บทที่ 5 การเรียนรู้ของเครื่องจักร (Machine Learning)

หัวข้อหลัก

- แบบจำลอง (model) คืออะไร?
- การเรียนรู้ของเครื่องจักร (Machine learning) คืออะไร?
- ปัญหาการจดจำ (Overfitting) และการไม่สามารถเรียนรู้รูปแบบได้ (Underfitting)
- ความถูกต้องของแบบจำลอง (Correctness)
- การถ่วงดุลค่าความโน้มเอียงกับค่าความแปรปรวน (Bias-Variance Tradeoff)

งานหลักของนักวิทยาศาสตร์ข้อมูล คือ การแปลงปัญหาทางธุรกิจ (business problem) ไปเป็นโจทย์ปัญหาเกี่ยวกับข้อมูล (data problem) จากนั้นจึงทำการเก็บรวบรวมข้อมูล ทำความเข้าใจข้อมูล จัดเตรียมข้อมูลให้พร้อมสำหรับการสร้างแบบจำลอง จากนั้นจึงเลือกสรรเทคนิควิธีการของการเรียนรู้ของเครื่องจักร (machine learning) มาใช้สกัดรูปแบบที่แฝงอยู่ภายในชุดข้อมูล เพื่อนำไปใช้เพิ่มคุณค่าให้กับธุรกิจต่อไป จะเห็นได้ว่าการเรียนรู้ของเครื่องจักรเป็นเพียงส่วนหนึ่งของกระบวนการทางวิทยาศาสตร์ข้อมูลเท่านั้น แต่ก็จำเป็นอย่างยิ่งที่นักวิทยาศาสตร์ข้อมูลจะต้องศึกษาและทำความเข้าใจเกี่ยวกับการเรียนรู้ของเครื่องจักรอย่างถ่องแท้ ในบทนี้ เราจะทำความเข้าใจเกี่ยวกับแนวคิดพื้นฐานของการเรียนรู้ของเครื่องจักร (Machine Learning) โดยจะเป็นการสรุปเฉพาะเนื้อหาที่เกี่ยวข้องโดยตรงกับวิทยาศาสตร์ข้อมูล สำหรับผู้ที่สนใจเรียนรู้เพิ่มเติมเกี่ยวกับวิชานี้ สามารถอ่านเพิ่มเติมได้จากตำราที่เกี่ยวข้อง [1,2,4,5]

5.1 แบบจำลอง (Model)

แบบจำลองหรือโมเดล คือข้อกำหนดที่แสดงความสัมพันธ์ทางคณิตศาสตร์ที่มีอยู่ระหว่างตัวแปรต่าง ๆ ตัวอย่างเช่น

- หากเราต้องการประเมินราคาบ้านเดี่ยวในเขตจังหวัดชลบุรี เราอาจจะสร้างโมเดลที่รับค่าอินพุท เช่น ราคาที่ดินเฉลี่ย จำนวนห้องนอน ขนาดพื้นที่ใช้สอย และให้ค่าเอาท์พุท เป็นราคาบ้าน ในกรณีนี้ โมเดลที่สร้างขึ้นจะแสดงความสัมพันธ์ระหว่าตัวแปรอิสระ (independent variables) สามตัวแปรซึ่งเป็นค่าอินพุท ได้แก่ ราคาที่ดินเฉลี่ย จำนวนห้องนอน และ ขนาดพื้นที่ใช้สอย กับตัวแปรตาม (dependent variable) คือ ราคาบ้าน ซึ่งเป็นค่าเอาท์พุท
- ในการทำนายปริมาณการใช้ไฟฟ้าของประชากรในจังหวัดชลบุรี เราอาจจะสร้างโมเดลที่รับค่าอินพุท เป็นปริมาณการใช้ไฟฟ้าย้อนหลังสามวัน และ คำพยากรณ์อุณหภูมิเฉลี่ยของจังหวัดชลบุรี และให้ค่าเอาท์พุทเป็นปริมาณการใช้ไฟฟ้า (หน่วย MWh) เป็นต้น ในกรณีนี้ โมเดลที่สร้างขึ้น จะแสดงความสัมพันธ์ระหว่างตัวแปรอิสระคือค่าอินพุท ได้แก่ อุณหภูมิเฉลี่ย และปริมาณการใช้ไฟฟ้าย้อนหลัง กับตัวแปรตาม คือ ปริมาณการใช้ไฟฟ้าในวันถัดไป ซึ่งเป็นค่าเอาท์พุท
- หากเราต้องการจำแนกประเภทของอีเมล์ออกเป็น อีเมล์ปกติ และ อีเมล์ขยะ เราอาจจะสร้างโมเดลที่รับค่าอินพุท เป็นชื่อเรื่อง ชื่อผู้ส่ง และคำสำคัญในข้อความอีเมล์ และให้ค่าเอาท์พุทเป็นประเภทของอีเมล์ (อีเมล์ปกติ หรือ อีเมล์ขยะ) ในกรณีนี้ โมเดลที่สร้างขึ้น จะแสดงความสัมพันธ์ระหว่างตัวแปรอิสระคือค่าอินพุท ได้แก่ ชื่อเรื่อง ชื่อผู้ส่ง และคำสำคัญ กับตัวแปรตามคือประเภทของอีเมล์ ซึ่งเป็นค่าเอาท์พุท

