การศึกษาการทำนายค่าความเสถียรของวัคซีนเอ็มอาร์เอ็นเอสำหรับโรคโควิด19 ด้วยวิธีการเรียนรู้ของเครื่องจักร

PREDICTING STABLE COVID-19 mRNA VACCINE BY MACHINE LEARNING

ณัฐนรี พอสูงเนิน

บัณฑิตวิทยาลัย มหาวิทยาลัยศรีนครินทรวิโรฒ

2566

การศึกษาการทำนายค่าความเสถียรของวัคซีนเอ็มอาร์เอ็นเอสำหรับโรคโควิด19 ด้วยวิธีการเรียนรู้ของเครื่องจักร

ณัฐนรี พอสูงเนิน

สารนิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
วิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการข้อมูล
คณะวิทยาศาสตร์ มหาวิทยาลัยศรีนครินทรวิโรฒ
ปีการศึกษา 2566

PREDICTING STABLE COVID-19 mRNA VACCINE BY MACHINE LEARNING

NUTNAREE POOSUNGNOEN

A Master's Project Submitted in Partial Fulfillment of the Requirements

for the Degree of MASTER OF SCIENCE

(Data Science)

Faculty of Science, Srinakharinwirot University

2023

สารนิพนธ์

เรื่อง

การศึกษาการทำนายค่าความเสถียรของวัคซีนเอ็มอาร์เอ็นเอสำหรับโรคโควิด19

ด้วยวิธีการเรียนรู้ของเครื่องจักร

ของ

ณัฐนรี พอสูงเนิน

ได้รับอนุมัติจากบัณฑิตวิทยาลัยให้นับเป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

ปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการข้อมูล

ของมหาวิทยาลัยศรีนครินทรวิโรฒ

(รองศาสตราจารย์ นายแพทย์ฉัตรชัย เอกปัญญาสกุล)

คณบดีบัณฑิตวิทยาลัย

คณะกรรมการสอบปากเปล่าสารนิพนธ์

..... ที่ปรึกษาหลัก

(อาจารย์ ดร.ศุภร คนธภักดี)

..... ประธาน

(ผู้ช่วยศาสตราจารย์ ดร.รัตน์ชัยนันท์ ธรรมสุจริต)

..... กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.ศิริสรรพ เหล่าหะเกียรติ)

ชื่อเรื่อง	การศึกษาการทำนายค่าความเสถียรของวัคซีนเอ็มอาร์เอ็นเอสำหรับโรคโควิด19 ด้วยวิธีการเรียนรู้ของเครื่องจักร
ผู้วิจัย	ณัฐนรี พอสูงเนิน
ปริญญา	วิทยาศาสตรมหาบัณฑิต
ปีการศึกษา	2566
อาจารย์ที่ปรึกษา	อาจารย์ ดร. ศุภร คนธภักดี

จากการระบาดของโรคโควิด19 มีการนำวัคซีนเอ็มอาร์เอ็นเอที่มีประสิทธิภาพมาใช้ในการป้องกันโรค งานวิจัยนี้นำเสนอวิธีการเรียนรู้ของเครื่องจักรโดยใช้เทคนิค 4 แบบ คือ XGboost, Random Forest, Catboost และ LightGBM เพื่อทำนายความเสถียรของการย่อยสลายของลำดับอาร์เอ็นเอ โดยเทคนิค LightGBM เป็นตัวทำนายที่ดีที่สุดโดยได้ MCRMSE เท่ากับ 0.31265 โดยผลลัพธ์ที่ได้ชี้ให้เห็นความเป็นไปได้ในการใช้วิธีการดังกล่าวในการพัฒนาวัคซีนเอ็มอาร์เอ็นเอในอนาคต

คำสำคัญ : โรคโควิด-19, วัคซีนเอ็มอาร์เอ็นเอ, การเรียนรู้ของเครื่องจักร, การย่อยสลาย

Title	PREDICTING STABLE COVID-19 mRNA VACCINE BY MACHINE LEARNING
Author	NUTNAREE POOSUNGNOEN
Degree	MASTER OF SCIENCE
Academic Year	2023
Thesis Advisor	Dr. Subhorn Khonthapagdee

As a result of the COVID-19 outbreak, an effective mRNA vaccine has been used to prevent disease. This research proposes a machine learning method using four techniques: XGboost, Random Forest, Catboost, and LightGBM to predict the stability of RNA sequence degradation. It was found that LightGBM was the best predictor. The results found that the MCRMSE is 0.31265, with the results suggesting the possibility of using this method in the future development of the MRNA vaccine.

Keyword : COVID-19, mRNA vaccine, Machine learning, Degradation

กิตติกรรมประกาศ

สารนิพนธ์นี้สำเร็จลุล่วงได้ด้วยความอนุเคราะห์จาก ดร.ศุภร คนธภักดี และ ผศ.ดร.จันตรี ผลประเสริฐ อาจารย์ที่ปรึกษาทั้ง 2 ท่านที่ให้คำปรึกษา คำแนะนำในการทำสารนิพนธ์ ตลอดจนสนับสนุนข้อมูลทางวิชาการ

ขอกราบขอบพระคุณคณะกรรมการสอบสารนิพนธ์ ที่ได้ให้คำแนะนำและข้อเสนอแนะสำหรับการปรับปรุงสารนิพนธ์ และการใช้เทคนิคการเรียนรู้ของเครื่องเป็นเครื่องมือในการวิเคราะห์และแก้ไขปัญหาทางข้อมูลต่อไป

ขอกราบขอบพระคุณ บัณฑิตวิทยาลัย และ คณะวิทยาศาสตร์ มหาวิทยาลัยศรีนครินทรวิโรฒ สำหรับการช่วยเหลือด้านเอกสารและการจัดทำเล่มสารนิพนธ์

ขอกราบขอบพระคุณ บิดา มารดา และ ขอขอบคุณ พี่เอกปรียา ใบสนิ สำหรับกำลังใจในการทำสารนิพนธ์นี้

ณัฐนรี พอสูงเนิน

สารบัญ

หน้า

สารบัญตาราง

หน้า

สารบัญรูปภาพ

หน้า

บทที่ 1
บทนำ

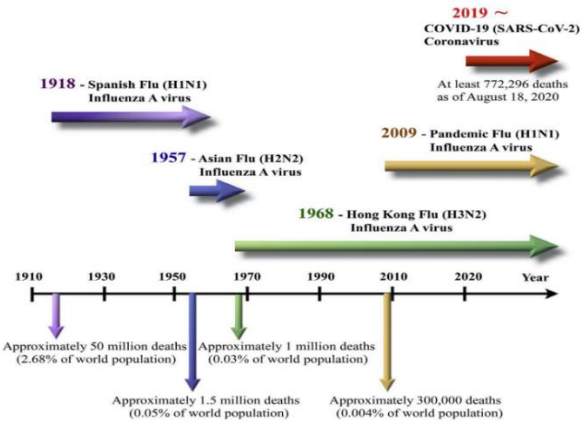
ความสำคัญและความเป็นมาของงานวิจัย

โรคโควิด-19 (covid-19) คือโรคติดต่อเกิดจากไวรัสโคโรนา 2019-nCoV พบครั้งแรกที่เมืองอู่ฮั่น ประเทศจีน ไวรัสมีการแพร่เชื้อระหว่างคนเหมือนกับไข้หวัดใหญ่ ซึ่งไวรัส 2019-nCoV ระบาดทั่วประเทศจีนภายในเวลาเพียงไม่กี่สัปดาห์ และยังมีการระบาดไปยังประเทศต่าง ๆ ในทุกทวีปทั่วโลกภายในเวลาไม่นาน การแพร่ระบาดของเชื้อไวรัส 2019-nCoV นี้เป็นการระบาดใหญ่ (pandemic) ที่สร้าง ผลกระทบในทุกประเทศทั่วโลก ดังภาพประกอบที่ 1 ซึ่งครั้งนี้จัดเป็นการระบาดใหญ่ครั้งที่ 5 หลังจากการระบาด ใหญ่ครั้งที่ 4 เมื่อปี ค.ศ. 2009 ซึ่งเกิดจากเชื้อไวรัสไข้หวัดใหญ่สายพันธุ์ H1N1

โดยสถานการณ์ในประเทศไทย ในช่วงเดือนมิถุนายน 2023 พบจำนวนผู้ป่วยรายใหม่ที่รักษาในโรงพยาบาล จำนวน 3,085 ราย และพบเสียชีวิตจำนวน 68 ราย ซึ่งผู้เสียชีวิตทั้งหมดเป็นผู้ที่ไม่ได้รับวัคซีนครบ 2 เข็ม หรือไม่ได้รับวัคซีนเข็มกระตุ้น ดังนั้นการฉีดวัคซีนเข็มกระตุ้นหรือวัคซีนประจำปี จึงเป็นสิ่งสำคัญในการลดอาการป่วยหนักและเสียชีวิตจากโควิด 19 ได้

วัคซีนโควิด 19 มีประสิทธิภาพในการป้องกันการติดเชื้อไม่สมบูรณ์ ช่วง 4 เดือนแรก และสำหรับป้สายพันธุ์กลายพันธุ์ จะต้องทำการฉีดอย่างน้อย 3 เข็ม ถึงจะสามารถป้องกันความรุนแรงได้ระยะเวลานาน ป้องกันการเสียชีวิต อีกทั้งวัคซีนยังช่วยลดการเกิดภาวะ Long COVID ได้ (เจาะลึกระบบสุขภาพ, 2023)

ดังนั้นการแพร่ระบาดของโควิดจะต้องมีวัคซีนที่มีประสิทธิภาพซึ่งสามารถกระจายได้อย่างเท่าเทียมกัน วัคซีนเอ็มอาร์เอ็นเอเป็นหนึ่งในตัวเลือกของการทำวัคซีนที่เร็วที่สุดสำหรับโรคโควิด ข้อจำกัดที่เกิดขึ้นอย่างหนึ่งในตอนนี้คือการออกแบบโมเลกุลเอ็มอาร์เอ็นเอ ที่มีความเสถียรสูง วัคซีนทั่วไป (เช่นไข้หวัดใหญ่ตามฤดูกาล) วัคซีนทั่วไปสามารถบรรจุในกระบอกฉีดยาแบบใช้แล้วทิ้งและจัดส่งผ่านตู้เย็นทั่วโลก แต่ปัจจุบันวัคซีนเอ็มอาร์เอ็นเอยังไม่สามารถทำได้เนื่องจากไม่สามารถรักษาความเสถียรจากคุณสมบัติการย่อยสลายตามธรรมชาติของตัววัคซีนเอง



ภาพประกอบ 1 แผนผังการแพร่ระบาดครั้งใหญ่ของเชื้อไวรัสไข้หวัดใหญ่
และเชื้อ SARS-CoV-2

ที่มา : (รัตนโรจน์พงศ์, 2021)

ในการระบาดของโรคโควิด-19 วัคซีนเป็นส่วนสำคัญในการช่วยยับยั้งและป้องกันการระบาดได้มากขึ้น โดยดูจาก Neutralizing Antibody (ความสามารถในการสร้างภูมิ)และดูจากความสามารถในการป้องกันโรคที่เกิดขึ้น โดยพบว่าวัคซีน mRNA ได้แก่ Pfizer และ Moderna มีประสิทธิภาพเกิน 90% ,วัคซีน AstraZeneca ประสิทธิภาพประมาณ 80 %,วัคซีน Johnson&Johnson ประสิทธิภาพ 65-70% และ วัคซีน Sinovac ประสิทธิภาพประมาณ 50-70% (ช้างแก้ว)

สำหรับวัคซีน mRNA ผลิตด้วยเทคโนโลยีใหม่ล่าสุด ทำให้สามารถผลิตได้ในปริมาณมาก ประสิทธิภาพสูงและมีราคาค่อนข้างถูกแต่มีข้อเสียที่ต้องเก็บในอุณหภูมิต่ำกว่าวัคซีนทั่วไป (-20°C ขึ้นไป)

ชุมชน Eterna เป็นแพลตฟอร์มในการออกแบบข้อมูลของอาร์เอ็นเอโดยโซลูชันดังกล่าวได้รับการสังเคราะห์และทดสอบโดยนักวิจัยที่ Stanford เพื่อให้ได้ข้อมูลเชิงลึกใหม่ ๆ เกี่ยวกับโมเลกุลอาร์เอ็นเอการปรับปรุงความเสถียรของวัคซีนเอ็มอาร์เอ็นเอ โดยส่งมอบข้อมูลในทาง Kaggle ซึ่งเป็นแพลตฟอร์มสำหรับการแข่งขันการเรียนรู้โดยเครื่องจักรเพื่อให้ทุกคนได้ร่วมแข่งขันในการออกแบบแบบจำลองในการหาอัตราการย่อยสลายของวัคซีน การแก้ปัญหานี้ความท้าทายทางวิทยาศาสตร์ ภายในไม่กี่เดือนเพื่อเร่งการวิจัยวัคซีน เอ็มอาร์เอ็นเอ และส่งมอบวัคซีนป้องกัน

จากที่กล่าวมาในข้างต้นทางผู้วิจัยสนใจที่จะศึกษาการทำนายอัตราการย่อยสลายที่เป็นไปได้ที่แต่ละฐานของโมเลกุล RNA ซึ่งได้รับการฝึกฝนจากชุดข้อมูลย่อยของชุดข้อมูล Eterna

ซึ่งประกอบด้วยโมเลกุล RNA มากกว่า 3,000 โมเลกุลของโรค covid-19โดยใช้เทคนิคการเรียนรู้ของเครื่องจักรเพื่อเพิ่มโอกาสในการนำไปทดสอบเพื่อป้องกันโรคดังกล่าวต่อไป

