง (Reinforcement Learning) : แบบจำลองที่เรียนรู้ผ่านการทดลองและการตอบสนองจากสภาพแวดล้อม

8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

Reviews

การแบ่งข้อมูลออกเป็นชุดข้อมูลฝึกสอน (training set) และชุดข้อมูลทดสอบ (test set)

การแบ่งข้อมูลออกเป็นชุดข้อมูลฝึกสอน (training set) และชุดข้อมูลทดสอบ (test set) เป็นกระบวนการสำคัญในขั้นตอนการสร้างและประเมินโมเดลการวิเคราะห์ข้อมูลหรือการเรียนรู้ของเครื่อง (Machine Learning) โดยมีรายละเอียดดังนี้:

1. **ชุดข้อมูลฝึกสอน (Training Data Set)**
2. **ชุดข้อมูลทดสอบ (Test Data Set)**

การแบ่งข้อมูลเป็นขั้นตอนสำคัญที่ช่วยให้โมเดลการเรียนรู้ของเครื่องมีความน่าเชื่อถือและมีประสิทธิภาพในการนำไปใช้งานจริง

8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

Reviews

การแบ่งข้อมูลออกเป็นชุดข้อมูลฝึกสอน (training set) และชุดข้อมูลทดสอบ (test set)

1. ชุดข้อมูลฝึกสอน (Training Set)

ความหมาย: ชุดข้อมูลฝึกสอนเป็นชุดข้อมูลที่ใช้ในการฝึกสอนโมเดล โดยโมเดลจะทำการเรียนรู้จากข้อมูลชุดนี้ ซึ่งหมายถึงการทำความเข้าใจรูปแบบ (patterns) และความสัมพันธ์ระหว่างข้อมูลต่าง ๆ เช่น ข้อมูลอินพุต (input features) และผลลัพธ์ (output labels) ที่ถูกต้อง

การใช้งาน: ในกระบวนการฝึกสอน โมเดลจะปรับพารามิเตอร์ภายในเพื่อให้สามารถทำนายผลลัพธ์จากข้อมูลอินพุตได้อย่างถูกต้องมากที่สุด

8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

Reviews

การแบ่งข้อมูลออกเป็นชุดข้อมูลฝึกสอน (training set) และชุดข้อมูลทดสอบ (test set)

2. ชุดข้อมูลทดสอบ (Test Set)

ความหมาย: ชุดข้อมูลทดสอบเป็นชุดข้อมูลที่ใช้ในการประเมินประสิทธิภาพของโมเดลหลังจากที่ได้ทำการฝึกสอนด้วยข้อมูลฝึกสอนแล้ว ชุดข้อมูลทดสอบนี้จะไม่ถูกใช้ในขั้นตอนการฝึกสอน เพื่อให้สามารถตรวจสอบได้ว่าโมเดลสามารถทำงานได้ดีเพียงใดกับข้อมูลใหม่ที่ไม่เคยเห็นมาก่อน

การใช้งาน: ใช้เพื่อวัดความแม่นยำและประสิทธิภาพของโมเดลในการทำนายข้อมูลที่ไม่รู้จัก ทำให้สามารถประเมินได้ว่าโมเดลมีความสามารถในการทั่วไป (generalization) ดีเพียงใด

8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

Reviews

การแบ่งข้อมูลออกเป็นชุดข้อมูลฝึกสอน (training set) และชุดข้อมูลทดสอบ (test set)

การแบ่งสัดส่วน (Split Data)

การแบ่งข้อมูลออกเป็น training set และ test set มักจะใช้สัดส่วนต่าง ๆ ตามปริมาณข้อมูลและลักษณะของปัญหา เช่น:

70/30: 70% ของข้อมูลสำหรับการฝึกสอน และ 30% สำหรับการทดสอบ
80/20: 80% สำหรับการฝึกสอน และ 20% สำหรับการทดสอบ
90/10: 90% สำหรับการฝึกสอน และ 10% สำหรับการทดสอบ

8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

Reviews

วิธี K-Fold Cross-Validation

K-Fold Cross-Validation เป็นเทคนิคที่ใช้เพื่อประเมินประสิทธิภาพของโมเดลการเรียนรู้ของเครื่อง (machine learning) โดยไม่ต้องใช้ข้อมูลทั้งหมดในการฝึกสอนและทดสอบเพียงครั้งเดียว ช่วยให้การประเมินมีความแม่นยำและน่าเชื่อถือมากขึ้น โดยเฉพาะในกรณีที่ข้อมูลมีขนาดไม่มากพอหรือมีความไม่สมดุลในการกระจายของข้อมูล

ขั้นตอนการทำ K-Fold Cross-Validation

1. แบ่งข้อมูลออกเป็น K ส่วน (Folds)
    1. ข้อมูลทั้งหมดจะถูกแบ่งออกเป็น K ส่วน (folds) ที่มีขนาดใกล้เคียงกัน แต่ละ fold จะมีการกระจายของข้อมูลที่หลากหลายและครอบคลุม
    2. ตัวอย่างเช่น หากมี 100 ตัวอย่างข้อมูลและเลือก K = 5 ข้อมูลจะถูกแบ่งออกเป็น 5 fold แต่ละ fold จะมี 20 ตัวอย่าง
2. วนรอบการฝึกสอนและทดสอบ K ครั้ง
    1. ในแต่ละรอบ (iteration) หนึ่งใน K fold จะถูกใช้เป็นชุดทดสอบ (Testing Set) และอีก K-1 fold ที่เหลือจะถูกใช้เป็นชุดฝึกสอน (Training Set)
    2. โมเดลจะถูกฝึกด้วยข้อมูล K-1 fold และทดสอบด้วย fold ที่ถูกเลือกให้เป็นชุดทดสอบ
    3. ทำเช่นนี้ซ้ำไปเรื่อย ๆ จนครบทั้ง K fold โดยในแต่ละครั้ง fold ที่ใช้เป็นชุดทดสอบจะแตกต่างกัน
3. เฉลี่ยผลการทดสอบ
    1. หลังจากการทดสอบ K ครั้งเสร็จสิ้น จะนำผลการทดสอบจากแต่ละรอบมาคำนวณค่าเฉลี่ย เพื่อให้ได้ประเมินประสิทธิภาพของโมเดลโดยรวม
    2. ค่าเฉลี่ยนี้สามารถเป็นค่าความแม่นยำ (accuracy), ค่าเฉลี่ยความคลาดเคลื่อน (mean error), หรือค่าประสิทธิภาพอื่น ๆ ที่เหมาะสมกับงานที่ทำ

8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

Reviews

วิธี K-Fold Cross-Validation

ข้อดีของ K-Fold Cross-Validation

- ลด Bias: โดยการใช้ข้อมูลทั้งหมดในการฝึกสอนและทดสอบ ทำให้ผลการประเมินมีความเป็นกลางมากขึ้น
- ลด Variance: เนื่องจากใช้ข้อมูลหลายชุดในการฝึกสอนและทดสอบ ผลลัพธ์ที่ได้จึงมีความน่าเชื่อถือมากขึ้นและมีความเสถียร
- ใช้ข้อมูลอย่างมีประสิทธิภาพ: K-Fold ใช้ข้อมูลทุกตัวอย่างในการฝึกสอนและทดสอบ ทำให้ไม่สูญเสียข้อมูลในการประเมิน

ข้อเสียของ K-Fold Cross-Validation

- ใช้เวลามากขึ้น: เนื่องจากต้องทำการฝึกสอนและทดสอบ K ครั้ง เวลาในการคำนวณจะเพิ่มขึ้นอย่างมากโดยเฉพาะกับข้อมูลขนาดใหญ่
- การปรับพารามิเตอร์ที่ซับซ้อน: การเลือกค่า K ที่เหมาะสมอาจต้องทดลองหลายครั้ง และในบางกรณี K ที่แตกต่างกันอาจให้ผลลัพธ์ที่แตกต่างกัน

การเลือกค่า K ที่เหมาะสม

- โดยทั่วไป ค่า K ที่ใช้บ่อยคือ 5 หรือ 10 เนื่องจากให้ความสมดุลระหว่างความแม่นยำและเวลาที่ใช้ในการคำนวณ
- สำหรับข้อมูลขนาดใหญ่ที่ใช้เวลามาก อาจเลือก K ที่น้อยลง เช่น K = 3
- สำหรับข้อมูลขนาดเล็ก การใช้ค่า K ที่มากขึ้น (เช่น K = 10) อาจช่วยเพิ่มความแม่นยำในการประเมิน

K-Fold Cross-Validation เป็นวิธีที่มีประสิทธิภาพและเป็นที่นิยมในการประเมินโมเดล สามารถใช้กับปัญหาต่าง ๆ ทั้งการจำแนก (classification) และการทำนายค่า (regression)

8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

Reviews

การแบ่งข้อมูลออกเป็นชุดข้อมูลฝึกสอน (training set) และชุดข้อมูลทดสอบ (test set)