5.2 การเรียนรู้ของเครื่องจักร (Machine Learning)

มีผู้ให้คำนิยามของการเรียนรู้ของเครื่องจักรไว้หลายคำนิยาม แต่หากกล่าวโดยสรุป การเรียนรู้ของเครื่องจักร ก็คือ**การสร้างและใช้โมเดลที่เรียนรู้ได้จากข้อมูล** ในบางบริบท ก็เรียกการเรียนรู้จากข้อมูลว่า การสร้างโมเดลการทำนายค่า (*predictive modeling*) หรือ การทำเหมืองข้อมูล (data mining)

โดยทั่วไป เป้าหมายของการเรียนรู้ของเครื่องจักรคือการสร้างโมเดลที่เราสามารถนำไปใช้ทำนายค่าเป้าหมายสำหรับข้อมูลอินพุทที่ไม่เคยพบมาก่อน (unseen data) หรือข้อมูลใหม่ได้

สำหรับคำนิยามอื่นๆ ที่พบในแหล่งข้อมูลต่างๆ เช่น

- Machine Learning คือ วิทยาศาสตร์แขนงหนึ่งที่ศึกษาเกี่ยวกับอัลกอริทึมและโมเดลทางสถิติ ที่ระบบคอมพิวเตอร์ใช้กระทำงานบางอย่างได้โดยไม่จำเป็นต้องมีคำสั่งที่ชัดเจน แต่อาศัยรูปแบบ (patterns) และการอนุมาน (inference) จากข้อมูลแทน (อ้างอิง: https://en.wikipedia.org/wiki/Machine_learning)
- Machine Learning คือ การศึกษาเกี่ยวกับคอมพิวเตอร์อัลกอริทึม ที่สามารถพัฒนาตัวเองได้แบบอัตโนมัติโดยการเรียนรู้จากประสบการณ์ (อ้างอิง: http://www.cs.cmu.edu/~tom/mlbook.html)
- Machine Learning คือ สาขาหนึ่งของปัญญาประดิษฐ์ (artificial intelligence) ที่ค้นหาคำตอบของคำถามว่า “เราจะสามารถสร้างระบบคอมพิวเตอร์ที่พัฒนาตัวเองได้โดยอัตโนมัติผ่านการเรียนรู้จากประสบการณ์ได้อย่างไร” และ “อะไรคือกฎพื้นฐานที่ควบคุมกระบวนการเรียนรู้ต่าง ๆ”

5.3 โอเวอร์ฟิตติ่งและอันเดอร์ฟิตติ่ง (Overfitting and Underfitting)

ปัญหาที่มักเกิดขึ้นในการสร้างโมเดลที่เรียนรู้จากข้อมูล มี สองปัญหาหลัก คือ overfitting (โอเวอร์ฟิตติ่ง) และ underfitting (อันเดอร์ฟิตติ่ง)