วัตถุประสงค์

1. เพื่อศึกษาการสร้างแบบจำลองการทำนายค่าความเสถียรของวัคซีนเอ็มอาร์เอ็นเอสำหรับโรคโควิด 19 ด้วยวิธีการเรียนรู้ของเครื่องจักร
2. เพื่อเปรียบเทียบการทำนายความเสถียรของโมเดลการเรียนรู้เครื่องจักร
3. หาปัจจัยที่ส่งผลต่อความเสถียรของวัคซีน

ขอบเขตงานวิจัย

1. ศึกษาโดยใช้ข้อมูลของโครงสร้างอาร์เอ็นเอโดยโมเดลจะบอกว่ารูปร่างของอาร์เอ็นเอแบบใดที่จะมีความเสถียรซึ่งคืออัตราการย่อยสลายจากสภาวะภายนอก
2. ทำนายค่าความเสถียรของวัคซีนเอ็มอาร์เอ็นเอสำหรับโรคโควิด 19 มีทั้งหมด 5 เอาท์พุต คือ ค่า reactivity, ค่า deg_pH10 ค่า deg_mg_pH10,ค่า deg_50C และ ค่า deg_mg_50C ซึ่งเป็นอัตราการย่อยสลายในสภาวะต่างๆ โดยใช้หลักการของ mean columnwise root mean squared error (MCRMSE)

วิธีการดำเนินการวิจัย

1. ทำ Literature Review ที่เกี่ยวข้อง
2. หาข้อมูลเพื่อใช้ในงานวิจัย
3. เตรียมข้อมูล
4. ศึกษาวิธีการสร้างโมเดลในการทำนายข้อมูล
5. นำค่าวัดประสิทธิภาพของโมเดล
6. ประเมินและสรุปงานวิจัย

ประโยชน์ที่คาดว่าจะได้รับจากงานวิจัย

1. วิธีการเรียนรู้ของเครื่องจักรสามารถนำมาใช้ในการพัฒนาทางการแพทย์ได้
2. สามารถศึกษาการทำนายค่าความเสถียรของวัคซีนเอ็มอาร์เอ็นเอสำหรับไวรัสโควิด
3. สามารถนำข้อมูลที่ได้จากการศึกษาไปต่อยอดในงานวิจัยอื่นๆต่อไป

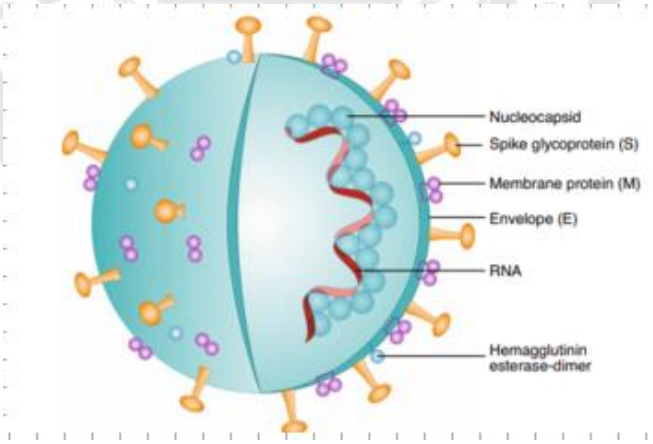
บทที่ 2
วรรณกรรมและงานวิจัยที่เกี่ยวข้อง

โรคโควิด 19 (Covid-19)

โรคโควิด 19 คือโรคเกิดจากไวรัสโคโรนาที่มีการค้นพบในเดือนธันวาคมปี 2019 โดยผู้ป่วยที่พบป่วยด้วยโรคปอดบวม ที่เมืองอู่ฮั่น ประเทศจีน ต่อมานักวิจัยชาวจีนพบว่าโรคติดเชื้อเชื้อไวรัสโคโรนา ชนิดใหม่ (novel coronavirus) และตั้งชื่อว่า “2019 novel Coronavirus” หรือเรียก อย่างย่อว่า “2019-nCoV “ โดยจะสามารถแพร่จากคนไปสู่คนได้โดยตรง (human-to-human transmission) และแพร่ระบาดไปทั่วโลกได้ โดยแพร่จากคนสู่คนผ่านทางฝอยละอองที่ออกจากจมูกหรือปากด้วยการ ไอหรือจาม โดยผู้ป่วยจะเริ่มมีอาการหลังจากติดเชื้อไม่เกิน 14 วัน

ไวรัสโคโรนา 2019

ไวรัสโคโรนาเป็นไวรัสในวงศ์ใหญ่มีหลากหลายสายพันธุ์ทำให้เกิดโรคระบบทางเดินหายใจ เช่น โรคทางเดินหายใจตะวันออกกลาง (MERS) และโรคระบบทางเดินหายใจเฉียบพลันร้ายแรง (SARS) และโรคติดเชื้อไวรัสโคโรนา 2019 หรือโควิด 19 ลักษณะโครงสร้างไวรัสโดยรวมจะเป็นทรงกลมมีหนามแหลม

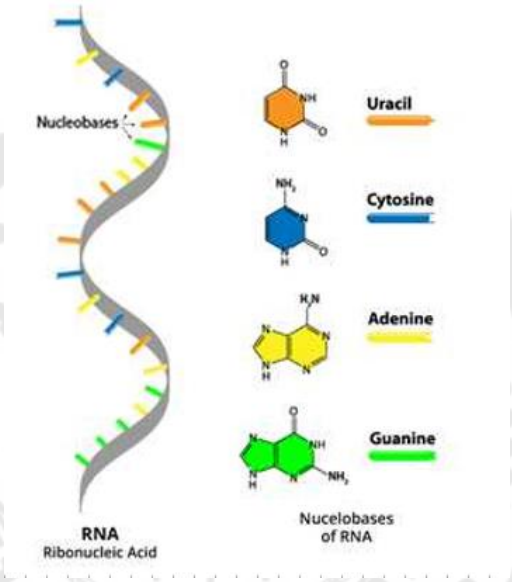


ภาพประกอบ 2 โครงสร้างไวรัสโคโรนา

ที่มา: (Florindo และคนอื่น ๆ, 2020)

อาร์เอ็นเอ (RNA)

อาร์เอ็นเอ (RNA) หรือ กรดไรโบนิวคลีอิก (Ribonucleic acid – RNA) เป็นสายพอลิเมอร์ของนิวคลีโอไทด์(Nucleotide) ไม่มีการแตกกิ่งก้านและมีความยาวสั้นกว่าโมเลกุลของ DNA มีส่วนประกอบด้วย น้ำตาลไรโบส (Ribose) และเบส 4 ชนิด คืออะดีนีน (Adenine, A) , ยูราซิล (Uracil, U) , ไซโตซีน (Cytosine, C) และกัวนีน (Guanine, G) และฟอสเฟต



ภาพประกอบ 3 อาร์เอ็นเอหรือกรดไรโบนิวคลีอิก

ที่มา : (Mackenzie, 2020)

โดยอาร์เอ็นเอ (RNA) จะเป็นโพลีนิวคลีโอไทด์สายเดี่ยว(single strand) ซึ่งนิวคลีโอไทด์ (Nucleotide) เชื่อมด้วยพันธะฟอสโฟไดเอสเทอร์ (phosphodiester bond) อาร์เอ็นเอ (RNA) จะมีการถอดรหัส (transcription) จากดีเอ็นเอ (DNA) ด้วย RNA Polymerase ซึ่ง ตัวอาร์เอ็นเอ(RNA) เหมือนเป็นแม่แบบ(Template) ในการแปลข้อมูลจากยีนไปเป็นโปรตีน แล้วขนย้ายกรดอะมิโนเข้าไปใน ribosome ของเซลล์ เพื่อผลิตโปรตีน และแปลรหัส (translation) เป็นข้อมูลในโปรตีน

อาร์เอ็นเอ (RNA)มีทั้งหมด 3 ชนิด คือ เอ็มอาร์เอ็นเอ หรือ เมสเซนเจอร์ อาร์เอ็นเอ (messenger RNA, mRNA), ,อาร์อาร์เอ็นเอ หรือ ไรโบโซมอล อาร์เอ็นเอ (ribosomal RNA, rRNA) และทีอาร์เอ็นเอ หรือ ทรานสเฟอร์ อาร์เอ็นเอ (transfer RNA, tRNA)

เอ็มอาร์เอ็นเอ (mRNA)

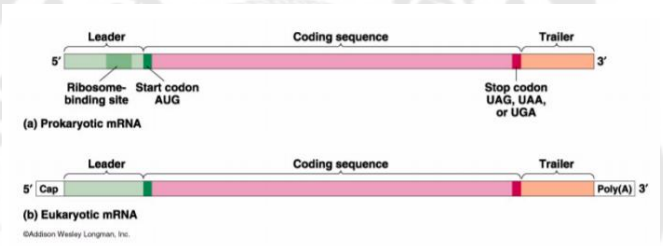
เอ็มอาร์เอ็นเอ (mRNA) หรือ เมสเซนเจอร์ อาร์เอ็นเอ (messenger RNA) เป็น อาร์เอ็นเอ (RNA) ชนิดหนึ่ง โดยมีหน้าที่การเรียงลำดับของนิวคลีโอไทด์ที่ Complement กับข้อมูลบริเวณจำเพาะของ DNA ซึ่ง mRNA ทำหน้าที่เป็น template ในการสร้างโปรตีน

โครงสร้างของเอ็มอาร์เอ็นเอ ประกอบด้วย 3 ส่วน

1. Leader sequence หรือ 5 untranslated leader sequence (5/ULR) เป็นลำดับเบสจำเพาะที่อยู่บริเวณปลาย 5/ P ของ mRNA เกาะกับสาย mRNA เพื่อเริ่มกระบวนการแปรรหัส

2. Coding sequence บริเวณกำหนดชนิดของกรดอะมิโน ในกระบวนการแปรรหัส

3. tailer sequence หรือ 3 untranslated tailer sequence (3/ UTR) คือ ตำแหน่งปลาย 3



ภาพประกอบ 4 แสดงโครงสร้างโมเลกุล mRNA ที่สมบูรณ์

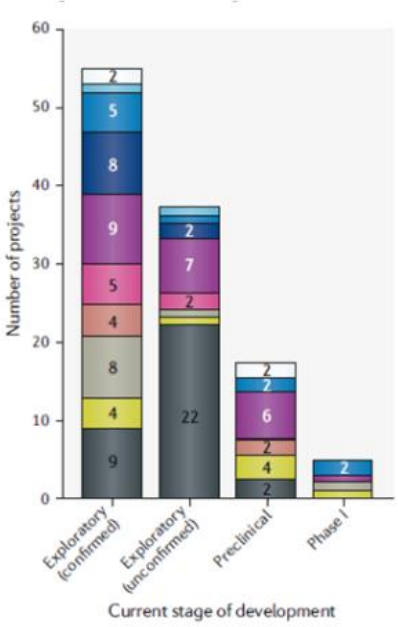
ที่มา : ("โมโลกุลอาร์เอ็นเอ (RNA Molecules) ")

วัคซีน (vaccine)

วัคซีน คือสารชนิดหนึ่งที่ฉีดเข้าไปร่างกาย เพื่อสร้างภูมิคุ้มกันโรคต่างๆ ทำมาจากเชื้อโรค แบ่งเป็น 2 ประเภท คือ ทำจากเชื้อโรคที่ตาย และ ทำจากเชื้อโรค เมื่อฉีดเข้าไปในร่างกายก็จะสร้างภูมิคุ้มกัน กับโรคนั้นๆ(มหาวิทยาลัยมหิดล, 2020)

การพัฒนาวัคซีน COVID-19

ข้อมูลด้านพันธุกรรมของไวรัส SARS-CoV-2 ได้รับการเผยแพร่ในวันที่ 11 มกราคม 2020 ซึ่งส่งผล ให้เกิดการเริ่มต้นพัฒนาวัคซีนในหลายๆที่ทั่วโลก โดยมีรายงานในเดือนเมษายน 2020 ว่า ที่อยู่ในกระบวนการพัฒนาโดยที่วัคซีน 78 ชนิดมีการระบุอย่างชัดเจนในการพัฒนา (รัตนโรจน์ พงศ์, 2021)โดยมีรายละเอียดชนิดของวัคซีนที่พัฒนาดังแสดงในรูปที่ 5



ภาพประกอบ 5 ชนิดของวัคซีน COVID-19 ที่มีการพัฒนา

ที่มา: (รัตนโรจน์พงศ์, 2021)

สำหรับการพัฒนาวัคซีนในประเทศไทยนั้นมีทั้งหมด 7 เทคโนโลยีการผลิต ได้แก่

1. mRNA Vaccine ได้มาจากการสังเคราะห์เอ็มอาร์เอ็นเอ ที่เฉพาะเจาะจงกับเชื้อไวรัส วัคซีนจะทำหน้าที่พาเอ็มอาร์เอ็นเอเข้าเซลล์ และทำให้เซลล์ผลิตสารโปรตีนสไปค์ของเชื้อไวรัสโดย โปรตีนนี้จะกระตุ้นระบบภูมิคุ้มกันของร่างกาย

2. DNA Vaccine การนำสารพันธุกรรมดีเอ็นเอฉีดเข้ากล้ามเนื้อ สามารถที่จะทำให้เกิดการแสดงออกของยีนได้ ดีเอ็นเอวัคซีนประกอบด้วยยีน ที่ต้องการซึ่งสามารถสร้างโปรตีนหรือแอนติเจนที่ได้รับการ ดัดแปลงอย่างเหมาะสมภายใต้การควบคุมของส่วนที่เป็น regulatory sequences โปรตีนที่สร้างจากดีเอ็นเอวัคซีนจะ อยู่ในลักษณะ native conformation

3. Viral-like particle (VLP) วัคซีนอนุภาค ไวรัสเสมือนเป็นเทคโนโลยีการสร้างโครงสร้างเลียนแบบอนุภาคไวรัส แต่ไม่มีสารพันธุกรรมของไวรัส