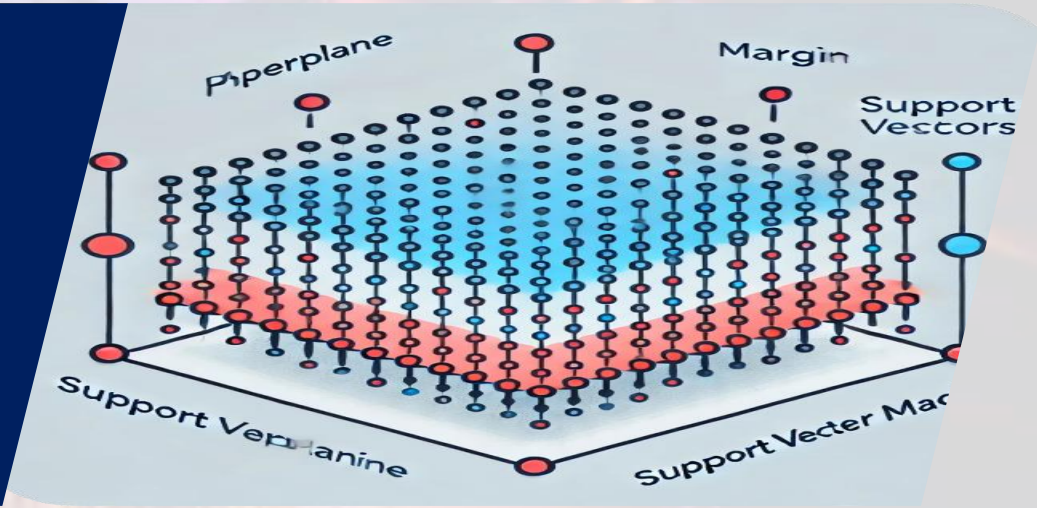
เหตุผลในการแบ่งข้อมูล

- ลดการเกิด Overfitting หากไม่มีการแบ่งข้อมูล โมเดลอาจเรียนรู้เฉพาะรูปแบบจากชุดข้อมูลฝึกสอนเพียงอย่างเดียว ซึ่งอาจทำให้เกิด Overfitting หรือการทำงานไม่ดีเมื่อเจอกับข้อมูลใหม่
- เพิ่มความเชื่อมั่นในผลลัพธ์ การมีชุดข้อมูลทดสอบช่วยให้สามารถตรวจสอบได้ว่าโมเดลมีความสามารถในการทั่วไปที่ดี ไม่ได้จำแค่ข้อมูลฝึกสอนเพียงอย่างเดียว

8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

การจำแนกข้อมูลด้วยวิธี Support Vector Machine



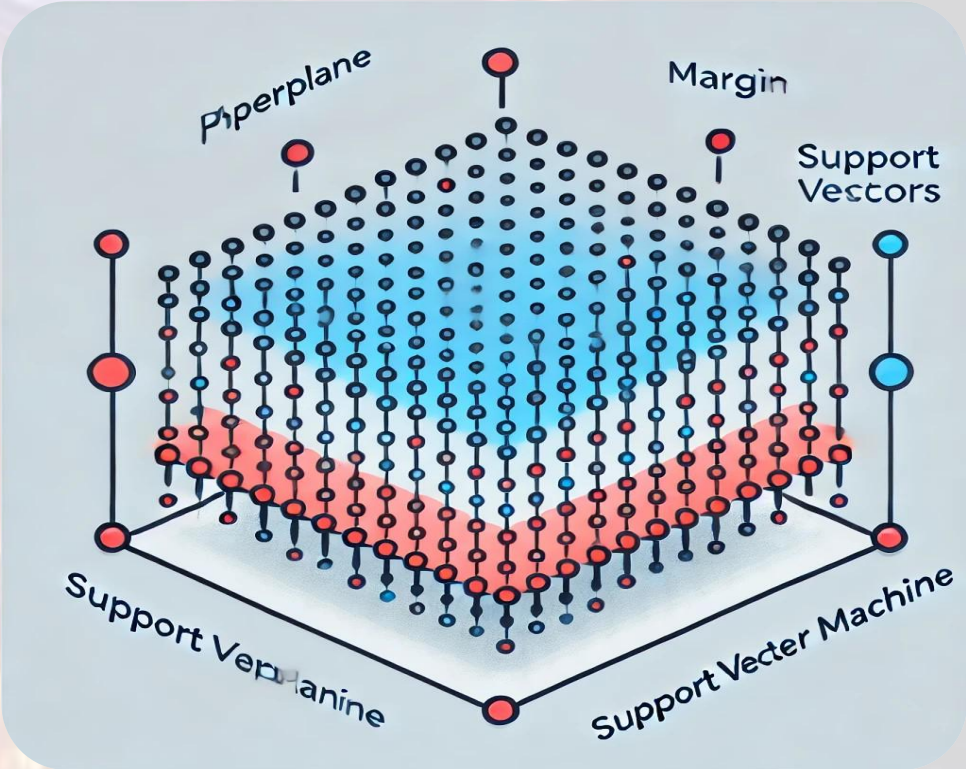
8

8.1 ความหมายของ Support Vector Machine (SVM)

8.1 Support Vector Machine (SVM) คืออะไร

ความหมายของ Support Vector Machine (SVM)

Support Vector Machine (SVM) เป็นอัลกอริทึมการเรียนรู้ของเครื่อง (Machine Learning) ที่ถูกใช้สำหรับงานจำแนกข้อมูล (classification) และงานถดถอย (regression) โดยเฉพาะอย่างยิ่งมีชื่อเสียงในการทำงานจำแนกข้อมูลแบบสองกลุ่ม (binary classification)



8

8.1 ความหมายของ Support Vector Machine (SVM)

8.1 Support Vector Machine (SVM) คืออะไร

8.1 ความหมายของ Support Vector Machine (SVM)

การจำแนกข้อมูล (classification) ด้วยวิธี Support Vector Machine (SVM) เป็นเทคนิคที่นิยมใช้ในงานด้าน Machine Learning สำหรับงานที่ต้องการแบ่งกลุ่มข้อมูลออกเป็นหลายกลุ่ม (classes) โดยหลักการพื้นฐานของ SVM คือการหาขอบเขต (decision boundary) ที่เหมาะสมที่สุดสำหรับแยกข้อมูลในแต่ละกลุ่มออกจากกัน ซึ่งขอบเขตนี้เรียกว่า hyperplane

Support Vector Machine (SVM) เป็นอัลกอริทึมการเรียนรู้ของเครื่อง (Machine Learning) ที่ถูกใช้สำหรับงานจำแนกข้อมูล (classification) และงานถดถอย (regression) โดยเฉพาะอย่างยิ่งมีชื่อเสียงในการทำงานจำแนกข้อมูลแบบสองกลุ่ม (binary classification)

8

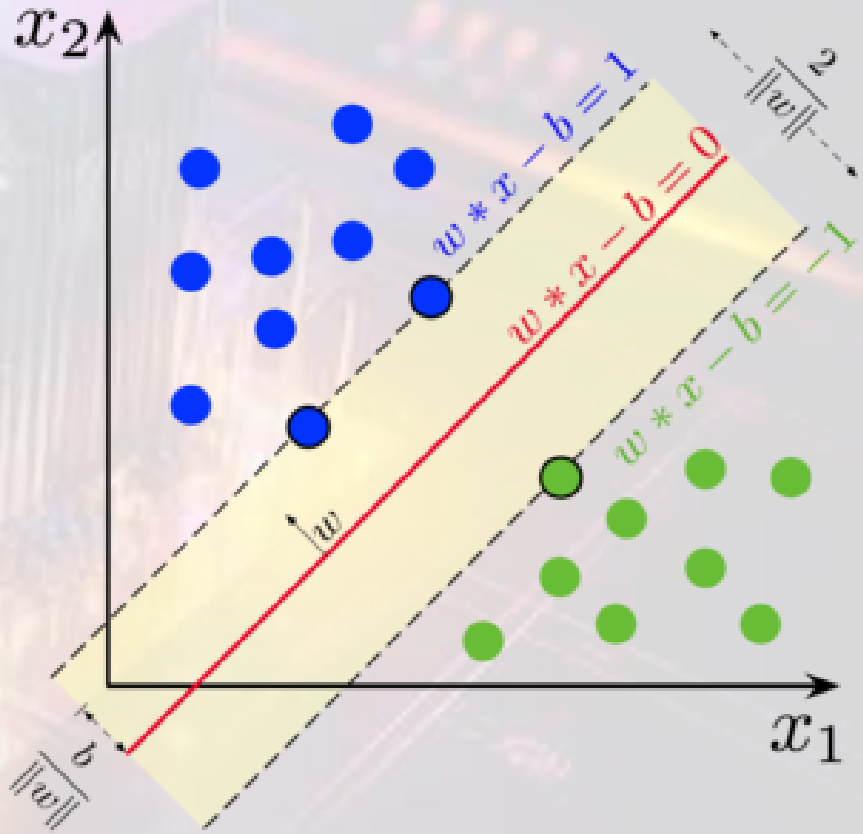
8.2 หลักการพื้นฐานของซัพพอร์ตเวกเตอร์แมกชีน

หลักการพื้นฐานของซัพพอร์ตเวกเตอร์แมกชีน

SVM ทำงานโดยการหาขอบเขต (decision boundary) ที่ดีที่สุดที่สามารถแบ่งข้อมูลออกเป็นกลุ่มต่าง ๆ ได้ ขอบเขตนี้เรียกว่า hyperplane ซึ่งในกรณีที่ข้อมูลอยู่ใน 2 มิติ hyperplane จะเป็นเส้นตรง แต่ถ้าอยู่ในหลายมิติ hyperplane จะเป็นพื้นที่หรือพื้นผิวที่แบ่งข้อมูลออกจากกัน