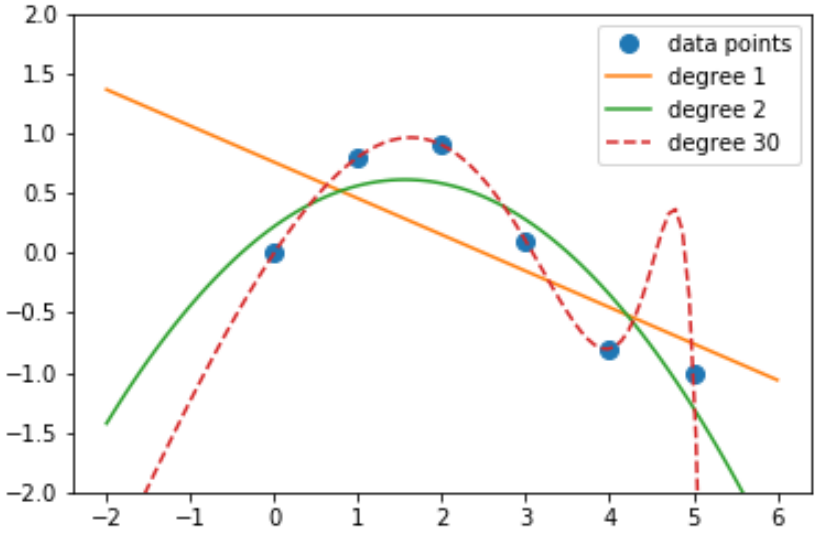
โอเวอร์ฟิตติ่ง คือปรากฏการณ์ที่โมเดลจดจำข้อมูล แทนที่จะเรียนรู้รูปแบบที่แฝงอยู่ในเนื้อข้อมูลส่งผลให้โมเดลที่ได้มีประสิทธิภาพต่ำเมื่อนำไปใช้กับชุดข้อมูลใหม่ที่แตกต่างไปจากชุดข้อมูลฝึกฝน ในทางเทคนิคเรียกว่า โมเดลไม่สามารถ generalize บนชุดข้อมูลใหม่ได้ โอเวอร์ฟิตติ่ง อาจเกิดจากการเลือกใช้โมเดลที่มีความซับซ้อนมากเกินไป หรือ อาจเกิดจากการที่โมเดลเรียนรู้ข้อมูลรบกวนหรือ noise แทนที่จะเรียนรู้รูปแบบที่แท้จริง หรือ อาจเกิดจากการที่โมเดลเรียนรู้ข้อมูลอินพุทแบบเฉพาะเจาะจงแทนที่จะเรียนรู้ปัจจัยที่สำคัญต่อการทำนายค่าเอาท์พุท

อันเดอร์ฟิตติ่ง คือการที่โมเดลที่ได้จากการเรียนรู้มีประสิทธิภาพต่ำทั้งบนชุดข้อมูลฝึกฝน และชุดข้อมูลทดสอบ โดยส่วนใหญ่มักเกิดเมื่อผู้สร้างโมเดลเลือกใช้โมเดลที่มีความสามารถในการเรียนรู้รูปแบบน้อยเกินกว่าความซับซ้อนของชุดข้อมูล ซึ่งสามารถแก้ไขได้โดยการทดลองใช้โมเดลที่มีความสามารถในการเรียนรู้รูปแบบที่มากขึ้นไปทีละขั้น จนกว่าจะได้โมเดลที่เหมาะสม

สมมติว่า เราสร้างโมเดล polynomial regression บนชุดข้อมูลที่ประกอบด้วย 6 จุดข้อมูลดังรูป 5.1 จากรูปจะเห็นได้ว่า โมเดลของ polynomial degree 1 มีความคลาดเคลื่อนจากค่าของจุดข้อมูลมากเนื่องจากโมเดลดังกล่าว *underfit* กับชุดข้อมูล (โมเดลมีประสิทธิภาพต่ำทั้งบนชุดข้อมูลฝึกฝนและบนชุดข้อมูลทดสอบ) ส่วนโมเดลของ polynomial degree 30 ให้ค่าคำทำนายถูกต้องทุกจุด แต่เกิดจากการที่โมเดลจดจำค่าผลลัพธ์ของอินพุท โมเดลดังกล่าวจึงเป็นโมเดลที่ไม่เหมาะกับการนำไปใช้งาน เนื่องจากเป็นโมเดลที่ *overfit* กับชุดข้อมูล โมเดลในรูปที่เหมาะกับการนำไปใช้งานมากที่สุดคือโมเดลของ polynomial degree 2