4. Protein Sub Unit Vaccine วัคซีนที่ผลิตโดยการสร้างโปรตีนของเชื้อไวรัสด้วยระบบ cell culture, yeast เป็นต้น ใช้ในการผลิตวัคซีนหลายชนิด เช่น วัคซีนป้องกันไข้หวัดใหญ่ วัคซีนป้องกันไวรัสตับอักเสบบี เป็นต้น (ศักดิ์ทองจีน, 2021)

5. Viral vector Veccine สร้างจากไวรัสที่ถูกทำให้อ่อนลงหรือไม่สามารถแบ่งตัวได้อีก ตัดแต่งพันธุกรรมเพื่อใช้เป็นพาหะ สามารถกระตุ้นภูมิคุ้มกันได้ดี เนื่องจากเลียนแบบการติดเชื้อที่ใกล้เคียงกับการติดเชื้อตามธรรมชาติ (โรงพยาบาลศิครินทร์)

6. Inactivated Vaccine วัคซีนกลุ่มนี้ผลิตโดยนำไวรัสมาเลี้ยงขยายจำนวนมาก และทำให้เชื้อตาย

7. Live attenuated Vaccine วัคซีนที่ผลิตขึ้นโดยใช้เชื้อโรคมาทำให้อ่อนฤทธิ์ลงจนไม่สามารถทำให้เกิดโรคแต่เพียงพอที่จะกระตุ้นภูมิคุ้มกันของร่างกายได้

โดย หน่วยงานที่เกี่ยวข้องทั้งหมด 9 หน่วยงาน ทั้งภาครัฐ รัฐวิสาหกิจ เอกชน และมหาวิทยาลัย โดยการวิจัยที่มี ความก้าวหน้ามากที่สุด และอยู่ระหว่างเตรียมเข้าสู่การทดสอบในมนุษย์ระยะที่ 1 นั้นมี 3 เทคโนโลยีการผลิต ได้แก่

1. วัคซีนชนิดเอ็มอาร์เอ็นเอซึ่งพัฒนาโดยศูนย์เชี่ยวชาญเฉพาะทางการวิจัยและพัฒนาวัคซีน คณะแพทยศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
2. วัคซีนชนิด DNA โดยบริษัท ไบโอเนท-เอเชีย จำกัด
3. วัคซีนที่ใช้เทคโนโลยีการ สกัดโปรตีนจากพืช (Plant based) โดยคณะเภสัชศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย และบริษัท ใบยาไฟโต ฟาร์ม จำกัด

การวิจัยการพัฒนาวัคซีน COVID-19 ในอดีตการพัฒนาวัคซีนป้องกันโรคติดเชื้อในกลุ่ม coronavirus เช่น SARS-CoV ไม่ค่อยสัมฤทธิ์ ผลเนื่องจากวัคซีนยังไม่มีประสิทธิภาพที่ดีพอในการป้องกันโรค ถึงแม้ว่าวัคซีนที่พัฒนาขึ้นมามีความสามารถในการกระตุ้นการตอบสนองของระบบภูมิคุ้มกันในสัตว์ทดลองและสิ่งที่กังวลคือวัคซีนอาจไม่นำไปสู่การป้องกันโรคได้ในระยะยาว และส่งผลก่อให้เกิดการติดเชื้อไวรัสซ้ำได้

ดังนั้นการพัฒนาวัคซีน COVID-19 จึงต้องคำนึงถึงความปลอดภัยของวัคซีนในการที่จะไม่ก่อให้เกิดพยาธิ สภาพจากการตอบสนองของระบบภูมิคุ้มกัน การป้องกันโรคได้อย่างยาวนาน และไม่ก่อให้เกิดการติดเชื้อซ้ำใน ผู้ที่ได้รับวัคซีน โดยศึกษาการพัฒนาวัคซีนในรูปแบบต่างๆที่สามารถนำไปสู่การผลิตวัคซีนที่มีประสิทธิภาพเพื่อ ลดข้อจำกัดของวัคซีนที่เคยมีงานวิจัยมาก่อนหน้านี้ในการพัฒนาวัคซีนต่อ SARS-CoV และ MERS-CoV ความปลอดภัยของผู้ที่ได้รับวัคซีนเป็นสิ่งที่สำคัญที่สุดในการพัฒนาวัคซีนเพื่อป้องกันโรคต่างๆ รวมทั้งวัคซีนป้องกันโรค COVID-19 เนื่องจากมีรายงาน การพัฒนาวัคซีนป้องกันไวรัส SARS-CoV และ MERS-Co-V ที่พบว่า วัคซีนก่อให้เกิดพยาธิสภาพที่ไม่พึงประสงค์โดยไปก่อให้เกิดพยาธิในปอด

อย่างไรก็ตามยังไม่มีรายงานว่า วัคซีนต่อ COVID-19 ที่พัฒนาและกำลังดำเนินการทดสอบทางคลินิกอยู่นั้นบางชนิดได้ผ่านการรับรองเพื่อนำมาใช้ในการป้องกันการระบาดของโรค สามารถส่งผลทำให้เกิดการตอบสนองของระบบภูมิคุ้มกันที่มี ผลกระทบต่อผู้ได้รับวัคซีนหรือไม่ อย่างไรในประชากรทั้งหมดที่ได้รับวัคซีน โดยเฉพาะผู้สูงอายุและผู้ที่มีโรค แทรกซ้อนเช่น โรคเบาหวาน ความดันสูงหรือโรคหัวใจที่มีความเสี่ยงที่จะมีอาการรุนแรงถ้าติดเชื้อไวรัส SARSCoV-2 ซึ่งการทดสอบทาง preclinic ในกลุ่มผู้สูงอายุหรือการใช้สัตว์ทดลองที่มีอายุจึงไม่ควรมองข้ามเพื่อยืนยันว่า วัคซีนที่พัฒนามีความปลอดภัยในกลุ่มเสี่ยงนี้ รวมทั้งองค์ความรู้ในการตอบสนอง ของระบบภูมิคุ้มกันในกลุ่มนี้ต่อวัคซีนแต่ละประเภทที่พัฒนาขึ้นเพื่อให้สามารถพัฒนาวัคซีนให้มีความปลอดภัย และสามารถกระตุ้นระบบภูมิคุ้มกันที่นำไปสู่การป้องกันโรคของคนกลุ่มนี้ได้อย่างมีประสิทธิภาพ ในปัจจุบันได้มีการนำเอาเทคโนโลยีใหม่ๆมาใช้ในการพัฒนาวัคซีนในรูปแบบต่างๆ

จากการร่วมมือกันหน่วยงานต่างๆระหว่างประเทศเพื่อให้ได้วัคซีนที่มีประสิทธิภาพดีที่สุดและผลิตได้เร็วที่สุดเพื่อ นำมาใช้ในการป้องกันการแพร่ระบาดของโรค COVID-19 ที่ยังเกิดขึ้นอย่างต่อเนื่องและไม่มีท่าทีว่าการระบาดจะเริ่มชลอตัวหรือลดน้อยลงโดยวัคซีนที่พัฒนาขึ้นจากเทคโนโลยีใหม่ๆนี้ เช่น nucleic acid vaccine (อาจ เป็นชนิด DNA หรือ RNA), viral vector vaccine (ใช้เชื้อไวรัสชนิดอื่นๆให้มีการแสดงออกของโปรตีนจากไวรัส SARS-CoV-2), Inactivated virus vaccine, attenuated virus vaccine, virus like particle vaccine 85 (VLP) และ protein subunit/recombinant protein subunit vaccine (ใช้เทคนิคทางด้านพันธุวิศวกรรมในการผลิตโปรตีนของไวรัสที่เป็นเป้าหมายในการกระตุ้นการทำงานของระบบภูมิคุ้มกัน)

วัคซีนเอ็มอาร์เอ็นเอ

วัคซีนเอ็มอาร์เอ็นเอเป็นพื้นฐานทางเทคโนโลยีของการบำบัดรักษาและวัคซีนนั้นมีความยืดหยุ่นสูงในด้านการผลิตและการใช้งานโปรตีนใด ๆ สามารถเข้ารหัสและแสดงโดยเอ็มอาร์เอเป็นทางเลือกที่ดีสำหรับวิธีการฉีดวัคซีนแบบเดิมเนื่องจากมีศักยภาพสูงความสามารถในการพัฒนาอย่างรวดเร็วและศักยภาพในการผลิตต้นทุนต่ำ(Schlake, Thess, Fotin-Mleczek, และ Kallen, 2012) วัคซีนเอ็มอาร์เอ็นเอได้กระตุ้นภูมิคุ้มกันที่มีศักยภาพต่อเป้าหมายของโรคติดเชื้อ เช่น ไวรัสไข้หวัดใหญ่ หรือมะเร็ง โดยเฉพาะอย่างยิ่งในช่วงไม่กี่ปีที่ผ่านมาโดยใช้เอ็มอาร์เอ็นเอ ที่ได้รับการปรับแต่งตามลำดับด้วยไขมันหรือรูปแบบเปลือยเปล่า

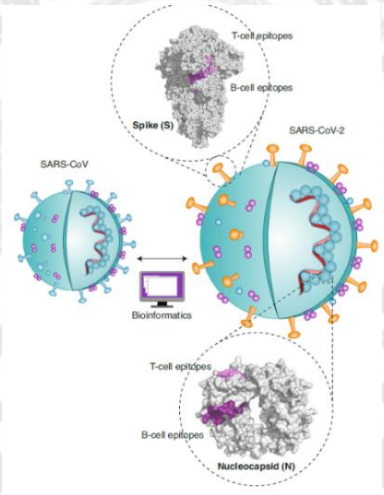
แนวทางที่หลากหลายในการฉีดวัคซีนเอ็มอาร์เอ็นเพื่อยับยั้งมะเร็งรวมถึงวัคซีนเซลล์เดนไดรติกและ วัคซีนเอ็มอาร์เอ็นเอชนิดฉีดโดยตรงหลายชนิดได้ถูกนำไปใช้ในการทดลองทางคลินิก

มะเร็งจำนวนมากโดยมีผลการทดลองบางอย่างที่แสดงให้เห็นถึงการตอบสนองของ T ของแอนติเจนที่จำเพาะต่อแอนติเจนและการอยู่รอดโดยปราศจากโรคเป็นเวลานานในบางกรณี

โดยในขณะนี้การผลิตวัคซีนเอ็มอาร์เอ็นเอสำหรับโรคโควิด 19 มีบริษัทผู้ผลิตยาโมเดอร์นาของสหรัฐอเมริกาเปิดเผยผลเบื้องต้นจากการทดลองทางคลินิกกับผู้ทดลองมากกว่า 30,000 คน ได้ผลในการป้องกันไวรัสถึง 94.5 % และคงสภาพได้เมื่อเก็บในอุณหภูมิระหว่าง 2-8 องศาเซลเซียส (ไทยโพสต์, 2020)

การทำงานของวัคซีนเอ็มอาร์เอ็นเอ

การทำงานของวัคซีนเอ็มอาร์เอ็นเอไปกระตุ้นให้ร่างกายสร้างโปรตีนที่มีรูปทรงแหลมคล้าย "หนามแหลม" (Spike) บนไวรัส SARS-CoV-2 เมื่อเราฉีดวัคซีนเข้าสู่ร่างกายจะทำให้ระบบภูมิคุ้มกันในร่างกายของคนเราเรียนรู้รูปร่างของไวรัสชนิดนี้ไว้ล่วงหน้าก่อนที่จะโดนไวรัสตัวจริงเข้าสู่ร่างกาย (Florindo และคนอื่น ๆ, 2020)



ภาพประกอบ 6 การทำงานของวัคซีนเอ็มอาร์เอ็นเอต่อไวรัส

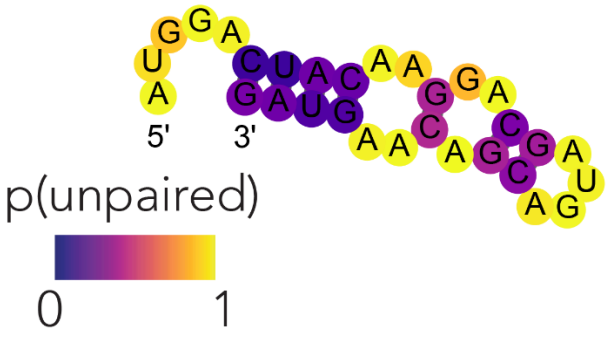
ที่มา: (Florindo และคนอื่น ๆ, 2020)

การออกแบบวัคซีนเอ็มอาร์เอ็นเอที่มีความเสถียร

จากบทความของชุมชน Eterna ที่อธิบายเกี่ยวกับวัคซีนอาร์เอ็นเอ โดย RNA จะสร้างคู่เบสเช่นเดียวกับ DNA แทนที่จะเป็นคู่เบส G-C และคู่เบส A-T ใน DNA RNA จะเป็นคู่เบส G-C และ A-U

RNA แบบจับคู่ฐานมีแนวโน้มที่จะย่อยสลายน้อยกว่า RNA แบบเกลียวเดี่ยว โดยแสดงตัวอย่างตาม รูปที่ 6 เอ็มอาร์เอ็นเอ ของโปรตีนขนาดเล็กโดยแต่ละฐานจะมีสีตามความน่าจะเป็น

ที่ไม่ได้จับคู่ เราจะเห็นว่าโมเลกุลมี "จุดร้อน" เป็นสีเหลืองซึ่งมีแนวโน้มที่จะย่อยสลายได้มากกว่า แต่ยังมีความคลาดเคลื่อนจากการทดลองจริงอยู่อีกพอสมควร (Eternagame, 2020)



ภาพประกอบ 7 mRNA of a mini protein, with bases colored by the probability they are unpaired.