8.2.1 Hyperplane และ Support Vectors

- Hyperplane เป็นเส้นหรือพื้นผิวที่แบ่งข้อมูลออกเป็นสองกลุ่ม ซึ่ง SVM จะพยายามหาตำแหน่งของ hyperplane ที่ทำให้ระยะห่างระหว่างข้อมูลทั้งสองกลุ่ม (margin) มีค่ามากที่สุด
- Support Vectors ข้อมูลที่อยู่ใกล้กับ hyperplane มากที่สุด เรียกว่า support vectors ข้อมูลเหล่านี้เป็นข้อมูลสำคัญที่ใช้ในการหาตำแหน่งของ hyperplane

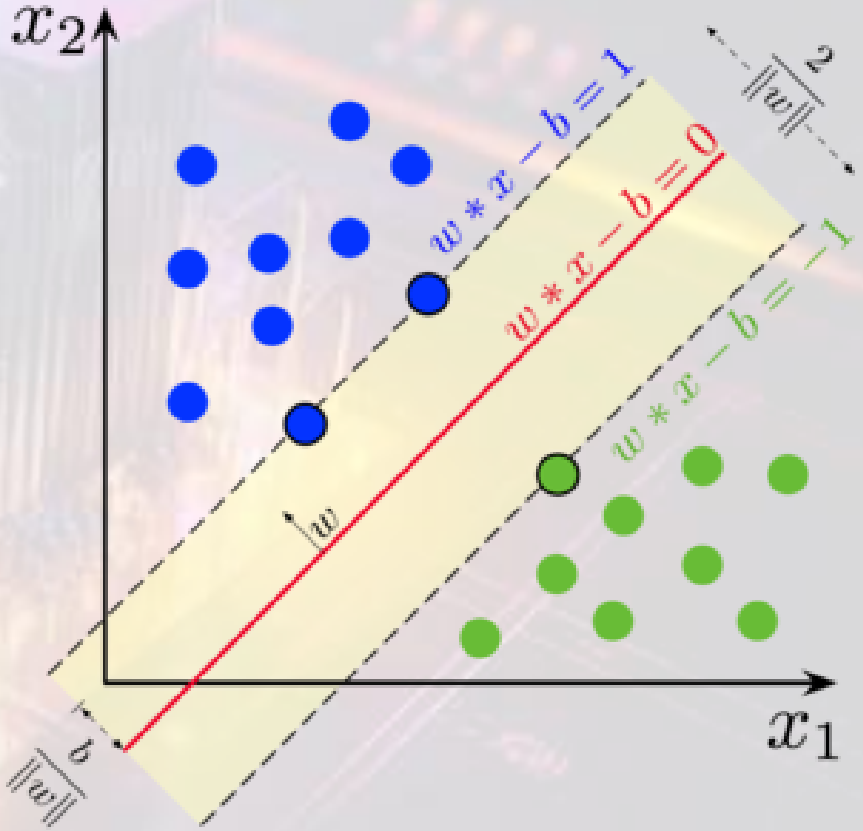


8

8.2 หลักการพื้นฐานของซัพพอร์ตเวกเตอร์แมกชีน

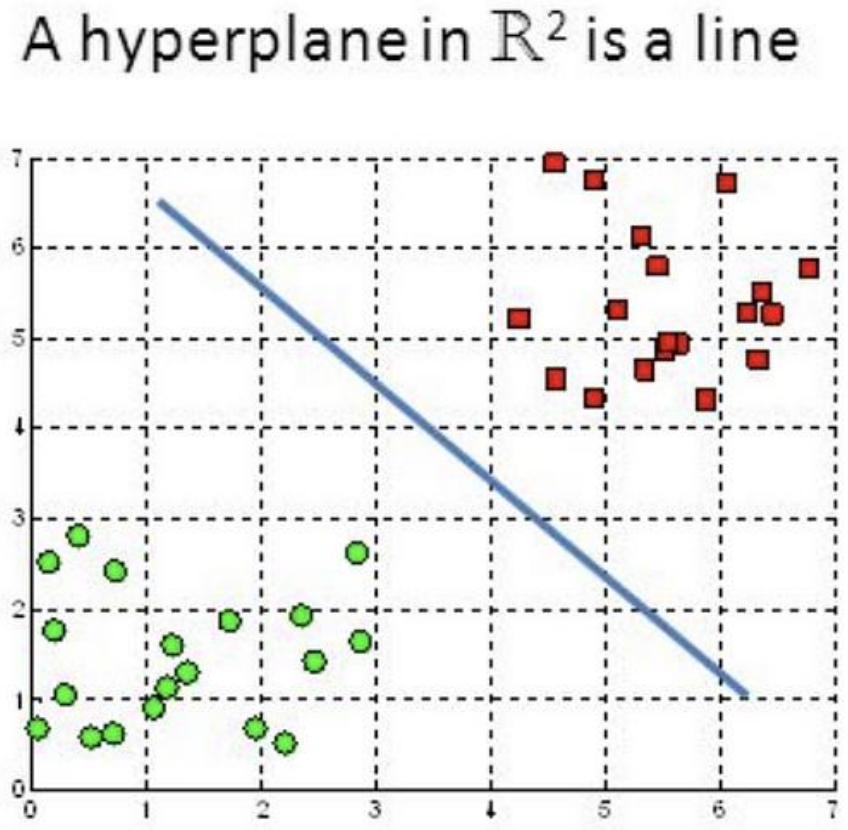
8.2.1 Hyperplane และ Support Vectors

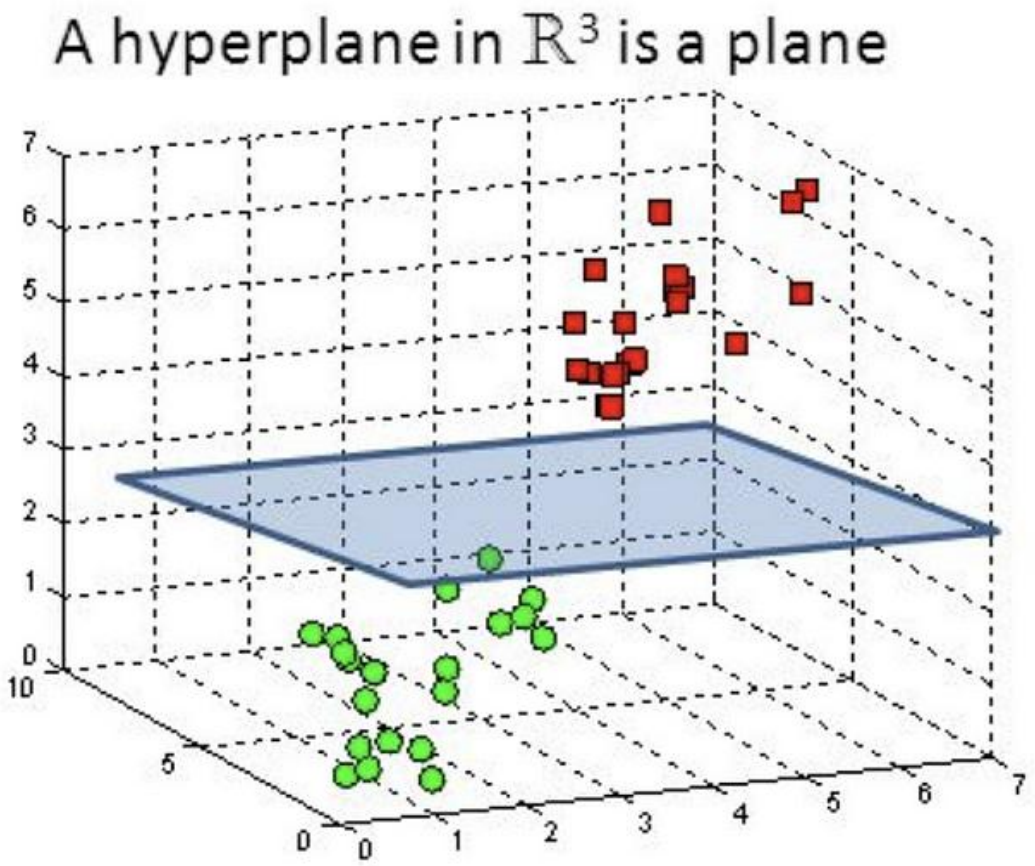
- Hyperplane เป็นเส้นหรือพื้นผิวที่แบ่งข้อมูลออกเป็นสองกลุ่ม ซึ่ง SVM จะพยายามหาตำแหน่งของ hyperplane ที่ทำให้ระยะห่างระหว่างข้อมูลทั้งสองกลุ่ม (margin) มีค่ามากที่สุด
- Support Vectors ข้อมูลที่อยู่ใกล้กับ hyperplane มากที่สุด เรียกว่า support vectors ข้อมูลเหล่านี้เป็นข้อมูลสำคัญที่ใช้ในการหาตำแหน่งของ hyperplane