วิธีการแก้ไขเมื่อโมเดลเกิด underfitting ทำได้โดยการเลือกโมเดลรูปแบบอื่นที่มีความสามารถในการเรียนรู้รูปแบบที่ซับซ้อนเพิ่มขึ้น ส่วนปัญหา overfitting เป็นปัญหาที่แก้ไขได้ไม่ง่ายนัก แต่วิธีการตรวจจับและป้องกัน overfitting ที่ง่ายที่สุดก็คือการแบ่งชุดข้อมูลออกเป็นสองชุดคือ ชุดข้อมูลฝึกฝน (training dataset) และ ชุดข้อมูลทดสอบ (testing dataset) แล้วใช้

ชุดข้อมูลฝึกฝนสำหรับสร้างโมเดล ส่วนชุดข้อมูลทดสอบใช้สำหรับทดสอบประสิทธิภาพของโมเดลที่ได้ หากมี overfitting เกิดขึ้น เราจะสามารถทราบได้โดยดูจากประสิทธิภาพของโมเดล กล่าวคือ โมเดลจะมีประสิทธิภาพดีบนชุดข้อมูลฝึกฝน แต่จะมีประสิทธิภาพต่ำบนชุดข้อมูลทดสอบ ในกรณีที่พบว่ามี overfitting เกิดขึ้น สามารถแก้ไขได้หลายวิธี เช่น การลดความซับซ้อนของโมเดลลง และการเพิ่มขนาดชุดข้อมูล เป็นต้น



รูปที่ 5.1 แสดง polynomial regression model ที่ดีกรี 1, 2, และ 30.

รูปที่ 5.2 แสดงซอร์สโค้ดไพธอน สำหรับแบ่งชุดข้อมูลออกเป็นชุดฝึกฝนและชุดทดสอบ

```
import random
from typing import TypeVar, List, Tuple

X = TypeVar('X')
Y = TypeVar('Y')

def split_data(data: List[X], prob: float) -> Tuple[List[X], List[X]]:
    """Split data into fractions [prob, 1-prob]"""
    data = data[:]
    random.shuffle(data)
    cut = int(len(data) * prob)
    return data[:cut], data[cut:]

def train_test_split(xs: List[X],
                     ys: List[Y],
                     test_pct: float) -> Tuple[List[X], List[X],
                                               List[Y], List[Y]]:
    idxs = [i for i in range(len(xs))]
    train_idxs, test_idxs = split_data(idxs, 1-test_pct)

    return ([xs[i] for i in train_idxs],
            [xs[i] for i in test_idxs],
            [ys[i] for i in train_idxs],
            [ys[i] for i in test_idxs])

xs = [x for x in range(1000)]
ys = [2 * x for x in xs]
x_train, x_test, y_train, y_test = train_test_split(xs, ys, 0.25)

assert len(x_train) == len(y_train) == 750
assert len(x_test) == len(y_test) == 250
assert all(y == 2 * x for x, y in zip(x_train, y_train))
assert all(y == 2 * x for x, y in zip(x_test, y_test))
```

รูปที่ 5.2 ตัวอย่างการแบ่งชุดข้อมูลออกเป็น training dataset และ test dataset (นำมาจาก [3])

5.4 การวัดความถูกต้อง (Correctness)

ในหัวข้อนี้จะกล่าวถึง วิธีการวัดความถูกต้องของโมเดลทำนายค่า (predictive model) สมมติว่า เราสร้างโมเดลสำหรับทำนายประเภทของอีเมล์ว่า เป็นอีเมล์ขยะหรือไม่ อีเมล์แต่ละฉบับที่ถูกป้อนให้กับโมเดลของเราจะสามารถแบ่งออกได้เป็นสี่ประเภทดังนี้คือ

1. **True Positive** คือ อีเมล์ที่เป็นอีเมล์ขยะ และโมเดลของเราทำนายถูกต้องว่าเป็นอีเมล์ขยะ
2. **False Positive (Type 1 error)** คือ อีเมล์ที่ไม่ใช่อีเมล์ขยะ แต่โมเดลของเราทำนายว่าเป็นอีเมล์ขยะ
3. **False Negative (Type 2 error)** คือ อีเมล์ที่เป็นอีเมล์ขยะ แต่โมเดลของเราทำนายว่าไม่เป็นอีเมล์ขยะ
4. **True Negative** คือ อีเมล์ที่ไม่ใช่อีเมล์ขยะ และโมเดลของเราทำนายถูกต้องว่าไม่ใช่อีเมล์ขยะ

ในการทดสอบประสิทธิภาพของโมเดล เรามักจะทำการนับจำนวน True Positive, False Positive, False Negative, และ True Negative แล้วนำมาสร้างเป็น confusion matrix ดังตาราที่ 5.1