ที่มา : (Eternagame, 2020)

จากการอ้างอิงงานวิจัยของ Wadhwa และคณะ (Wadhwa A, 2020)อ้างว่าภายใต้เงื่อนไขการขนส่งแบบ cool chain ในตู้เย็น ครึ่งชีวิตของวัคซีน mRNA อาจมีครึ่งชีวิต 900 วัน โดยมีอัตราการย่อยสลายอย่างน้อย 2% ทุกๆ 30 วัน ยิ่งกว่านั้นที่อุณหภูมิประมาณ 37°เชื่อกันว่าจะลดครึ่งชีวิตของ วัคซีนถึง 5 วันและ 10 วันตามลำดับ นอกเหนือไปจากนี้,เป็นที่น่าสังเกตว่าเมื่อมี Mg2+ ที่อุณหภูมิ 37°C ซึ่งเป็นสภาวะตามปกติสำหรับการทดสอบในหลอดทดลอง ครึ่งชีวิตของวัคซีนจะลดลงเหลือไม่เกิน 2 ชั่วโมงและผลลัพธ์ยังคงเหมือนเดิมแม้หลังจากลดความเข้มข้นของ Mg2+, ค่า pH หรืออุณหภูมิเพื่อลดการไฮโดรไลซิสซึ่ง วัคซีนเอ็มอาร์เอ็นเอจะไม่เสถียรเนื่องจากการย่อยสลายยังคงเกิดขึ้นในระหว่างกระบวนการลอกรหัส (transcription) (S. H. Ing, 2021)

ปัญหาเกี่ยวกับวัคซีนเอ็มอาร์เอ็นเอ

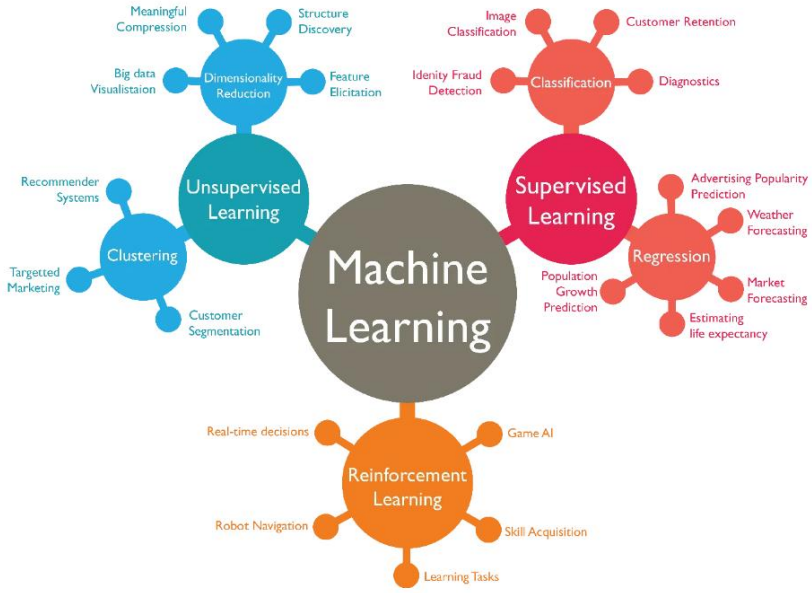
จากการออกแบบโมเลกุลอาร์เอ็นเอของสารที่มีความเสถียรสูง โมเลกุลของอาร์เอ็นเอมีแนวโน้มที่จะย่อยสลายเองตามธรรมชาติเป็นข้อจำกัด วัคซีนเอ็มอาร์เอ็นเอในปัจจุบันสำหรับป้องกันโรคโควิด 19 จะต้องได้รับการเตรียมและจัดส่งภายใต้การแช่เย็นที่เข้มข้น วัคซีนเอ็มอาร์เอ็นเอเผชิญกับความท้าทายเนื่องจากความไม่เสถียรทางความร้อน ซึ่งทำให้ไวต่อการย่อยสลายทางเคมี ด้วยเหตุนี้ วัคซีน เอ็มอาร์เอ็นเอ จึงต้องมีเงื่อนไขที่เข้มงวดในการผลิต การจัดเก็บ และการ

จัดส่งทั่วโลก ในการทำให้วัคซีน เอ็มอาร์เอ็นเอ สามารถเข้าถึงได้ในวงกว้างมากขึ้น จำเป็นอย่างยิ่งที่จะต้องทำความเข้าใจและปรับปรุงความเสถียรของวัคซีน

นอกจากเหตุผลทางด้านอุณหภูมิเอ็นอาร์เอ็นเอมีความไวสูงต่อเอนไซม์ RNase ซึ่งย่อยสลายได้ง่าย ไวต่อปฏิกิริยาออกซิเดชั่นมากกว่า ไวต่อการไฮโดรไลซิสที่ค่า pH สูงกว่า 6 สำหรับวัคซีน RNA เป็นสิ่งสำคัญในการป้องกันความสมบูรณ์ของโมเลกุลที่สมบูรณ์เนื่องจากความผิดปกติเพียงครั้งเดียวสามารถสลายได้ (Uddin MN, 2021)

การเรียนรู้ของเครื่องจักร (Machine Learning)

การเรียนรู้ของเครื่องจักร (Machine Learning) คือระบบที่สามารถเรียนรู้ได้จากตัวอย่างด้วยตนเองโดยไม่จำเป็นต้องมีคำสั่งของโปรแกรมเมอร์ สามารถเรียนรู้เพียงแค่จากข้อมูลอย่างเดียวแบ่งออกเป็น 3 ประเภท คือ Supervised Learning, Unsupervised Learning, Reinforcement Learning

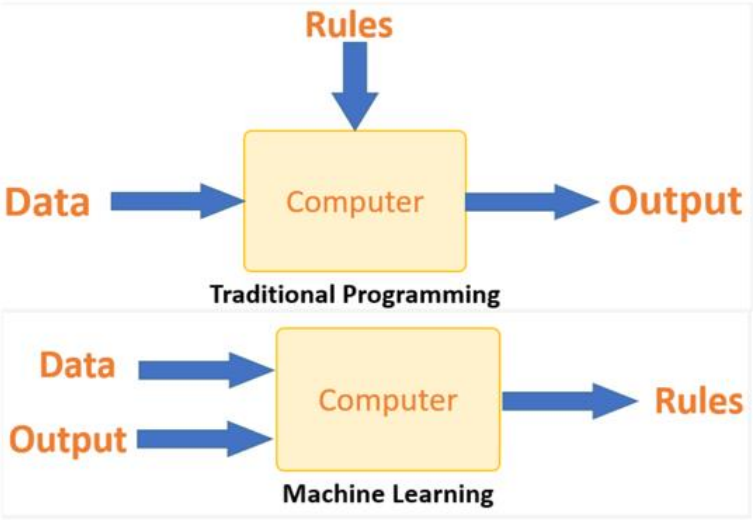


ภาพประกอบ 8 ประเภทของการเรียนรู้โดยเครื่องจักร

ที่มา : https://medium.com/investic/machine-learning

ความแตกต่างระหว่าง Machine Learning และการเขียนโปรแกรมดั้งเดิม

การเขียนโปรแกรมในสมัยก่อนนั้นโค้ดทั้งหมดจะต้องถูกกำหนดแนวทางไว้ชัดเจน ซึ่งแต่ละกฎจะขึ้นอยู่กับพื้นฐานความเข้าใจด้านตรรกศาสตร์ (Logic Foundation) ทำให้ผลลัพธ์ออกมาตามคำสั่ง เมื่อระบบเริ่มซับซ้อนมากขึ้นกฎจะมากขึ้นตาม ส่วนเครื่อง(machine)จะทำการเรียนรู้ว่าข้อมูลขาเข้าและข้อมูลขาออกเกี่ยวข้องกับอย่างไร และจะเขียนกฎขึ้นมา โดยโปรแกรมเมอร์ไม่จำเป็นต้องเขียนกฎใหม่ทุกครั้งที่ อัลกอริทึมจะปรับเข้ากับข้อมูลใหม่เอง



ภาพประกอบ 9 ความแตกต่างระหว่าง Machine Learning และการเขียนโปรแกรมดั้งเดิม

ประเภทของการเรียนรู้โดยเครื่องจักร

1.การเรียนรู้แบบมีผู้สอน (Supervised Learning)

จะอยู่ในลักษณะการทำนายผลลัพธ์ จะมีชุดข้อมูลฝึกฝน (Training Data) เป็นตัวฝึก โดยมนุษย์มาคอยแยกประเภทหรือบอกผลลัพธ์ (Label)ไว้ โดยนำข้อมูลที่ใช้ฝึกไปผ่านอัลกอริทึมสำหรับสร้างแบบจำลอง จากนั้นนำข้อมูลที่เครื่องไม่เคยเห็นให้เครื่องทำนาย (predict)

ประเภทของ supervised learning อยู่ 2 ประเภท

- การแบ่งแยกประเภท (Classification)
- การถดถอย (Regression)

การแบ่งแยกประเภท (Classification)

เป็นวิธีที่จะต้องมี Target ไว้สำหรับให้ตัว Model เรียนรู้จาก Input Data เพื่อหาคำตอบออกมาตามTarge tที่เราได้วางเอาไว้ ซึ่งผลลัพธ์จากการวิเคราะห์ข้อมูลด้วยClassification Model จะเป็นในรูปแบบของการจำแนกข้อมูลเพื่อให้ได้คำตอบที่เป็นตัวเลือก หรือกลุ่มข้อมูล ตัวอย่างของ Target ของโมเดล

การถดถอย (Regression)

เป็นการศึกษาความสัมพันธ์ระหว่างลักษณะตัวแปรตั้งแต่สองตัวแปรขึ้นไปโดยหาสมการความสัมพันธ์ระหว่างตัวแปร 2ตัวเพื่อที่จะนำไปสู่การคาดการณ์หรือประมาณค่าการวิเคราะห์พยากรณ์เหตุการณ์ในอนาคต เช่น การทำนายมูลค่าของหุ้น เป็นต้น

2.การเรียนรู้แบบไม่มีผู้สอน (Unsupervised Learning)

เป็นการให้เครื่องตามหาโครงสร้างข้อมูลที่เราไม่รู้จัก (Unknown Structure) โดยอัลกอริทึมจะตรวจสอบเฉพาะข้อมูลที่ป้อนเข้ามาเท่านั้นโดยปราศจากการให้ผลลัพธ์ที่จะเกิดขึ้น (เช่น การสำรวจข้อมูลประชากรเพื่อหาแบบแผน(pattern)ของข้อมูลนั้น)

3. **การเรียนรู้แบบเสริมแรง** (Reinforcement Learning)

การทำ Action ตามลำดับต่างๆ ออกมาแล้วคะแนนดี เราก็จะได้รางวัล (Reward) เครื่องจะเริ่มจดจำลำดับการทำ Action แล้วได้คะแนนดี และพยายามจะทำ Action นั้นเรื่อยๆ

ทฤษฎีของแบบจำลองที่นำมาใช้ในงานวิจัย

XGBoost (eXtreme Gradient Boosting)

XGBoost เป็นแบบจำลองที่นำเอาต้นไม้ตัดสินใจมาฝึกสอนต่อกันหลาย ๆ ต้น โดยที่ต้นไม้ตัดสินใจแต่ละต้นจะเรียนรู้จากค่าความผิดพลาดของต้นก่อนหน้าต่อเนื่องกันจนมีความลึกมากพอ แบบจำลองจะหยุดเรียนรู้เมื่อไม่เหลือค่าความผิดพลาดจากต้นไม้ตัดสินใจต้นก่อนหน้าให้เรียนรู้แล้ว

Random Forest

เป็นแบบจำลองที่นำ Decision Tree หลายๆ tree มา Train ร่วมกัน (ตั้งแต่ 10 ต้น ถึงมากกว่า 1000 ต้น) โดยที่แต่ละ tree จะได้รับ feature และ data เป็น subset ของ feature และ data ทั้งหมด แบบสุ่มตอนคาดการณ์ แต่ละ Decision Tree ทำการคาดการณ์ของตัวมันเอง และเลือกผล final prediction

Light GBM (Light Gradient Boosting)

Light GBM เป็น Gradient Boosting ใช้อัลกอริทึม Leaf-wise ซึ่งสามารถลดการสูญเสียได้ดีกว่าอัลกอริทึมแบบ Level-wise ให้ผลลัพธ์ที่แม่นยำและมีความเร็วมากกว่า

CatBoost (Gradient boosting with categorical features support)

เป็นอัลกอริทึมการเรียนรู้ของเครื่องจากที่เปิดให้ใช้งานแบบโอเพนซอร์ส CatBoost สามารถใช้งานร่วมกับเฟรมเวิร์กการเรียนรู้เชิงลึก เช่น Google's TensorFlow ได้ ซึ่ง โดยสามารถจัดการกับตัวแปรโดยอัตโนมัติ และไม่ต้องการการแปลงชุดข้อมูลให้เป็นรูปแบบเฉพาะ นอกจากนี้ CatBoost ยังสามารถจัดการกับตัวแปรและค่าข้อมูลบางส่วนที่หายไป (Missing Values) ได้อย่างมีประสิทธิภาพอีกด้วย

Cross validation

เป็นเทคนิคในการแบ่งข้อมูลเป็น k ส่วนเท่าๆกัน เพื่อสร้างและทดสอบแบบจำลอง (train + validate) คำนวณค่าเฉลี่ย accuracy หรือ error(i.e. model performance) ใช้เปรียบเทียบว่าข้อมูลชุดไหนดีที่สุดก่อนที่จะนำไปใช้ทำนายข้อมูลทดสอบ (test set) หรือ การเรียนรู้ (train) และทดสอบ (test) แบบจำลองในแต่ละรอบ เราสามารถทดสอบ hyper parameters หลายๆค่าพร้อม

กันได้ เพื่อหาค่าที่ดีที่สุดสำหรับแบบจำลองของเรา ใช้เปรียบเทียบระหว่างแบบจำลองได้ว่าแบบจำลองไหนดีกว่ากัน