8

8.2 หลักการพื้นฐานของซัพพอร์ตเวกเตอร์แมกชีน





8

8.2 หลักการพื้นฐานของซัพพอร์ตเวกเตอร์แมกชีน

8.2.2 Kernel Trick

ในบางกรณี ข้อมูลไม่สามารถแบ่งแยกได้ด้วยเส้นตรงในพื้นที่เดิม (input space) SVM สามารถใช้ Kernel Trick ในการเปลี่ยนข้อมูลไปยังพื้นที่ใหม่ (feature space) ที่สามารถแบ่งแยกได้ ตัวอย่างของ Kernel ที่นิยมใช้ได้แก่:

- Linear Kernel: ใช้เมื่อข้อมูลสามารถแบ่งได้ด้วยเส้นตรง
- Polynomial Kernel: ขยายพื้นที่ข้อมูลเป็นหลายมิติเพื่อให้สามารถแบ่งแยกได้
- Radial Basis Function (RBF) Kernel: ใช้สำหรับข้อมูลที่ไม่สามารถแบ่งได้ในพื้นที่เดิม

8.2.3 ค่าพารามิเตอร์สำคัญใน SVM

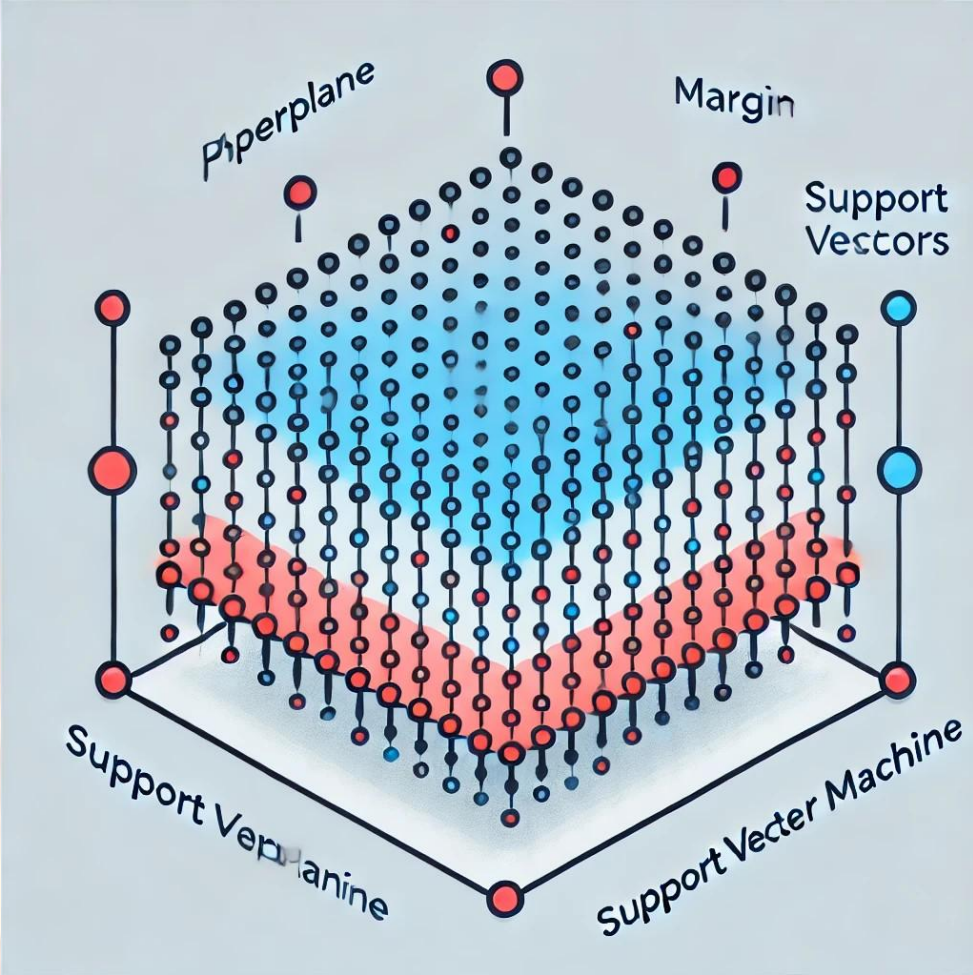
- C (Regularization Parameter): เป็นตัวควบคุมการยอมให้เกิดข้อผิดพลาดในข้อมูลฝึก (training data) ค่าที่ต่ำจะทำให้โมเดลยอมให้เกิดข้อผิดพลาดบ้างแต่จะสามารถ generalized กับข้อมูลใหม่ได้ดีขึ้น ค่าที่สูงจะพยายามทำให้ไม่มีข้อผิดพลาดในข้อมูลฝึกเลยแต่เสี่ยงต่อการเกิด overfitting

- γ (Gamma): เป็นพารามิเตอร์ของ Kernel เช่น RBF ซึ่งจะกำหนดว่าข้อมูลแต่ละจุดจะมีผลมากน้อยเพียงใดในการหาขอบเขต

8

8.2 หลักการพื้นฐานของซัพพอร์ตเวกเตอร์แมกชีน

ภาพที่แสดงถึงการทำงานของ Support Vector Machine (SVM) โดยมี hyperplane ที่แบ่งข้อมูลออกเป็นสองกลุ่ม และเส้นประที่แสดงถึง margin โดยมี support vectors เป็นจุดที่สัมผัสกับเส้นประทั้งสองด้าน



8

8.3 ขั้นตอนการจำแนกข้อมูลด้วย SVM

1. การเตรียมข้อมูล (Data Preparation)
2. การแบ่งข้อมูล (Data Splitting)
3. การเลือกและแปลงฟีเจอร์ (Feature Scaling)
4. การเลือกและฝึกโมเดล (Model Selection and Training)
5. การทดสอบโมเดล (Model Testing)
6. การปรับปรุงโมเดล (Model Optimization)
7. การใช้งานโมเดล (Model Deployment)

8

8.3 ขั้นตอนการจำแนกข้อมูลด้วย SVM

8.3.1 การเตรียมข้อมูล (Data Preparation)

1. การเตรียมข้อมูล (Data Preparation)

- **รวบรวมข้อมูล:** เริ่มต้นด้วยการรวบรวมชุดข้อมูลที่ต้องการจำแนก เช่น ข้อมูลที่มีฟีเจอร์ (features) และป้ายกำกับ (labels) ที่ระบุประเภทของข้อมูล
- **ทำความสะอาดข้อมูล:** กำจัดหรือปรับข้อมูลที่ขาดหาย ข้อมูลผิดพลาด หรือข้อมูลที่ไม่สอดคล้องกัน
- **การแปลงฟีเจอร์ (Feature Engineering):** หากฟีเจอร์เป็นข้อความหรือข้อมูลเชิงหมวดหมู่ จะต้องแปลงให้อยู่ในรูปแบบที่สามารถป้อนเข้าสู่ SVM ได้ เช่น การใช้เทคนิค One-Hot Encoding หรือ Bag of Words สำหรับข้อความ

```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.metrics import classification_report,
accuracy_score
# โหลดข้อมูลตัวอย่าง
data = datasets.load_breast_cancer()
X = data.data
y = data.target
```

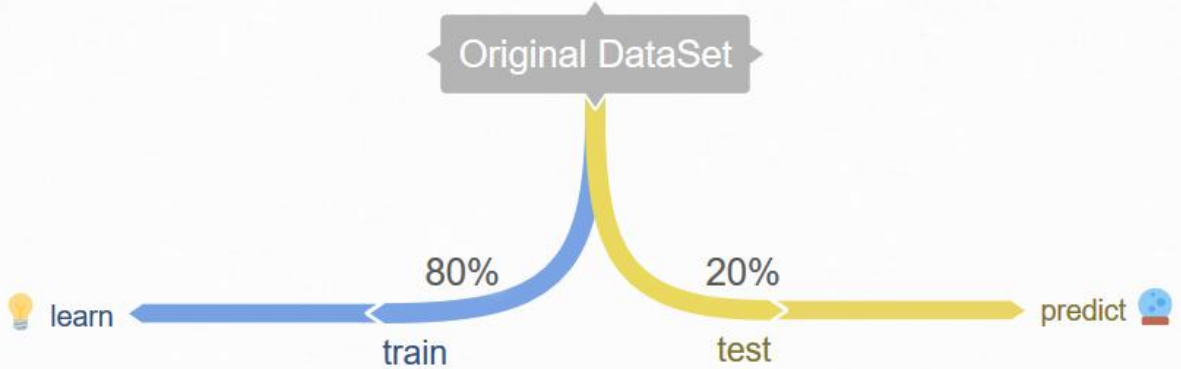
8

8.3 ขั้นตอนการจำแนกข้อมูลด้วย SVM

8.3.2 การแบ่งข้อมูล (Data Splitting)

2. การแบ่งข้อมูล (Data Splitting)

- **แบ่งข้อมูลออกเป็นชุดฝึก (Training Set) และชุดทดสอบ (Test Set):** โดยปกติจะแบ่งข้อมูลออกเป็น 70-80% สำหรับฝึกโมเดล และ 20-30% สำหรับทดสอบโมเดล เพื่อให้แน่ใจว่าโมเดลไม่ได้ถูกฝึกด้วยข้อมูลที่ใช้ในการประเมินผล



8

8.3 ขั้นตอนการจำแนกข้อมูลด้วย SVM

8.3.2 การแบ่งข้อมูล (Data Splitting)

2. การแบ่งข้อมูล (Data Splitting)