ตารางที่ 5.1 ตัวอย่าง Confusion Matrix

	อีเมล์ขยะ	**ไม่ใช่อีเมล์ขยะ**	**รวม**
โมเดลทำนายว่าเป็นอีเมล์ขยะ	*True Positive* *85*	*False Positive* *20*	**105**
โมเดลทำนายว่าไม่ใช่อีเมล์ขยะ	*False Negative* *15*	*True Negative* *880*	**895**
รวม	**100**	**900**	**1000**

จากข้อมูล confusion matrix เราสามารถคำนวณประสิทธิภาพของโมเดลประเภท binary classification ได้หลายวิธี โดยวิธีประเมินสิทธิภาพที่ใช้บ่อยสำหรับโมเดลทำนายค่า มี 3 วิธีดังนี้คือ

- **Accuracy** คือ อัตราส่วนของจำนวนครั้งที่โมเดลทำนายค่าได้ถูกต้อง ต่อ จำนวนข้อมูลทดสอบทั้งหมด

```
def accuracy(true_positive: int, false_positive: int,
             false_negative: int, true_negative: int) -> float:
    correct = true_positive + true_negative
    total = true_positive + false_positive + false_negative + true_negative
    return correct / total

assert accuracy(85, 20, 15, 880) == (85+880) / 1000
```

- **Precision** คือ อัตราส่วนของจำนวนครั้งที่ทำนายถูก ต่อ จำนวนข้อมูลทดสอบที่ถูกทำนายว่าเป็น positive class

```
def precision(true_positive: int, false_positive: int,
              false_negative: int, true_negative: int) -> float:
    return true_positive / (true_positive + false_positive)

assert precision(85, 20, 15, 880) == 85/(85+20)
```

- **Recall** คือ อัตราส่วนของจำนวนครั้งที่ทำนายถูก ต่อ จำนวนข้อมูลทดสอบเป็น positive class

```
def recall(true_positive: int, false_positive: int,
           false_negative: int, true_negative: int) -> float:
    return true_positive / (true_positive + false_negative)

assert recall(85, 20, 15, 880) == 85 / (85 + 15)
```

- **F1 score** คือ ค่า Harmonic Mean ของ Precision และ Recall ค่า F1 score ที่ดีที่สุดคือ 1 (ทั้ง precision และ recall มีค่า optimal) และค่าแย่ที่สุดคือ 0

```
def f1_score(true_positive: int, false_positive: int,
             false_negative: int, true_negative: int) -> float:
    P = precision(true_positive, false_positive, false_negative, true_negative)
    R = recall(true_positive, false_positive, false_negative, true_negative)

    return 2 * P * R / (P + R)
```

5.5 การถ่วงดุลค่าความโน้มเอียงกับค่าความแปรปรวน (Bias-Variance Tradeoff)

Bias-Variance Tradeoff เป็นคุณสมบัติของโมเดลทำนายค่า (predictive model) ที่ทำให้การพยายามลดค่าความผิดพลาดที่เกิดจากสองแหล่งต่อไปนี้ เกิดความขัดแย้งกัน

- Bias Error คือ ความผิดพลาดของโมเดลที่เกิดจากการใช้สมมติฐานที่ไม่ถูกต้องของอัลกอริทึมการเรียนรู้ โมเดลที่มีค่า bias สูง จะไม่สามารถเรียนรู้รูปแบบความสัมพันธ์ระหว่างอินพุทและเอาท์พุทได้ (Underfitting)
- Variance คือ ความผิดพลาดอันเนื่องมาจากความอ่อนไหว (sensitivity) ต่อความผันผวนในชุดฝึกฝน โมเดลที่มีค่า variance สูง จะเรียนรู้ random noise ในชุดข้อมูลฝึกฝน แทนที่จะเรียนรู้รูปแบบความสัมพันธ์ระหว่างอินพุทกับเอาท์พุท (Overfitting)