งานวิจัยที่เกี่ยวข้อง

1. Elisabeth J. Wurtmann, Sandra L. Wolin (2009) ได้ศึกษาความเสียหายต่ออาเอ็นเอเกิดจากแสงอัลตราไวโอเลต, การออกซิเดชั่น,คลอรีน,ไนเตรตและอะคิเลชันอาจรวมถึงการดัดแปลงทางเคมีของนิวคลีโอเบสเช่นเดียวกับการเชื่อมขวางอาร์เอ็นเอ-อาร์เอ็นเอและ อาร์เอ็นเอ-โปรตีน การศึกษาในหลอดทดลองได้อธิบายถึงปัจจัยที่สร้างความเสียหายได้หลายประเภทซึ่งบางส่วนได้รับการสนับสนุนว่าเกี่ยวข้องกับสรีรวิทยาโดยการสังเกตในร่างกายในการเจริญเติบโตปกติสภาวะความเครียดหรือสถานะของโรค ความเสียหายต่อทั้งเอ็มอาร์เอ็นเอและ อาร์เอ็นเอที่ไม่มีการเข้ารหัสอาจมีผลในการทำงานและงานได้เริ่มอธิบายถึงบทบาทของเส้นทางการหมุนเวียนของอาร์เอ็นเอและเส้นทางการรับรู้ความเสียหายเฉพาะในการล้างเซลล์ของอาร์เอ็นเอเหล่านี้ โดยงานวิจัยนี้ช่วยบอกปัจจัยที่มีผลต่อการสลายตัวของอาร์เอ็นเอ (Wurtmann และ Wolin, 2009)
2. Norbert Pardi และคณะ (2018) ได้ศึกษาว่าวัคซีนเอ็มอาร์เอ็นเอเป็นทางเลือกที่ดีสำหรับวิธีการฉีดวัคซีนทั่วไป เนื่องจากมีศักยภาพสูงความสามารถในการพัฒนาอย่างรวดเร็วและศักยภาพในการใช้ต้นทุนต่ำ การผลิตและการบริหารที่ปลอดภัย แต่ถูกจำกัด โดยความไม่เสถียรและการส่งมอบเอ็มอาร์เอ็นเอที่ไม่มีประสิทธิภาพวัคซีนเอ็มอาร์เอ็นเอหลายตัวต่อต้านโรคติดเชื้อและมะเร็งหลายชนิดบทวิจารณ์นี้ให้ภาพรวมโดยละเอียดของวัคซีนเอ็มอาร์เอ็นเอและพิจารณาทิศทางและความท้าทายในอนาคตในการพัฒนาแพลตฟอร์มวัคซีนใช้ในการรักษาอย่างแพร่หลาย (Pardi, Hogan, Porter, และ Weissman, 2018)
3. Edison Ong และคณะ (2020) พัฒนาวัคซีนที่มีประสิทธิภาพและปลอดภัย 15 วัคซีนสำหรับโรคติดต่อที่เกิดจากโคโรนาไวรัส ได้ใช้เครื่องมือ Vaxign reverse Vaccinology 19 และเครื่องมือการเรียนรู้ของเครื่อง Vaxign-ML ที่พัฒนาขึ้นใหม่เพื่อทำนายผู้ป่วยที่ได้รับวัคซีน COVID-19 อัลกอริธึมการจำแนก ML ได้แก่ การถดถอยโลจิสติก, ซัพพอร์ตเวกเตอร์แมชชีน, วิธีการเพื่อนบ้านใกล้ที่สุด, แรนด้อมฟอเรส และ xgboost ประสิทธิภาพที่ดีที่สุดคือ xgboost (Ong, Wong, Huffman, และ He, 2020)
4. Brett N. Bowman และคณะ (2011) ได้ศึกษาหาการออกแบบวัคซีนหน่วยย่อยโดยการคาดการณ์อย่างรวดเร็วว่าโปรตีนในโปรตีโอมของแบคทีเรียชนิดใดเป็นแอนติเจนที่ป้องกันได้โดยใช้วิธีเครื่องจักรเวกเตอร์สนับสนุน (Bowman และคนอื่น ๆ, 2011)

5. Mohammad N. Uddin และคณะพยายามหากลยุทธ์เพื่อเพิ่มความคงตัวของวัคซีนที่ใช้ mRNA ในอุณหภูมิที่ค่อนข้างสูงขึ้น เนื่องจากการใช้ทั่วโลกถูกจำกัดโดย ultracold ข้อกำหนดในการจัดเก็บ ดังนั้นประเทศที่ยากจนด้านทรัพยากรส่วนใหญ่ไม่มีที่เก็บห่วงโซ่ความเย็นเพื่อดำเนินการเก็บวัคซีนจำนวนมากดังนั้น การหากลยุทธ์เพื่อเพิ่มความคงตัวของวัคซีนที่ใช้เอ็มอาร์เอ็นเอในอุณหภูมิที่ค่อนข้างสูงขึ้นสามารถช่วยในการจัดการป้องกับโรคระบาดทั่วโลกในปัจจุบัน (Uddin MN, 2021)
6. Hannah K. Wayment-Steele และ คณะวัดความละเอียดของนิวคลีโอไทด์เดี่ยวบนโครงสร้าง RNA ที่หลากหลายของนิวคลีโอไทด์ 102-130 นิวคลีโอไทด์ โดยใช้ชุดข้อมูลของ Eterna การทดลองทั้งหมดเสร็จสิ้นภายในเวลาไม่ถึง 6 เดือน และ 41% ของการคาดการณ์ระดับนิวคลีโอไทด์จากแบบจำลองที่ชนะนั้นเกิดจากข้อผิดพลาดในการทดลองของการวัดค่าความจริงพื้นฐาน ยิ่งไปกว่านั้น แบบจำลองเหล่านี้ทำให้คาดการณ์ข้อมูลการสลายตัวแบบมุมฉากแบบสุ่มสี่สุ่มห้าบนโมเลกุล mRNA ที่ยาวกว่ามาก (นิวคลีโอไทด์ 504-1588) ด้วยความแม่นยำที่ดีขึ้นเมื่อเทียบกับแบบจำลองที่เผยแพร่ก่อนหน้านี้ (Wayment-Steele, 2022)
7. Amgad Muneer และ คณะได้ศึกษาคุณสมบัติการย่อยสลายของเบส RNA แต่ละตัวในโมเลกุล เพื่อตรวจสอบว่าการเรียนรู้เชิงลึกแบบผสมผสานสามารถทำนายการย่อยสลาย RNA จากลำดับ RNA ได้หรือไม่ โดยเสนอแบบจำลองโครงข่ายประสาท ได้แก่ GCN_GRU และ GCN_CNN ผลการทดลองพบว่าร GCN_GRU มีประสิทธิภาพดีกว่าโมเดล GCN_CNN แบบจำลองได้รับคะแนนสูงสุดจากคะแนนการทดสอบ MCRMSE 0.22614 และ 0.34152 ตามลำดับ ในที่สุด (Amgad Muneer, 2022)
8. Feiran Cheng ศึกษาปัจจัยที่มีอิทธิพลต่อความคงตัว ตั้งแต่โครงสร้าง mRNA สารเพิ่มปริมาณระบบการนำส่งอนุภาคนาโนไขมัน (LNP) และกระบวนการผลิตเป็นปัจจัยหลักส่งผลต่อความคงตัวของวัคซีนเอ็มอาร์เอ็นเอ การปรับโครงสร้างเอ็มอาร์เอ็นเอให้เหมาะสมและคัดกรองสารเพิ่มปริมาณได้อย่างมีประสิทธิภาพปรับปรุงความเสถียรของวัคซีนเอ็มอาร์เอ็นเอ นอกจากนี้ การปรับปรุงกระบวนการผลิตยังสามารถเตรียมความพร้อมวัคซีนเอ็มอาร์เอ็นเอที่มีความเสถียรทางความร้อนพร้อมความปลอดภัยและสรุปปัจจัยสำคัญที่มีผลต่อความคงตัวของวัคซีนเอ็มอาร์เอ็นเอและเสนอแนวทางการวิจัยที่เป็นไปได้เพื่อปรับปรุงความเสถียรของวัคซีนเอ็มอาร์เอ็นเอ(Cheng F, 2023)
9. จากชุดข้อมูลของ OpenVaccine ที่มีฐานข้อมูล RNA พร้อมการวัดอัตราการย่อยสลายโดยนักวิจัยของ Stanford งานวิจัยนี้ Sheikh Asif Imran และคณะได้พัฒนาแบบจำลองโครงข่าย

ประสาทเพื่อช่วยให้นักวิจัยด้านชีวสารสนเทศสามารถระบุได้ว่าเอ็มอาร์เอ็นเอที่จะไม่เสถียรมีแนวโน้มที่จะเกิดขึ้นที่ตำแหน่งด้วยแบบจำลอง LSTM พบว่าค่าการทดสอบ MCRMSE อยู่ที่ 0.38796 (S. Asif Imran, 2020)

10. ปัญญาประดิษฐ์และการเรียนรู้โดยเครื่องจักร โดยเฉพาะอย่างยิ่งการเรียนรู้เชิงลึกไxนำไปสู่การปรับปรุงครั้งใหญ่ในสาขาวิทยาศาสตร์และวิศวกรรมศาสตร์หลายสาขา เนื่องจากความสามารถในการเรียนรู้คุณลักษณะต่างๆ อย่างลึกซึ้ง โดยเฉพาะการค้นพบวัคซีนสำหรับโรค (Arash Keshavarzi Arshadi, 2020)

จากงานวิจัยข้างต้นที่กล่าวจะเห็นได้ว่ามีการศึกษาอัตราการสลายตัวของวัคซีนในหลายๆ งานเพื่อความเข้าใจในหลักการทำงานของวัคซีนเอ็นอาร์เอ็นเอ และพยายามที่ทำให้วัคซีนคงตัวนานที่สุด และในการศึกษานี้ ผู้วิจัยได้ทดลองแบบเปรียบเทียบแบบจำลองที่ไม่ซับซ้อนมากนักว่าจะสามารถทำงานได้ดีและให้ผลลัพธ์ใกล้เคียงกับงานวิจัยอื่นๆที่ใช้แบบจำลองซับซ้อน โดยเน้นที่การดูประสิทธิภาพของ Xg boost ,Random Forest,Cat boost และLightGBMโดยจะกล่าวถึงในส่วนถัดไป

สรุปงานวิจัยที่ศึกษาทั้งหมด

ลำดับที่	เรื่อง	ปีที่ตีพิมพ์	แบบจำลอง					อื่นๆ
			LR	RF	XgB	SVM	KNN	
1	RNA under attack: cellular handling of RNA damage	2009						ปัจจัยที่มีผลต่อการสลายตัวของอาร์เอ็นเอ
2	mRNA vaccines - a new era in vaccinology	2018						การพัฒนาวัคซีนเอ็มอาร์เอ็นเอ
3	COVID-19 coronavirus vaccine design using reverse vaccinology and machine learning	2020	/	/	/	/	/	
4	Improving reverse vaccinology with a machine learning approach	2011				/		
5	Challenges of Storage and Stability of mRNA-Based COVID-19 Vaccines	2021						หากลยุทธ์เพื่อเพิ่มความคงตัวของวัคซีนที่ใช้เอ็มอาร์เอ็นเอ
6	Deep learning models for predicting RNA degradation via dual crowdsourcing.	2022						GNN
7	Vaccine-Deep: Prediction of COVID-19 mRNA vaccine degradation using deep learning	2022						GCN_GRU และ GCN_CNN
8	Research Advances on the Stability of mRNA Vaccines	2023						ศึกษาปัจจัยที่มีอิทธิพลต่อความคงตัวโครงสร้าง mRNA
9	COVID-19 mRNA Vaccine Degradation Prediction using Regularized LSTM Model,	2022						LSTM
10	Artificial Intelligence for COVID-19 Drug Discovery and Vaccine Development	2020						วิเคราะห์การใช้ปัญญาประดิษฐ์ในการค้นพบวัคซีน

ภาพประกอบ 10 แสดงตารางสรุปงานวิจัยที่ศึกษา

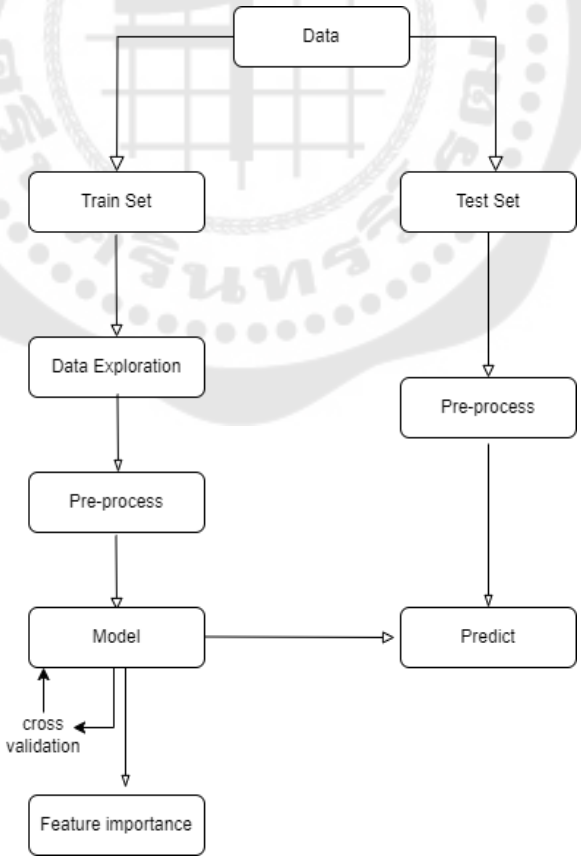
บทที่ 3
การดำเนินการวิจัย

แผนขั้นตอนดำเนินการวิจัย

ในการวิจัยครั้งนี้ ผู้วิจัยดำเนินงานตามหัวข้อดังนี้

1. ทำ Literature Review ที่เกี่ยวข้อง
2. หาชุดข้อมูล(data set)เพื่อใช้ในงานวิจัย
3. เตรียมข้อมูล
4. ศึกษาวิธีการสร้างและสร้างโมเดลในการทำนายข้อมูล
5. ปรับจูนโมเดลและวัดประสิทธิภาพของโมเดล
6. ประเมินและสรุปงานวิจัย