- **แบ่งข้อมูลออกเป็นชุดฝึก (Training Set) และชุดทดสอบ (Test Set):** โดยปกติจะแบ่งข้อมูลออกเป็น 70-80% สำหรับฝึกโมเดล และ 20-30% สำหรับทดสอบโมเดล เพื่อให้แน่ใจว่าโมเดลไม่ได้ถูกฝึกด้วยข้อมูลที่ใช้ในการประเมินผล

```
# โหลดข้อมูลตัวอย่าง
data = datasets.load_breast_cancer() X = data.data y = data.target
# แบ่งข้อมูลเป็นชุดฝึกและชุดทดสอบ
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

8

8.3 ขั้นตอนการจำแนกข้อมูลด้วย SVM

8.3.3 การเลือกและแปลงฟีเจอร์ (Feature Scaling)

3. การเลือกและแปลงฟีเจอร์ (Feature Scaling)

- แปลงฟีเจอร์ (Feature Scaling): ฟีเจอร์ต่าง ๆ ควรอยู่ในช่วงค่าที่ใกล้เคียงกัน เพื่อให้ SVM ทำงานได้มีประสิทธิภาพมากขึ้น โดยปกติจะใช้การ Standardization หรือ Normalization เพื่อแปลงฟีเจอร์ให้อยู่ในช่วงค่าที่เหมาะสม

```
# ทำ Feature Scaling
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

8

8.3 ขั้นตอนการจำแนกข้อมูลด้วย SVM

8.3.4 การเลือกและฝึกโมเดล (Model Selection and Training)

4. การเลือกและฝึกโมเดล (Model Selection and Training)

- เลือก Kernel: SVM สามารถใช้ kernel ที่แตกต่างกันในการจำแนกข้อมูล เช่น linear, polynomial, RBF (Radial Basis Function) และอื่น ๆ
- ฝึกโมเดล (Training the Model): ใช้ชุดข้อมูลฝึกในการฝึก SVM โมเดล โดยการหาขอบเขต (hyperplane) ที่สามารถแยกประเภทของข้อมูลออกจากกันได้ดีที่สุด

```
# เลือกและฝึก SVM โมเดล
model = SVC(kernel='linear')
model.fit(X_train, y_train)
```

8

8.3 ขั้นตอนการจำแนกข้อมูลด้วย SVM

8.3.5 การทดสอบโมเดล (Model Testing)

5. การทดสอบโมเดล (Model Testing)

- ทำนายผล (Prediction): ใช้โมเดลที่ถูกฝึกมาแล้วในการทำนายผลกับข้อมูลในชุดทดสอบ
- ประเมินผลลัพธ์ (Evaluation): วัดความแม่นยำของโมเดลด้วยการใช้เมตริกต่าง ๆ เช่น Accuracy, Precision, Recall, F1-score เพื่อประเมินว่าการจำแนกข้อมูลมีประสิทธิภาพเพียงใด

```
# ทำนายผลและประเมินโมเดล
y_pred = model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

8

8.3 ขั้นตอนการจำแนกข้อมูลด้วย SVM

8.3.6 การปรับปรุงโมเดล (Model Optimization)

6. การปรับปรุงโมเดล (Model Optimization)

- การปรับค่าพารามิเตอร์ (Hyperparameter Tuning): ปรับค่าพารามิเตอร์ของ SVM เช่น C, gamma, และ kernel เพื่อหาค่าที่ทำให้โมเดลมีประสิทธิภาพสูงสุด
- Cross-Validation: ใช้เทคนิค Cross-Validation เช่น k-fold cross-validation เพื่อตรวจสอบความสม่ำเสมอของโมเดลและป้องกันการ overfitting

8

8.3 ขั้นตอนการจำแนกข้อมูลด้วย SVM

8.3.7 การใช้งานโมเดล (Model Deployment)

7. การใช้งานโมเดล (Model Deployment)

- Deploy: เมื่อโมเดลมีประสิทธิภาพเพียงพอแล้ว สามารถนำไปใช้งานจริงในการจำแนกข้อมูลใหม่ ๆ ได้

8

8.4 ตัวอย่างการจำแนกข้อมูล Breast Cancer Dataset

8.4.1 ตัวอย่างชุดข้อมูล Breast Cancer Dataset จาก scikit-learn

ตัวอย่างชุดข้อมูล Breast Cancer Dataset จาก scikit-learn

Breast Cancer Dataset จาก scikit-learn เป็นชุดข้อมูลที่มักใช้สำหรับการเรียนรู้เชิงลึกและการจำแนกประเภท ชุดข้อมูลนี้ประกอบด้วยข้อมูลของผู้ป่วยมะเร็งเต้านม โดยมีคุณสมบัติที่ใช้สำหรับการจำแนกว่าผู้ป่วยมีมะเร็งเต้านมหรือไม่

ชุดข้อมูล Breast Cancer Dataset จาก scikit-learn เป็นปัญหาการจำแนกประเภท (classification) ที่ใช้จำแนกเซลล์มะเร็งว่าเป็น

Malignant (มะเร็งร้ายแรง) → 0

Benign (มะเร็งไม่ร้ายแรง) → 1

- โครงสร้างของ Breast Cancer Dataset
- จำนวนข้อมูลตัวอย่าง (Samples): 569 ตัวอย่าง
- จำนวนฟีเจอร์ (Features): 30 ฟีเจอร์เชิงตัวเลข (numeric features)
- ป้ายกำกับ (Target): 0 = มะเร็งชนิดไม่ร้ายแรง (benign), 1 = มะเร็งชนิดร้ายแรง (malignant)
- ฟีเจอร์ในชุดข้อมูล
- ฟีเจอร์ 30 ตัวในชุดข้อมูลนี้เกี่ยวข้องกับคุณสมบัติต่าง ๆ ของเซลล์มะเร็งเต้านม เช่น ขนาด, รูปร่าง, ความเรียบเนียน, ความกระด้าง, และอื่น ๆ

8

8.4 ตัวอย่างการจำแนกข้อมูล Breast Cancer Dataset

8.4.1 ตัวอย่างข้อมูล Breast Cancer Dataset จาก scikit-learn

ตัวอย่างข้อมูล Breast Cancer Dataset จาก scikit-learn

Breast Cancer Dataset จาก scikit-learn เป็นชุดข้อมูลที่มักใช้สำหรับการเรียนรู้เชิงลึกและการจำแนกประเภท ชุดข้อมูลนี้ประกอบด้วยข้อมูลของผู้ป่วยมะเร็งเต้านม โดยมีคุณสมบัติที่ใช้สำหรับการจำแนกว่าผู้ป่วยมีมะเร็งเต้านมหรือไม่

mean radius	mean texture	mean perimeter	mean area	mean smoothness	worst radius	worst texture	worst perimeter	worst area	worst smoothness	target
17.99	10.38	122.80	1001.0	0.11840	25.38	17.33	184.60	2019.0	0.16220	0
20.57	17.77	132.90	1326.0	0.08474	24.99	23.41	158.80	1956.0	0.12380	0
19.69	21.25	130.00	1203.0	0.10960	23.57	25.53	152.50	1709.0	0.14440	0
11.42	20.38	77.58	386.1	0.14250	14.91	26.50	98.87	567.7	0.20980	0
20.29	14.34	135.10	1297.0	0.10030	22.54	16.67	152.20	1575.0	0.13740	0

8

8.4 ตัวอย่างการจำแนกข้อมูล Breast Cancer Dataset

8.4.1 โปรแกรม Breast Cancer Dataset จาก scikit-learn

Part01 Lec08_Classification_ Breast_Cancer_SVM.py

```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.metrics import classification_report, accuracy_score

# โหลดข้อมูลตัวอย่าง
data = datasets.load_breast_cancer()
X = data.data
y = data.target

# แบ่งข้อมูลเป็นชุดฝึกและชุดทดสอบ
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SE
PS D:\DataAnalytics_Project> & C:/Users/User/AppData/Loca
py"
Accuracy: 0.9766081871345029
Classification Report:
              precision    recall  f1-score   support

           0       0.97      0.97      0.97        63
           1       0.98      0.98      0.98       108

    accuracy                           0.98       171
   macro avg       0.97      0.97      0.97       171
weighted avg       0.98      0.98      0.98       171

PS D:\DataAnalytics_Project>
```

8

8.4 ตัวอย่างการจำแนกข้อมูล Breast Cancer Dataset

8.4.1 โปรแกรม Breast Cancer Dataset จาก scikit-learn

Part02 Lec08_Classification_ Breast_Cancer_SVM.py

```
# ทำ Feature Scaling
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# เลือกและฝึก SVM โมเดล
model = SVC(kernel='linear')
model.fit(X_train, y_train)

# ทำนายผลและประเมินโมเดล
y_pred = model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

อธิบาย

โค้ดตัวอย่างนี้ใช้ Breast Cancer Dataset จาก scikit-learn เราทำการ Standard Scaling เพื่อแปลงข้อมูลให้มีมาตราส่วนใกล้เคียงกัน จากนั้นฝึกโมเดล SVM โดยใช้ linear kernel และประเมินผลลัพธ์