Bias-Variance Tradeoff เกี่ยวข้องกับ overfitting และ underfitting ที่ได้กล่าวไปในหัวข้อก่อนหน้า โดยเป็นอีกมุมมองหนึ่งของปัญหา overfitting และ underfitting กล่าวคือ เป็นการมองปัญหาในแง่ของความสัมพันธ์กับค่าความผิดพลาดของโมเดล โดยในมุมมองนี้ จะเห็นได้ว่า เมื่อโมเดล underfit กับชุดข้อมูล แสดงว่าค่า bias error สูง แต่ค่า variance ต่ำ ดังนั้นจากนิยามของ bias error ข้างต้น วิธีการแก้ไขจึงทำได้โดยการเพิ่มจำนวนฟีเจอร์ของจุดข้อมูล (features คือค่าตัวแปรอินพุทแต่ละตัวที่ป้อนให้กับโมเดล) หรือเลือกอัลกอริทึมการเรียนรู้ที่มีสมรรถนะในการเรียนรู้ดีขึ้น ในทางตรงกันข้ามหากโมเดล overfit กับชุดข้อมูล แสดงว่าค่าความผิดพลาดจาก variance สูง แต่มีค่า bias ต่ำ ดังนั้นจากนิยามของค่าความผิดพลาดจาก variance ข้างต้น วิธีการแก้ไขจึงทำได้โดยจากลด sensitivity ของโมเดลลง ซึ่งทำได้โดยการลดจำนวนฟีเจอร์ของจุดข้อมูลลง หรือเพิ่มจำนวนข้อมูลฝึกฝนให้มากขึ้น

จากคำอธิบายข้างต้น จะเห็นได้ว่า หากชุดข้อมูลมีจำนวนฟีเจอร์น้อยเกินไปจะทำให้โมเดล underfit ในทางกลับกันหากชุดข้อมูลมีจำนวนฟีเจอร์มากเกินไปจะทำให้โมเดล overfit จากข้อสังเกตนี้ ได้นำไปสู่วิธีแก้ไข overfitting อีกแนวทางหนึ่ง คือ การทำ feature extraction and selection ซึ่งมีแนวคิดหลักคือ การลดจำนวนฟีเจอร์ของข้อมูลลงโดยการคัดเลือกเฉพาะฟีเจอร์ที่สำคัญ (ในกรณีของการจำแนกประเภท คือฟีเจอร์ที่สามารถใช้จำแนกประเภทข้อมูลได้ดี) หรือโดยการสร้างฟีเจอร์ใหม่จากฟีเจอร์ที่มีอยู่เดิม

แบบฝึกหัด

1. จงยกตัวอย่างปัญหาที่เหมาะสมกับการแก้ปัญหาโดยการเรียนรู้จากข้อมูล (learning from data)
2. ทำไมในการสร้างของโมเดลการเรียนรู้ของเครื่องจักร เราจึงควรแบ่งชุดข้อมูลออกเป็น สองชุด คือ ชุดข้อมูลฝึกฝน และชุดข้อมูลทดสอบ?
3. จงยกตัวอย่างวิธีการแก้ปัญหา overfitting ที่เกิดขึ้นในการสร้างโมเดลการเรียนรู้จากข้อมูล
4. เมื่อค่าความผิดพลาดเนื่องจาก variance สูง จะมีส่งผลกระทบต่อประสิทธิภาพของโมเดลอย่างไร overfit หรือ underfit?
5. กำหนด confusion matrix ของผลลัพธ์การประเมินตัวจำแนกประเภทอีเมล์ขยะ ดังตารางที่ 5.2 จงคำนวณหาค่า accuracy, precision, recall, และ f1-score

ตารางที่ 5.2 ตัวอย่าง Confusion Matrix

	อีเมล์ขยะ	**ไม่ใช่อีเมล์ขยะ**	**รวม**
โมเดลทำนายว่าเป็นอีเมล์ขยะ	425	75	**500**
โมเดลทำนายว่าไม่ใช่อีเมล์ขยะ	25	475	**500**
รวม	**450**	**550**	**1000**

เอกสารอ้างอิง

[1] Christopher M. Bishop. Pattern Recognition and Machine Learning, Springer, 2011.

[2] Ian H. Witten, Eibe Frank, et al. Data Mining: Practical Machine Learning Tools and Techniques, Morgan Kaufmann, 2016.

[3] Joel Grus. Data Science from Scratch (2ed), O'Reilly Media, Inc., 2019.

[4] Stephen Marsland. Machine Learning: An Algorithmic Perspective (2ed), CRC Press, 2014

[5] Yaser S. Abu-Mostafa and Malik Magdon-Ismail, and Hsuan-Tien Lin. Learning from Data: A Short Course, AMLBook 2012.