แผนผังดำเนินงานระบบ



ภาพประกอบ 11 แผนผังการทำงานของระบบ

จากภาพประกอบ 11 ได้อธิบายถึงกระบวนการสร้างแบบจำลองและการวัดผลลัพธ์จากการวิเคราะห์ข้อมูล โดยเริ่มจากการนำข้อมูล (Data) จาก Kaggle เข้ามาทำการแบ่งชุดข้อมูลออกเป็นชุดเรียนรู้ (train) และชุดทดสอบ (test) นำข้อมูลชุดเรียนรู้มาทำการสำรวจเบื้องต้น ก่อนนำเข้าสู่กการเตรียมข้อมูลให้พร้อม (Pre-processing) เพื่อทำไปใช้ในการสร้างแบบจำลอง (Model) ทำ Cross validation เพื่อหาค่าที่ทำให้แบบจำลองดีที่สุดแล้วทำการรันแบบจำลอง ที่หากผลลัพธ์ของแบบจำลองยังไม่เป็นที่น่าพอใจ จะกลับไปปรับการปรับพารามิเตอร์ เพื่อให้ได้มาซึ่งแบบจำลองที่มีประสิทธิภาพดีที่สุด และนำแบบจำลองไปใช้กับชุดข้อมูลทดสอบที่ทำการเตรียมข้อมูลแล้ว

ชุดข้อมูล(data set)เพื่อใช้ในงานวิจัย

ในการวิจัยครั้งนี้ผู้วิจัยได้ใช้ข้อมูลโครงการ OpenVaccine: COVID-19 mRNA Vaccine Degradation Prediction จาก Kaggle โดยความร่วมมือของมหาวิทยาลัยแสตนฟอร์ดและชุมชนอีเทอน่า (Stanford, 2020) โดยจากข้อมูลจะแบ่งเป็น

train.json – ข้อมูลชุดเรียนรู้มีค่าของการเก็บข้อมูลภาคสนาม (Ground Truth) ประกอบด้วย 5 เมตริกของการเกิดปฏิกิริยาย่อยสลาย ดังที่ระบุไว้ในตารางที่ 1 อย่างไรก็ตามการศึกษาครั้งนี้มุ่งเน้นไปที่การประเมิน 3 เมตริกแรก ได้แก่ ปฏิกิริยา deg_Mg_ph10 และ deg_Mg_50C

test.json - ชุดทดสอบโดยไม่มีคอลัมน์ใด ๆที่เกี่ยวข้องกับค่าของการเก็บข้อมูลภาคสนาม

ตาราง 1 รายละเอียดคอลัมน์ของชุดฝึกฝน

คอลัมน์	รายละเอียดคอลัมน์
Reactivity	ค่าความสามารถในการเกิดปฏิกิริยาทั่วไป
deg_pH10	ความเป็นไปได้ของการสลายตัวหลังการบ่มที่ pH สูง (pH10) โดยไม่มีแมกนีเซียม
deg_Mg_pH10	ความเป็นไปได้ของการสลายตัวหลังจากบ่มด้วยแมกนีเซียมใน pH สูง (pH 10)
deg_50C	ความเป็นไปได้ของการสลายตัวหลังจากการบ่มโดยไม่มีแมกนีเซียมที่อุณหภูมิสูง (50 องศาเซลเซียส)
deg_Mg_50C	ความเป็นไปได้ของการสลายตัวหลังจากบ่มด้วยแมกนีเซียมที่อุณหภูมิสูง (50 องศาเซลเซียส)
Id	รหัสสำหรับแต่ละตัวอย่าง
seq_scored	ค่าจำนวนเต็มแสดงถึงจำนวนตำแหน่งที่ใช้ในการให้คะแนนด้วยค่าที่ทำนาย
seq_length	ความยาวของลำดับ
Sequence	ลำดับ RNA การรวมกันของ A, G, U และ C สำหรับแต่ละตัวอย่าง

กระบวนการสำรวจข้อมูลเบื้องต้น (Exploratory Data Analysis)

มีขั้นตอนดังต่อไปนี้

1. ดูค่า Summary แต่ละคอลัมน์ และ ตรวจสอบรายละเอียดของข้อมูล
2. ตรวจสอบ Signal to noise และ SN filter
3. ตรวจสอบค่าเฉลี่ยการย่อยสลายของเป้าหมาย

ดูชนิดของข้อมูลและ เช็ค Summary **ของแต่ละคอลัมน์**

ตรวจสอบว่าข้อมูลมีส่วนที่ว่างหรือสูญหายหรือไม่ โดยข้อมูลชุดนี้ไม่มีการสูญหายของข้อมูล และดูชนิดของข้อมูลว่าประกอบด้วยอะไรบ้างในแต่ละคอลัมน์

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2400 entries, 0 to 2399
Data columns (total 19 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   index                2400 non-null   int64  
 1   id                   2400 non-null   object 
 2   sequence             2400 non-null   object 
 3   structure            2400 non-null   object 
 4   predicted_loop_type  2400 non-null   object 
 5   signal_to_noise      2400 non-null   float64
 6   SN_filter            2400 non-null   int64  
 7   seq_length           2400 non-null   int64  
 8   seq_scored           2400 non-null   int64  
 9   reactivity_error     2400 non-null   object 
 10  deg_error_Mg_pH10    2400 non-null   object 
 11  deg_error_pH10       2400 non-null   object 
 12  deg_error_Mg_50C     2400 non-null   object 
 13  deg_error_50C        2400 non-null   object 
 14  reactivity           2400 non-null   object 
 15  deg_Mg_pH10          2400 non-null   object 
 16  deg_pH10             2400 non-null   object 
 17  deg_Mg_50C           2400 non-null   object 
 18  deg_50C              2400 non-null   object 
dtypes: float64(1), int64(4), object(14)
memory usage: 356.4+ KB
```

ภาพประกอบ 12 ดูชนิดข้อมูล

เช็คค่าสถิติพื้นฐานของข้อมูลว่าระบุมาเพียงพอสำหรับการทำแบบจำลองไหม

	index	signal_to_noise	SN_filter	seq_length	seq_scored
count	2400.000000	2400.000000	2400.000000	2400.0	2400.0
mean	1199.500000	4.530456	0.662083	107.0	68.0
std	692.964646	2.835142	0.473099	0.0	0.0
min	0.000000	-0.103000	0.000000	107.0	68.0
25%	599.750000	2.391000	0.000000	107.0	68.0
50%	1199.500000	4.442500	1.000000	107.0	68.0
75%	1799.250000	6.294250	1.000000	107.0	68.0
max	2399.000000	17.194000	1.000000	107.0	68.0

ภาพประกอบ 13 ดู Summary ของข้อมูล

ทำการตรวจสอบรายละเอียดของข้อมูลในตารางเพื่อดูเอาท์พุตที่ต้องใช้งานโดยยกตัวอย่างมา 1 ตัวอย่าง ตามรูปที่ 14

```
One_sequence : GGAAAAGCUCUAAUAACAGGAGACUAGGACUACGUAUUUCUAGGUAACUGGAAUAACCCAUACCAGCAGUUAGAGUUCGCUCUAACAAAAGAAACAACAACAACAAC
One_structure : .....((((((.......)))).)).((.....((..(((((((....)))))))..)).....)).....(((((((....)))))))..................
One_predicted_loop_type : EEEEESSSSSSHHHHHHHSSSSBSSXSSIIIIISSIISSSSSSHHHHSSSSSSIISSIIIIISSXXXXSSSSSSSHHHHSSSSSSSEEEEEEEEEEEEEEEEEEEE
One_reactivity : [0.3297, 1.5693000000000001, 1.1227, 0.8686, 0.7217, 0.4384, 0.256, 0.33640000000000003, 0.21680000000000002, 0.3583,
```

ภาพประกอบ 14 ตัวอย่างของข้อมูลตามคอลัมน์

ใช้คำสั่ง Count และ len ในการนับจำนวนนิวคลีโอไทด์แรก 68 (ตัวอักษรในลำดับเอ็มอาร์เอ็นเอ 107 ตัวอักษร)