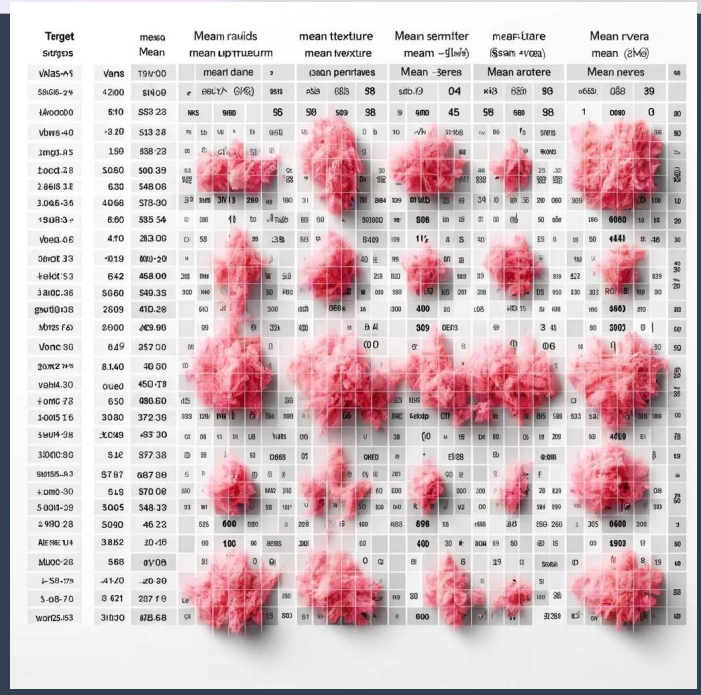
8

8.4 ตัวอย่างการจำแนกข้อมูล Breast Cancer Dataset

8.4.1 โปรแกรม Breast Cancer Dataset จาก scikit-learn

คำอธิบายคอลัมน์

- mean radius: รัศมีเฉลี่ยของเซลล์
- mean texture: ความสม่ำเสมอของเซลล์
- mean perimeter: เส้นรอบวงเฉลี่ยของเซลล์
- mean area: พื้นที่เฉลี่ยของเซลล์
- mean smoothness: ความเรียบเฉลี่ยของเซลล์
- worst radius: รัศมีของเซลล์ที่แย่ที่สุด
- worst texture: ความสม่ำเสมอของเซลล์ที่แย่ที่สุด
- worst perimeter: เส้นรอบวงของเซลล์ที่แย่ที่สุด
- worst area: พื้นที่ของเซลล์ที่แย่ที่สุด
- worst smoothness: ความเรียบของเซลล์ที่แย่ที่สุด
- target: ป้ายกำกับ (0 = มะเร็งชนิดไม่ร้ายแรง, 1 = มะเร็งชนิดร้ายแรง)



ตัวอย่างข้อมูล Breast Cancer Dataset จาก scikit-learn ในรูปแบบตาราง

ตารางนี้แสดงฟีเจอร์หลักบางส่วนที่ใช้ในการวิเคราะห์และจำแนกประเภทของเซลล์มะเร็งเต้านม

8

8.4 ตัวอย่างการจำแนกข้อมูล Breast Cancer Dataset

8.4.1 โปรแกรม Breast Cancer Dataset จาก scikit-learn

คำอธิบายคอลัมน์

- mean radius: รัศมีเฉลี่ยของเซลล์
- mean texture: ความสม่ำเสมอของเซลล์
- mean perimeter: เส้นรอบวงเฉลี่ยของเซลล์
- mean area: พื้นที่เฉลี่ยของเซลล์
- mean smoothness: ความเรียบเฉลี่ยของเซลล์
- worst radius: รัศมีของเซลล์ที่แย่ที่สุด
- worst texture: ความสม่ำเสมอของเซลล์ที่แย่ที่สุด
- worst perimeter: เส้นรอบวงของเซลล์ที่แย่ที่สุด
- worst area: พื้นที่ของเซลล์ที่แย่ที่สุด
- worst smoothness: ความเรียบของเซลล์ที่แย่ที่สุด
- target: ป้ายกำกับ (0 = มะเร็งชนิดไม่ร้ายแรง, 1 = มะเร็งชนิดร้ายแรง)

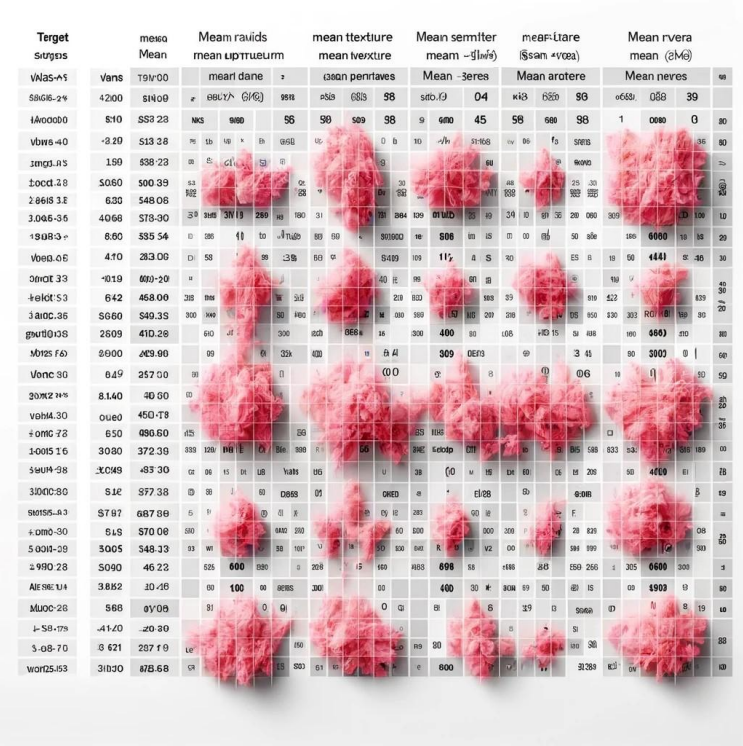
ตัวอย่างข้อมูล Breast Cancer Dataset จาก scikit-learn ในรูปแบบตาราง

ตารางนี้แสดงฟีเจอร์หลักบางส่วนที่ใช้ในการวิเคราะห์และจำแนกประเภทของเซลล์มะเร็งเต้านม

8

8.4 ตัวอย่างการจำแนกข้อมูล Breast Cancer Dataset

8.4.1 โปรแกรม Breast Cancer Dataset จาก scikit-learn



```
PS D:\DataAnalytics_Project> & C:/Users/User/AppData/Local/Micr
py"
Accuracy: 0.9766081871345029
Classification Report:
              precision    recall  f1-score   support

           0       0.97      0.97      0.97        63
           1       0.98      0.98      0.98       108

    accuracy                           0.98       171
   macro avg       0.97      0.97      0.97       171
weighted avg       0.98      0.98      0.98       171

PS D:\DataAnalytics_Project>
```

ตัวอย่างข้อมูล Breast Cancer Dataset จาก scikit-learn ในรูปแบบตาราง

8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

8.5 การประยุกต์ใช้ซัพพอร์ตเวกเตอร์แมกชีน SVM

8.5 การประยุกต์ใช้ซัพพอร์ตเวกเตอร์แมกชีน SVM ถูกใช้ในหลากหลายงาน เช่น:

1. การจดจำลายมือ: ใช้ในการแยกลายมือออกจากกัน
2. การจำแนกอีเมล: ใช้แยกอีเมลออกเป็นสแปมหรือไม่สแปม
3. การตรวจจับวัตถุ: ใช้ในงานภาพเพื่อแยกวัตถุออกจากฉากหลัง
4. การวิเคราะห์และจำแนกประเภทของเซลล์มะเร็งเต้านม (Breast Cancer Dataset จาก scikit-learn)
5. อื่น ๆ

SVM เป็นอัลกอริทึมที่มีประสิทธิภาพมากในการจำแนกข้อมูล โดยเฉพาะเมื่อจำนวนคุณสมบัติ (features) มีมากกว่าจำนวนตัวอย่าง (samples) แต่ก็อาจมีข้อจำกัดในการใช้กับข้อมูลจำนวนมากเนื่องจากความซับซ้อนของการคำนวณ

8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

8.5.1 การประยุกต์ใช้ซัพพอร์ตเวกเตอร์แมกชีน SVM ในการจดจำลายมือ

การประยุกต์ใช้ซัพพอร์ตเวกเตอร์แมกชีน (Support Vector Machine, SVM) ในการจดจำลายมือเป็นที่นิยมมาก เนื่องจากความสามารถของ SVM ในการแยกข้อมูลที่ซับซ้อนออกเป็นคลาสที่แตกต่างกันอย่างแม่นยำ โดยเฉพาะเมื่อลายมือของแต่ละบุคคลมีรูปแบบที่แตกต่างกันอย่างชัดเจน เช่น การเขียนตัวอักษร ตัวเลข หรือแม้แต่การวาดรูปทรงต่าง ๆ

ขั้นตอนในการนำ SVM มาใช้ในการจดจำลายมือ:

1. การเตรียมข้อมูล (Data Preparation)
2. การแปลงภาพลายมือให้เป็นข้อมูลเชิงตัวเลข (เช่น รูปแบบ grayscale หรือ binary)
3. การสร้างโมเดล SVM
4. การฝึกโมเดล (Model Training)
5. การทดสอบโมเดล (Model Testing)
6. การปรับแต่งโมเดล (Model Tuning)

8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

8.5.1 การประยุกต์ใช้ซัพพอร์ตเวกเตอร์แมกชีน SVM ในการจดจำลายมือ

ตัวอย่างการใช้งาน SVM ในการแยกลายมือ:

1. การจดจำตัวเลขเขียนมือ (Handwritten Digit Recognition) เช่น การใช้ SVM กับฐานข้อมูล MNIST ซึ่งเป็นฐานข้อมูลที่เก็บข้อมูลภาพตัวเลข 0-9 ที่เขียนด้วยมือ
2. การจดจำลายเซ็นหรือรูปแบบการเขียนลายมือเฉพาะบุคคล ซึ่งสามารถนำไปใช้ในการพิสูจน์ตัวตน
3. การใช้ SVM สำหรับ OCR (Optical Character Recognition) เพื่อนำมาแปลงลายมือเป็นตัวอักษร

SVM สามารถสร้างผลลัพธ์ที่ดีเมื่อมีการใช้ฟีเจอร์และพารามิเตอร์ที่เหมาะสม และสามารถจัดการกับความซับซ้อนของลายมือได้อย่างมีประสิทธิภาพ

8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

8.5.1 การประยุกต์ใช้ซัพพอร์ตเวกเตอร์แมกชีน SVM ในการจดจำลายมือ

ตัวอย่างการใช้ซัพพอร์ตเวกเตอร์แมกชีน (SVM) ในการจดจำลายมือด้วยฐานข้อมูล MNIST โดยใช้ภาษา Python และไลบรารี scikit-learn

Part01 Lec08_Classification_ MNIST_SVM.py

```
# นำเข้าไลบรารีที่จำเป็น
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import classification_report, accuracy_sc
import matplotlib.pyplot as plt

# โหลดฐานข้อมูล MNIST
digits = datasets.load_digits()
```

```
PS D:\DataAnalytics_Project> & C:/Users/User/AppData/Loca
Accuracy: 0.9888888888888889
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        33
           1       1.00      1.00      1.00        28
           2       1.00      1.00      1.00        33
           3       1.00      0.97      0.99        34
           4       1.00      1.00      1.00        46
           5       0.98      0.98      0.98        47
           6       0.97      1.00      0.99        35
           7       0.97      0.97      0.97        34
           8       1.00      1.00      1.00        30
           9       0.97      0.97      0.97        40

    accuracy                           0.99       360
   macro avg       0.99      0.99      0.99       360
weighted avg       0.99      0.99      0.99       360

PS D:\DataAnalytics_Project>
```

8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

8.5.1 การประยุกต์ใช้ซัพพอร์ตเวกเตอร์แมกชีน SVM ในการจดจำลายมือ

Part02 Lec08_Classification_ MNIST_SVM.py

```
# แสดงตัวอย่างข้อมูลลายมือ
plt.gray()
plt.matshow(digits.images[0])
plt.show()

# แยกฟีเจอร์ (X) และฉลาก (y)
X = digits.data  # ฟีเจอร์ (ภาพลายมือถูกแปลงเป็น 64 ฟีเจอร์)
y = digits.target  # ฉลาก (0-9)

# แบ่งข้อมูลเป็นชุดฝึก (training) และชุดทดสอบ (test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

8.5.1 การประยุกต์ใช้ซัพพอร์ตเวกเตอร์แมกชีน SVM ในการจดจำลายมือ

ตัวอย่างการใช้ซัพพอร์ตเวกเตอร์แมกชีน (SVM) ในการจดจำลายมือด้วยฐานข้อมูล MNIST โดยใช้ภาษา Python และไลบรารี scikit-learn

Part03 Lec08_Classification_ MNIST_SVM.py

```
# สร้างโมเดล SVM โดยใช้ RBF kernel
svm_model = SVC(kernel='rbf', gamma=0.001, C=10)
# ฝึกโมเดล
svm_model.fit(X_train, y_train)
# ทำนายผลด้วยชุดทดสอบ
y_pred = svm_model.predict(X_test)
# แสดงผลลัพธ์ความแม่นยำ
print(f"Accuracy: {accuracy_score(y_test, y_pred)}")
```

```
# แสดงรายงานการจำแนก (classification report)
print(classification_report(y_test, y_pred))

# แสดงตัวอย่างผลลัพธ์การทำนาย
plt.matshow(X_test[0].reshape(8, 8),
cmap='gray')
plt.title(f"Predicted: {y_pred[0]}, True:
{y_test[0]}")
plt.show()
```

8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

8.5.1 การประยุกต์ใช้ซัพพอร์ตเวกเตอร์แมกชีน SVM ในการจดจำลายมือ

ตัวอย่างการใช้ซัพพอร์ตเวกเตอร์แมกชีน (SVM) ในการจดจำลายมือด้วยฐานข้อมูล MNIST โดยใช้ภาษา Python และไลบรารี scikit-learn

อธิบายโค้ด:
นำเข้าไลบรารีที่จำเป็น:
datasets สำหรับโหลดฐานข้อมูล MNIST
train_test_split สำหรับแบ่งข้อมูล
SVC สำหรับสร้างโมเดล SVM
classification_report และ accuracy_score สำหรับประเมินผล
matplotlib สำหรับแสดงภาพตัวอย่างข้อมูล
โหลดฐานข้อมูล MNIST:
digits = datasets.load_digits() ทำการโหลดฐานข้อมูล MNIST ที่มีภาพตัวเลข 0-9 ขนาด 8x8 พิกเซล

8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

8.5.1 การประยุกต์ใช้ซัพพอร์ตเวกเตอร์แมกชีน SVM ในการจดจำลายมือ

ตัวอย่างการใช้ซัพพอร์ตเวกเตอร์แมกชีน (SVM) ในการจดจำลายมือด้วยฐานข้อมูล MNIST โดยใช้ภาษา Python และไลบรารี scikit-learn

แสดงภาพลายมือ:
ใช้ plt.matshow() เพื่อแสดงภาพตัวอย่างลายมือของตัวเลข
การแบ่งข้อมูล:
ข้อมูลถูกแบ่งเป็นชุดฝึก (80%) และชุดทดสอบ (20%) โดยใช้ฟังก์ชัน train_test_split()
สร้างและฝึกโมเดล SVM:
โมเดล SVM ถูกสร้างขึ้นด้วยฟังก์ชัน SVC() โดยใช้ RBF kernel ซึ่งเหมาะกับข้อมูลที่ไม่เป็นเส้นตรง
การประเมินผล:
ใช้ฟังก์ชัน accuracy_score() เพื่อแสดงค่าความแม่นยำ และ classification_report() เพื่อแสดงผลลัพธ์การจำแนกประเภทสำหรับแต่ละตัวเลข
แสดงผลลัพธ์การทำนาย:
แสดงภาพตัวอย่างของข้อมูลชุดทดสอบ พร้อมผลลัพธ์การทำนายจากโมเดล

8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

8.6 สรุป

8.6 สรุป

การจำแนกข้อมูลเป็นกระบวนการสำคัญในด้านการเรียนรู้ของเครื่องแบบมีผู้สอน โดยการใช้ข้อมูลที่มีป้ายกำกับในการฝึกสอนและสร้างโมเดลที่สามารถทำนายประเภทของข้อมูลใหม่ได้ กระบวนการนี้รวมถึงการเตรียมข้อมูล การเลือกอัลกอริทึม การฝึกสอนและทดสอบโมเดล และการประเมินผลโมเดล ซึ่งช่วยให้เราได้โมเดลที่มีความแม่นยำและประสิทธิภาพสูง พร้อมใช้งานในด้านต่างๆ อย่างหลากหลาย

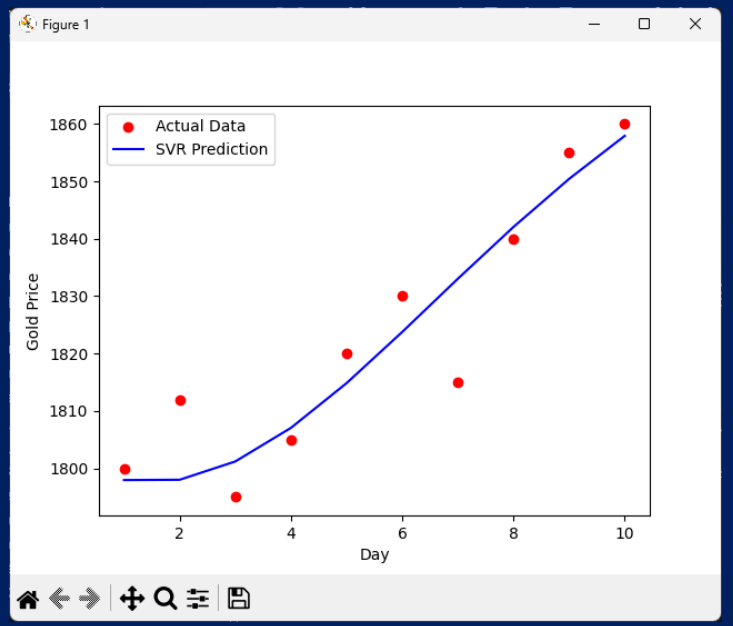
8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

8.6 ตัวอย่างโค้ด Support Vector Regression (SVR) ใน Python สำหรับการทำนายราคาทองคำ

Support Vector Regression (SVR) เป็นเทคนิคของ Support Vector Machine (SVM) ที่ใช้สำหรับการทำนายค่าตัวเลขแบบต่อเนื่อง (Regression) โดย SVR พยายามหาฟังก์ชันที่เหมาะสมที่สุดสำหรับการทำนายข้อมูลที่มีความซับซ้อนสูง และสามารถรับมือกับข้อมูลที่ไม่เป็นเส้นตรงได้ดี โดยใช้ kernel trick เช่น Radial Basis Function (RBF) เพื่อทำให้การเรียนรู้มีประสิทธิภาพมากขึ้น