```
len(One_reactivity)
```

```
68
```

```
Counter(One_sequence)
```

```
Counter({'A': 45, 'C': 23, 'G': 19, 'U': 20})
```

```
sum_one_sequence = sum(Counter(One_sequence).values())
sum_one_sequence
```

```
107
```

```
Counter(One_structure)
```

```
Counter({'(': 23, ')': 23, '.': 61})
```

```
sum_one_structure = sum(Counter(One_structure).values())
sum_one_structure
```

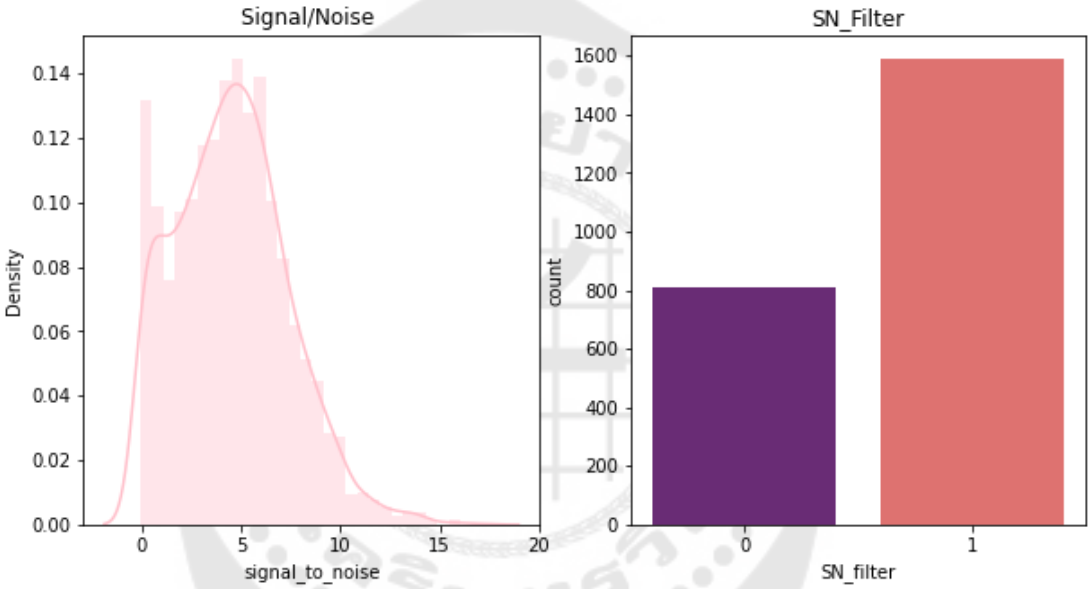
```
107
```

ภาพประกอบ 15 การใช้คำสั่ง count และ len

ตรวจสอบ Signal to noise และ SN filter

Signal to noise มาจากอัตราส่วนสัญญาณต่อสัญญาณรบกวน (หรือ SNR, S/N) และแสดงความกว้างของสัญญาณตามสัญญาณรบกวนเป็นการวัดมาตรฐานที่ใช้โดยทั่วไปในสัญญาณความถี่สูงเนื่องจากชุดฝึกฝนถูกสร้างขึ้นจากผลการทดลองในห้องปฏิบัติการ เราต้องการอธิบายว่าค่าต่างๆ ได้ผ่านขั้นตอนการควบคุมคุณภาพแล้ว

ในส่วนของ SN_filter มีไว้เพื่อเป็นการตรวจสอบว่าขั้นตอนการควบคุมคุณภาพทำงานได้ตามปกติ

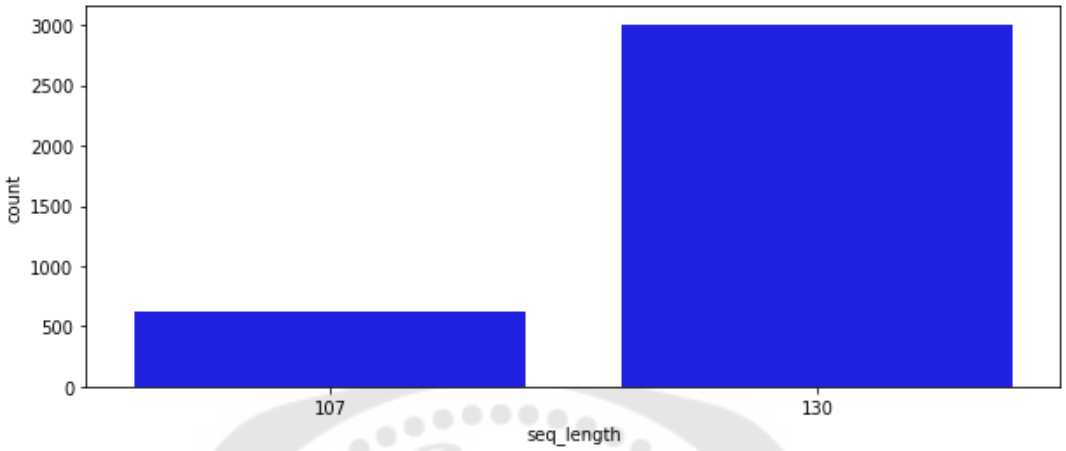


ภาพประกอบ 16 จำนวน SN_filter และ signal to noise

ในงานวิจัยนี้ ค่าต่ำสุดของเงื่อนไขทั้ง 5 ตามตารางที่ 1 ต้องมากกว่า -0.5 Signal/Noise เฉลี่ยทั้ง 5 เงื่อนไขต้องมากกว่า 1.0 หากเป็นไปตามเงื่อนไข SN_filter จะเท่ากับ 1

จากรูปที่ 16 ยังพบ signal_to_noise มีค่าน้อยกว่า - 0.5 และ SN_filter เท่ากับ 0 ซึ่งอาจจะเป็น outlier ที่เกิดขึ้น โดย signal_to_noise มากกว่า 1 จำนวน 2096 ตัวอย่าง ,SN_filter เท่ากับ 1 จำนวน 1589 ตัวอย่าง

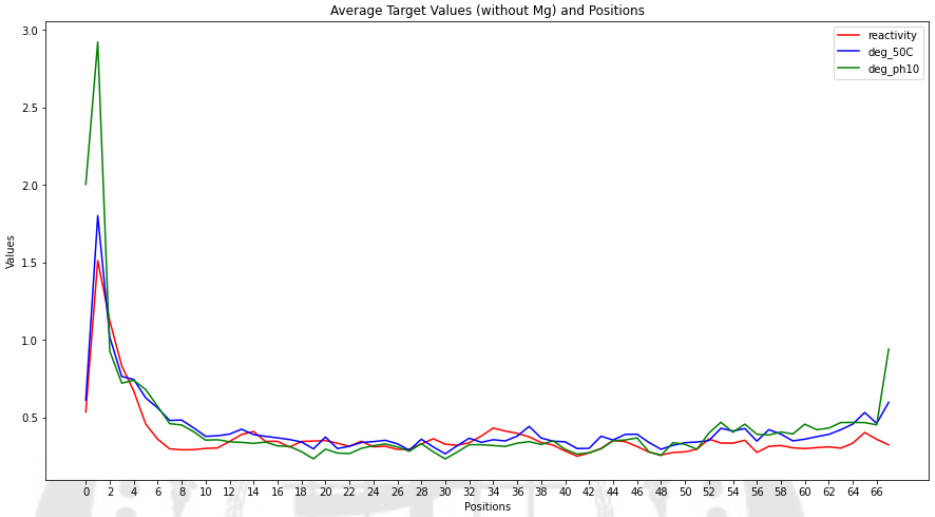
ข้อมูลฝึกฝนประกอบด้วยความยาว sequence 107 เท่านั้น ข้อมูลการทดสอบส่วนใหญ่มีความยาว sequence 130



ภาพประกอบ 17 จำนวนความยาว sequence ข้อมูลทดสอบ

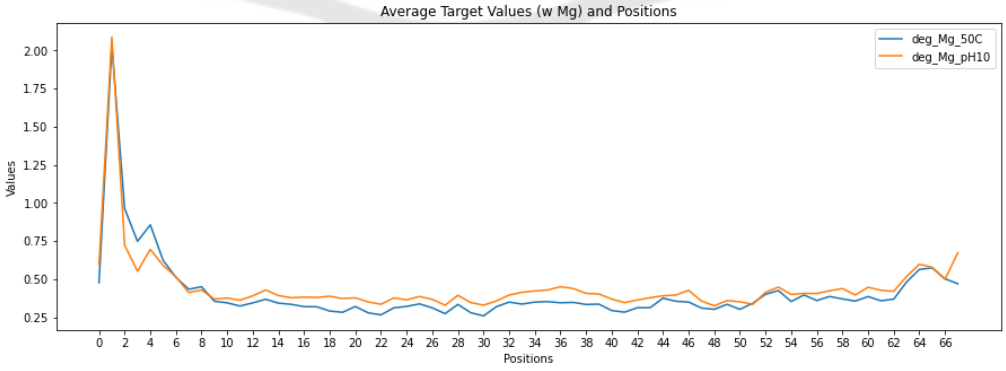
ตรวจสอบค่าเฉลี่ยการย่อยสลายของเป้าหมาย

ทำการตรวจสอบค่าเฉลียการย่อยสลายของเป้าหมายโดยเราจะเห็นได้ว่าตำแหน่งการย่อยสลายและการเกิดปฏิกิริยาจะเกิดสูงสุดที่ตำแหน่งต้นสาย sequence โดยเฉพาะที่ pH10 ตามรูปที่ 18 จากภาพเราจะเห็นความสัมพันธ์ระหว่างอุณหภูมิสูง 50C และ pH10



ภาพประกอบ 18 คำนวณค่าเฉลี่ยสำหรับคอลัมน์เป้าหมายแบบไม่ใช้แมกนีเซียม

และในสภาวะการใช้แมกนีเซียมร่วมโดยส่วนใหญ่แล้ว deg_Mg_pH10 จะมีการลลายตัวมากกว่าที่ deg_Mg_50C แต่ทั้ง 2 เงื่อนไขมีความสัมพันธ์กันอย่างเห็นได้ชัดจากรูปที่ 19



ภาพประกอบ 19 คำนวณค่าเฉลี่ยสำหรับคอลัมน์เป้าหมายแบบใช้แมกนีเซียม

ศึกษาวิธีการสร้างและสร้างโมเดลในการทำนายข้อมูล

นำข้อมูลที่ได้จากกระบวนเตรียมข้อมูลมาใช้ทดสอบโมเดลโดยงานวิจัยนี้ใช้โมเดลเปรียบเทียบกันทั้งหมด 4 โมเดล คือ Xgboost ใช้เป็น Baseline ,Random Forest, Catboost และ LightGBM ในการตั้งค่าพารามิเตอร์ ใช้เทคนิค 5-Fold cross- validation และทำการปรับโมเดลเพื่อหาค่าพารามิเตอร์ให้เหมาะสมด้วยการใช้ GridSearchCV

ทดสอบการทำแบบจำลองด้วยเทคนิค XGboost (baseline)

เป็นแบบจำลองที่นำเอา Decision Tree มาต่อกันโดยที่แต่ละ decision tree จะเรียนรู้จาก error ของ tree ก่อนหน้า เมื่อมีการเรียนรู้ของ tree ต่อเนื่องกันจนมีความลึกมากพอ และโมเดลจะหยุดเรียนรู้เมื่อไม่เหลือรูปแบบของ error จาก tree ก่อนหน้าให้เรียนรู้แล้ว

```
xgb = XGBRegressor(
    subsample=0.8,
    colsample_bytree=0.75,
    reg_lambda=2,
    reg_alpha=1,
    random_state=28
)

params = {
    'estimator__n_estimators': [ 800, 900, 1000],
    'estimator__learning_rate': [0.1, 0.25, 0.30],
    'estimator__max_depth': [3, 4, 5]
}
```

ภาพประกอบ 20 การกำหนดค่าพารามิเตอร์เทคนิค XGboost

```
gs.best_params_
```

```
{'estimator__learning_rate': 0.1,
 'estimator__max_depth': 5,
 'estimator__n_estimators': 800}
```

ภาพประกอบ 21 พารามิเตอร์ที่ได้จาก GridSearchCV ในเทคนิค XGboost

ทดสอบการทำแบบจำลองด้วยเทคนิค Random Forest

เป็นแบบจำลองที่นำ Decision Tree หลายๆ tree มา Train ร่วมกัน (ตั้งแต่ 10 ต้น ถึงมากกว่า 1000 ต้น) โดยที่แต่ละ tree จะได้รับ feature และ data เป็น subset ของ feature และ data ทั้งหมด แบบสุ่มตอนคาดการณ์ แต่ละ Decision Tree ทำการคาดการณ์ของตัวมันเอง และเลือกผล final prediction

```
rf = RandomForestRegressor(random_state=42,verbose=1)
params = {
    'estimator__n_estimators': [ 100, 200, 300],
    'estimator__max_depth': [8, 9, 10],

}
reg = MultiOutputRegressor(rf)
# Perform 5 fold cross validation on set 1 features
rfr = GridSearchCV(estimator=reg, param_grid = params, cv = 5, n_jobs=-1, scoring=custom_scorer)
```

ภาพประกอบ 22 การพารามิเตอร์ของเทคนิค Random Forest

ทดสอบการทำแบบจำลองด้วยเทคนิค Catboost

เป็นแบบจำลองการจัดการสำหรับคุณสมบัติตามหมวดหมู่ ได้รับความนิยมเมื่อเทียบกับอัลกอริธึม Gradient boosting อื่น ๆ เนื่องจากคุณสมบัติการใช้ Oblivious Trees หรือ Symmetric Trees เพื่อให้โมเดลทำงานได้เร็วขึ้น

```
[ ] cbr = CatBoostRegressor(random_state=28)
    params = {
        'estimator__n_estimators': [ 800, 900, 1000],
        'estimator__learning_rate': [0.1, 0.25, 0.30],
        'estimator__max_depth': [3, 4, 5]
    }
    reg = MultiOutputRegressor(cbr)
    # Perform 5 fold cross validation on set 1 features
    cb = GridSearchCV(estimator=reg, param_grid = params, cv = 5, n_jobs=-1, scoring=custom_scorer)
```

ภาพประกอบ 23 การกำหนดพารามิเตอร์ของเทคนิค Catboost

ทดสอบการทำแบบจำลองด้วยเทคนิค Light GBM

เป็นเฟรมเวิร์ก gradient boosting framework ที่รวดเร็ว,ประสิทธิภาพสูงโดยใช้อัลกอริธึมแผนผังและงานการเรียนรู้ของเครื่องอื่น ๆ อีกมากมาย ใช้วิธีการแยกความลึกของ Trees