```
PS C:\AppServ\www\siam2dev_net\E_Learning\DataMining\DM_Projects> & C:/Users/
ec08_svm_gold_price_prediction.py
Mean Absolute Error (MAE): 9.27875072985239
Root Mean Squared Error (RMSE): 10.409346487830376
Predicted Prices for Unknown Days:
Day 11: $1864.46
Day 12: $1869.93
Day 13: $1874.31
```



8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

8.6 ตัวอย่างโค้ด Support Vector Regression (SVR) ใน Python สำหรับการทำนายราคาทองคำ

Part01 Lec08_svm_gold_price_prediction.py

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.svm import SVR
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_absolute_error, mean_squared_error

# ตัวอย่างข้อมูลราคาทองคำ (สมมติว่าเป็นราคาย้อนหลังรายวัน)
data = {
    'Day': np.arange(1, 11),
    'Price': [1800, 1812, 1795, 1805, 1820, 1830, 1815, 1840, 1855, 1860]
}
```

8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

8.6 ตัวอย่างโค้ด Support Vector Regression (SVR) ใน Python สำหรับการทำนายราคาทองคำ

Part02 Lec08_svm_gold_price_prediction.py

```
df = pd.DataFrame(data)

# กำหนด Features (X) และ Target (y)
X = df[['Day']].values
y = df['Price'].values

# แบ่งข้อมูลเป็น Training และ Testing set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# สร้างและ Train โมเดล SVR
scaler_x = StandardScaler()
scaler_y = StandardScaler()
X_train_scaled = scaler_x.fit_transform(X_train)
y_train_scaled = scaler_y.fit_transform(y_train.reshape(-1, 1)).ravel()
```

8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

8.6 ตัวอย่างโค้ด Support Vector Regression (SVR) ใน Python สำหรับการทำนายราคาทองคำ

Part03 Lec08_svm_gold_price_prediction.py

```
svr = SVR(kernel='rbf', C=100, gamma=0.1, epsilon=0.1)
svr.fit(X_train_scaled, y_train_scaled)

# ทำนายค่าทดสอบ
X_test_scaled = scaler_x.transform(X_test)
y_pred_scaled = svr.predict(X_test_scaled)
y_pred = scaler_y.inverse_transform(y_pred_scaled.reshape(-1, 1)).ravel()

# คำนวณค่า Error
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
```

8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

8.6 ตัวอย่างโค้ด Support Vector Regression (SVR) ใน Python สำหรับการทำนายราคาทองคำ

Part04 Lec08_svm_gold_price_prediction.py

```
print(f"Mean Absolute Error (MAE): {mae}")
print(f"Root Mean Squared Error (RMSE): {rmse}")

# ทำนายค่าที่ไม่รู้จัก (เช่น ราคาทองคำในวันที่ 11, 12, 13)
unknown_days = np.array([[11], [12], [13]])
unknown_days_scaled = scaler_x.transform(unknown_days)
unknown_pred_scaled = svr.predict(unknown_days_scaled)
unknown_pred = scaler_y.inverse_transform(unknown_pred_scaled.reshape(-1, 1)).ravel()

print("Predicted Prices for Unknown Days:")
for day, price in zip(unknown_days.flatten(), unknown_pred):
    print(f"Day {day}: ${price:.2f}")
```

8

บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

8.6 ตัวอย่างโค้ด Support Vector Regression (SVR) ใน Python สำหรับการทำนายราคาทองคำ

Part05 Lec08_svm_gold_price_prediction.py

```
# Visualization
plt.scatter(X, y, color='red', label='Actual Data(ข้อมูลจริง)')
plt.plot(X, scaler_y.inverse_transform(svr.predict(scaler_x.transform(X)).reshape(-1, 1)), color='blue',
label='SVR Prediction')
plt.xlabel('Day')
plt.ylabel('Gold Price')
plt.legend()
plt.show()
```

8

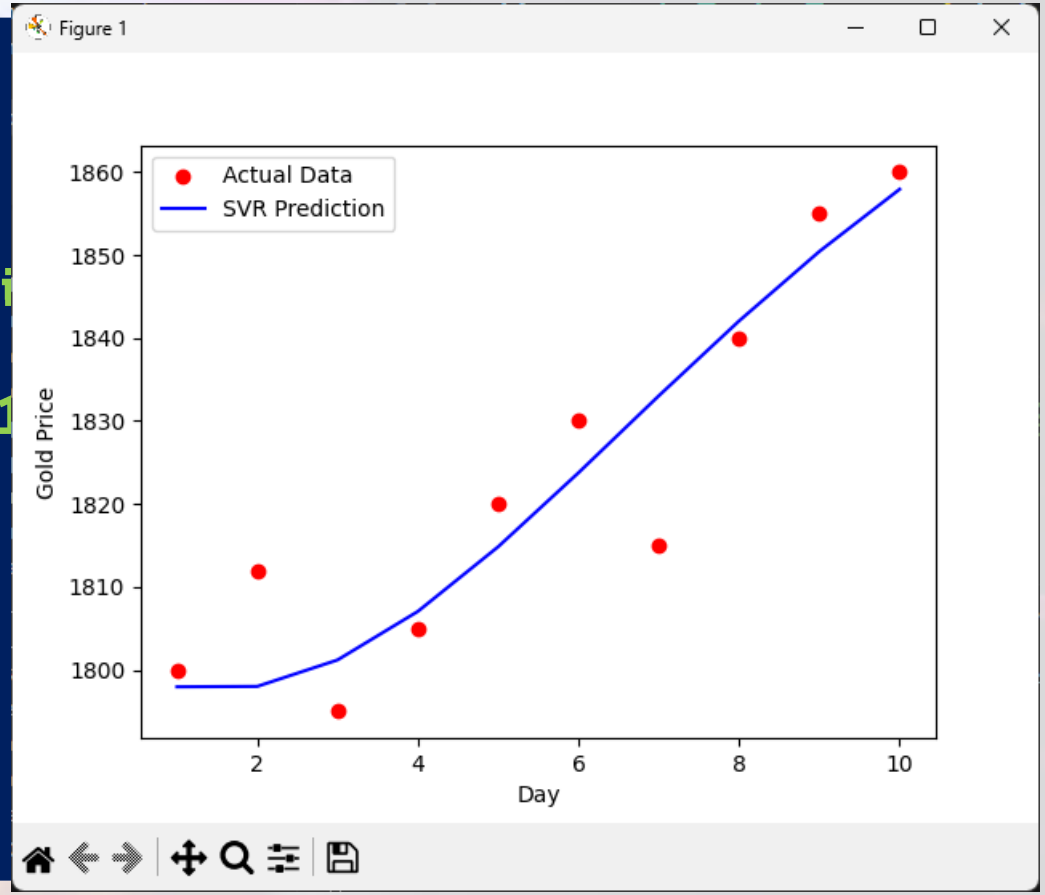
บทที่ 8 การจำแนกข้อมูลด้วยวิธี Support Vector Machine

8.6 ตัวอย่างโค้ด Support Vector Regression (SVR) ใน Python สำหรับการทำนายราคาทองคำ

คำอธิบายโค้ด:

- สร้างชุดข้อมูล - จำลองข้อมูลราคาทองคำรายวัน
- แปลงข้อมูล - ใช้ StandardScaler เพื่อ Normalize ข้อมูล
- ฝึกโมเดล SVR - ใช้ rbf kernel และปรับค่า C, gamma, epsi
- ทำนายและประเมินผล - คำนวณ MAE และ RMSE
- ทำนายค่าที่ไม่รู้จัก - ทำนายราคาทองคำในอนาคต (วันที่ 11, 12, 1
- วาดกราฟ - เปรียบเทียบค่าจริงกับค่าที่โมเดลทำนาย

```
PS C:\AppServ\www\siam2dev_net\E_Learning\DataMining\DM_Projects> & C:/Users/
ec08_svm_gold_price_prediction.py
Mean Absolute Error (MAE): 9.27875072985239
Root Mean Squared Error (RMSE): 10.409346487830376
Predicted Prices for Unknown Days:
Day 11: $1864.46
Day 12: $1869.93
Day 13: $1874.31
```



อ้างอิง

1. เว็บไซต์
- Itsci MJU
- มหาวิทยาลัยแม่โจ้. (n.d.). *Itsci MJU*. สืบค้นจาก https://www.itsci.mju.ac.th
2. บทความจากเว็บไซต์ Softnix
- จิราพร, ส. (2018, 6 กันยายน). *ว่าด้วย K-Means และการประยุกต์*. Softnix. สืบค้นจาก https://www.softnix.co.th/2018/09/06/ว่าด้วย-k-means-และการประยุกตร/
3. บทความจากเว็บไซต์ Coraline
- ปิยะ, ร. (2018, 5 พฤศจิกายน). *Customer segmentation by clustering model*. Coraline. สืบค้นจาก https://www.coraline.co.th/single-post/2018/11/05/Customer-Segmentation-by-clustering-model