```
FOLD_N = 5
gkf = GroupKFold(n_splits=FOLD_N)
```

```
params = {'objective': 'regression',
          'boosting': 'gbdt',
          'metric': 'rmse',
          'learning_rate': 0.05,
          'seed' : SEEDS}
```

ภาพประกอบ 24 การกำหนดพารามิเตอร์ของเทคนิค Light GBM

ปรับจูนโมเดลและวัดประสิทธิภาพของแบบจำลอง

ในการศึกษานี้ ผู้วิจัยเปรียบเทียบแบบจำลองทั้ง 4 แบบ แล้วนำโมเดลที่ดีที่สุดในงานวิจัยไปหา feature importance เพื่อหาตำแหน่งที่มีการย่อยสลาย

สำหรับงานวิจัยนี้มีการปรับค่าพารามิเตอร์หลักในการสร้างแบบจำลองทำนาย**ดังนี้**

learning_rate	Hyperparameter ควบคุมปรับ Weight ใน Step ของการฝึกฝน
n_estimators	weight function ที่ใช้ในการฝึกฝนและทำนาย
subsample	สัดส่วนของตัวอย่างข้อมูลสำหรับเรียนรู้
colsample_bytree	สัดส่วนในการสุ่มคอลัมน์
reg_lambda	L1 regularization เป็นตัวคูณหน้า weight
reg_alpha	L2 regularization เป็นตัวคูณหน้า weight
max_depth	เมตริกที่ใช้วัดระยะใน tree

และมีการปรับจูนพารามิเตอร์รายละเอียดตามตารางที่ 2

ตาราง 2 พารามิเตอร์ที่เหมาะสมของแต่ละโมเดลในที่ใช้ในการเรียนรู้

แบบจำลอง	พารามิเตอร์
XGBoost	learning_rate: 0.1, n_estimators: 800 subsample=0.8 colsample_bytree=0.75 reg_lambda=2 reg_alpha=1
Random Forest	max_depth: 11 n_estimators: 800
Catboost	learning_rate: 0.1 max_depth: 8 n_estimators: 300
LigthGBM	learning_rate: 0.05

การวิเคราะห์และประเมินผลลัพธ์เพื่อเปรียบเทียบโมเดลจะใช้วิธีการเปรียบเทียบด้วยค่า mean columnwise root mean squared

MCRMSE (mean columnwise root mean squared error)

โดยปกติเราสามารถคำนวณ root mean squared error (RMSE) เพื่อให้ได้เมตริกการประเมินตัวเลขเดียวสำหรับการคาดคะเนของเรา แต่ถ้าเราคาดการณ์หลายค่าพร้อมกัน ในกรณีของการแข่งขัน OpenVaccine เราจำเป็นต้องทำนายอัตราการย่อยสลายภายใต้เงื่อนไขหลายประการ เราจะได้ค่า RMSE ที่แตกต่างกันหลายค่า หนึ่งค่าสำหรับแต่ละคอลัมน์

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \quad (1)$$

ดังนั้น MCRMSE เป็นเพียงค่าเฉลี่ยของค่า RMSE ทั้งหมดสำหรับแต่ละคอลัมน์ของเรา ดังนั้นเราจึงยังคงสามารถใช้เมตริกการประเมินด้วยตัวเลขเดียวได้ แม้ในกรณีที่มีหลายผลลัพธ์

$$MCRMSE = \frac{1}{N_t}\sum_{j=1}^{N_t}\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_{ij} - \hat{y}_{ij})^2} \quad (2)$$

ที่มา:(Stanford, 2020)

จากสมการตัวแปร N_t คือจำนวนตัวแปรที่ใช้คาดการณ์

n คือ จำนวนของชุดทดสอบ

ตัวแปร y และตัวแปร $\hat{y}$ คือค่าจริงและค่าที่ทำนายตามลำดับค่าเฉลี่ย

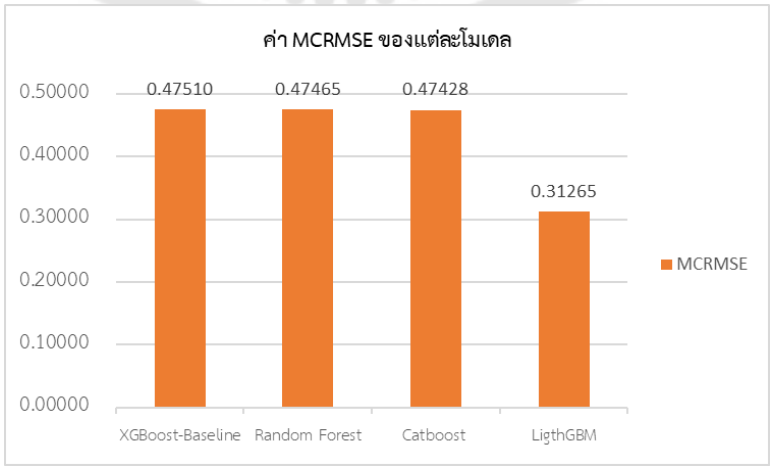
บทที่4
ผลการดำเนินงานวิจัย

ในการศึกษาการทำนายค่าความเสถียรของวัคซีนเอ็มอาร์เอ็นเอ ซึ่งใช้ข้อมูลเกี่ยวกับการอัตราการย่อยสลายและตำแหน่งการย่อยสลายของ RNA ของข้อมูลที่ได้จากชุมชน Kaggle ผู้วิจัยได้วิจัยโดยการศึกษาขั้นตอนต่างๆ ตามจุดประสงค์ในการวิจัยคือ

1. ศึกษาการสร้างแบบจำลองการทำนายค่าความเสถียรของวัคซีนเอ็มอาร์เอ็นเอสำหรับโรคโควิด 19 ด้วยวิธีการเรียนรู้ของเครื่องจักร
2. เปรียบเทียบการทำนายความเสถียรของโมเดลการเรียนรู้เครื่องจักร

ผลการเปรียบเทียบค่า MCRMSE จาก baseline ได้ค่า MCRMSE 0.4751 นั้น เมื่อนำเทคนิค Random Forest และ Catboost มาใช้งานและหาพารามิเตอร์ที่เหมาะสมพบว่าค่า MCRMSE ลดลงเล็กน้อยคือ 0.47465 และ 0.47428 แต่เมื่อใช้ LigthGBM พบว่าค่าลดลงจาก baseline -คือ 0.31265 เมื่อค่า MCRMSE น้อยแสดงให้เห็นว่าเทคนิค LigthGBM มีค่าความคลาดเคลื่อนที่ค่อนข้างดีกว่าเทคนิคอื่นๆ ซึ่งแสดงตามภาพประกอบที่ 25 ดังนั้นจึงให้ผลลัพธ์ที่แม่นยำกว่า

ทางผู้วิจัยจึงเลือกเทคนิคนี้ในการนำไปหา Feature Importance ต่อไป



ภาพประกอบ 25 ค่า MCRMSE ของแต่ละแบบจำลอง

การหา Feature importance

Feature importance เป็นฟังก์ชั่นที่ถูกนำมาใช้เพื่อบอกว่า feature ใดที่มีผลต่อแบบจำลองโดยนำมาเทียบกับ feature อื่นๆ ในการทดลองนี้ผู้วิจัยนำแบบจำลอง LightGBM ที่ได้มาหา Feature importance โดยกำหนด parameter ตามภาพประกอบ 26 ดังนี้

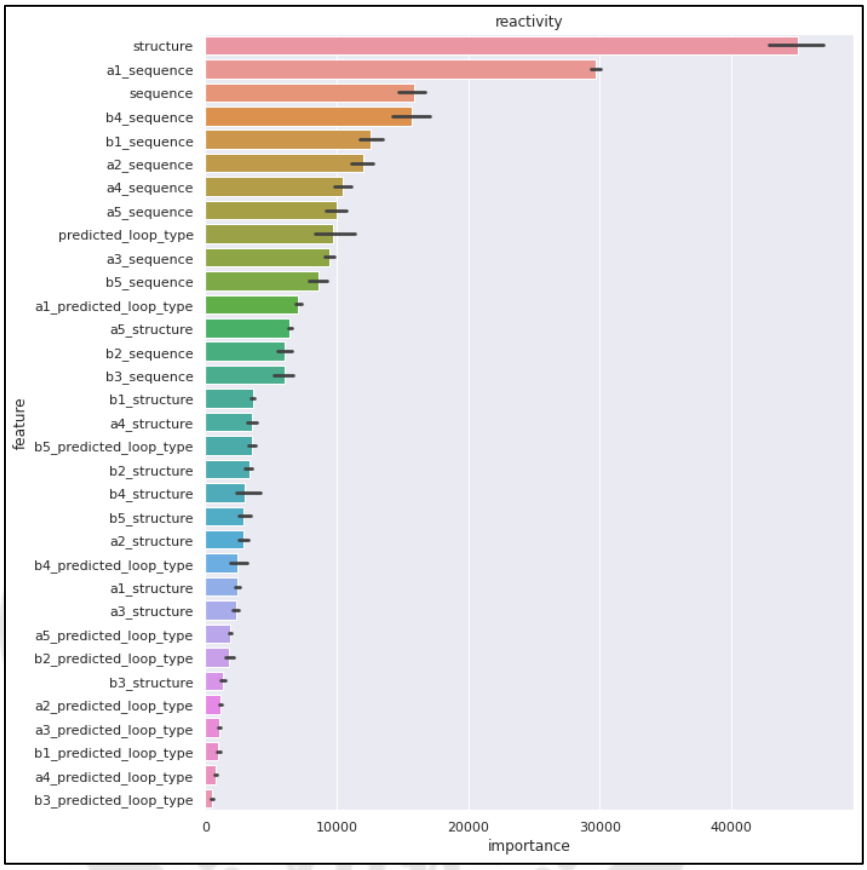
```
def feature_importances_(self):
    if self.model_type == 'lgb':
        return self.model.feature_importance(importance_type='gain')
```

ภาพประกอบ 26 parameter ของ feature importance

ปัจจัย (importance) ที่ส่งผลของประสิทธิภาพในการทำนายอัตราการย่อยสลายของ sequence จะมีทั้งหมด 5 ชนิด

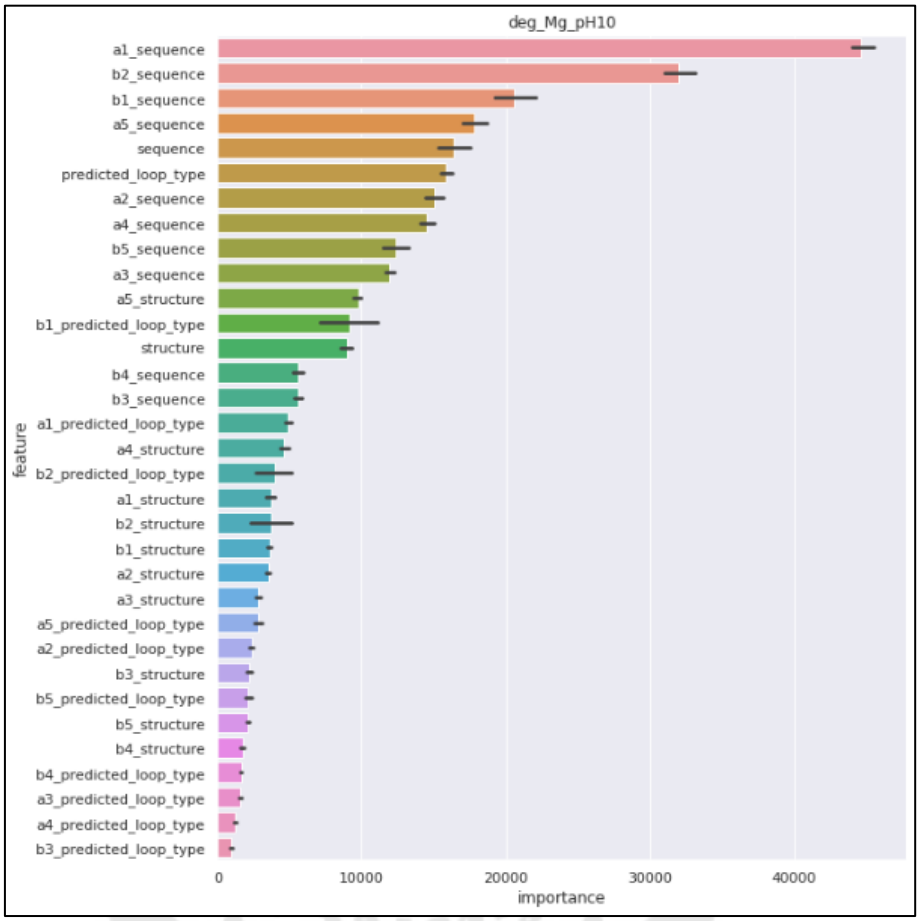
1. การเกิด reactivity
2. การย่อยสลายโดยการใช้แมกนีเซียมที่ pH10
3. การย่อยสลายโดย pH10 อย่างเดียว
4. การย่อยสลายโดยการใช้แมกนีเซียมอุณหภูมิ 50 องศาเซลเซียส
5. การย่อยสลายที่อุณหภูมิ 50 องศาเซลเซียสอย่างเดียว

Feature ส่งผลของประสิทธิภาพในการทำนายอัตราการย่อยสลายของ sequence โดยการเกิด reactivity เป็นส่วนที่เรียกว่า structure (ส่วนที่ไว้ใช้ในการจับคู่สายของ เอ็มอาร์เอ็นเอ) มีค่า importance สูงที่สุด รองลงมาคือส่วนของ a1_sequence จากรูปที่ 27



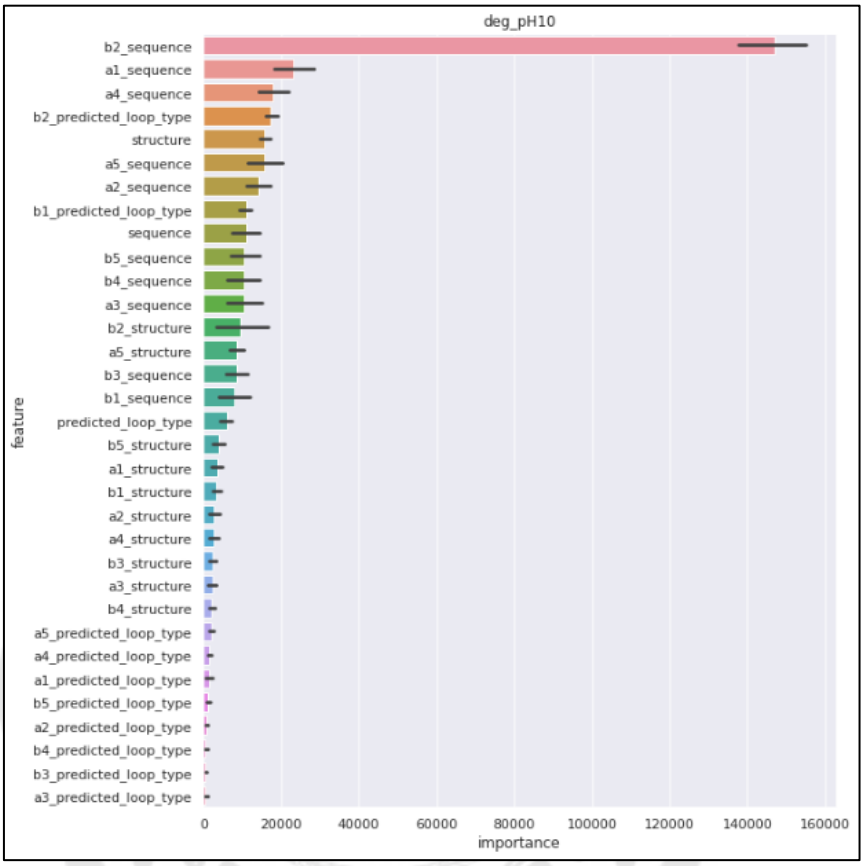
ภาพประกอบ 27 Feature importance ของค่า reactivity

Feature ที่ส่งผลต่อประสิทธิภาพในการทำนายอัตราการย่อยสลายโดยการใช้แมกนีเซียมที่ pH10 เป็นส่วนที่เรียกว่า a1 sequence มีค่า importance สูงที่สุด เช่นกัน ตามรูปที่ 28



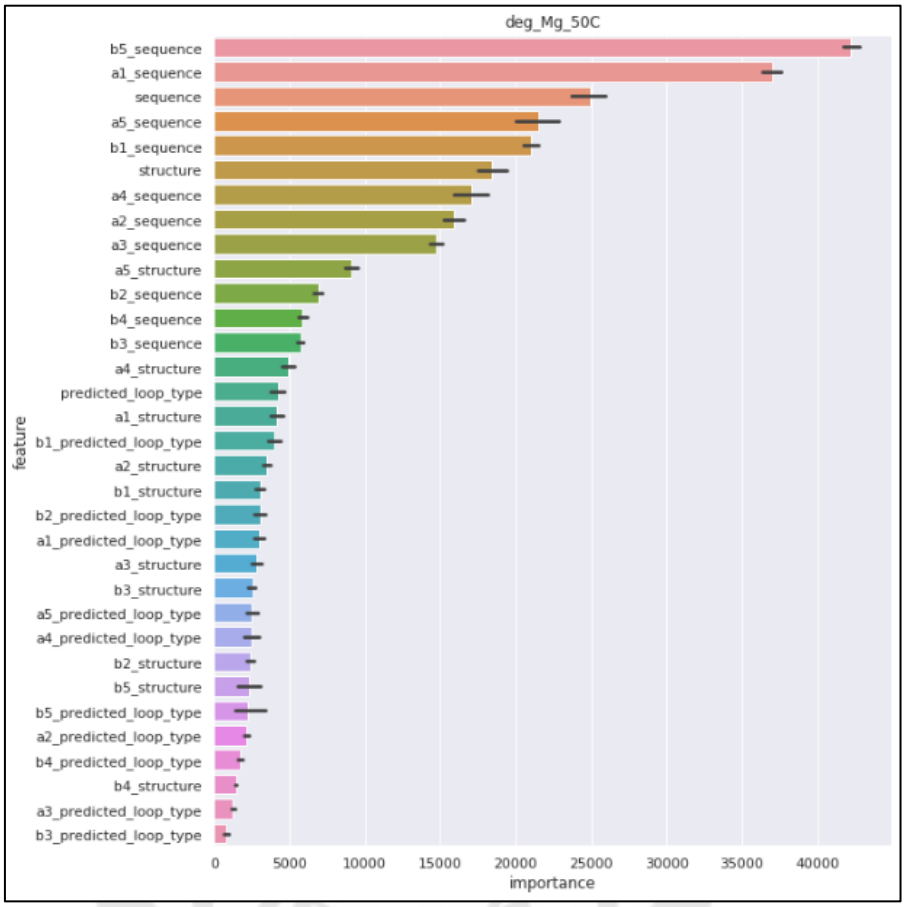
ภาพประกอบ 28 Feature importance ของค่า deg_Mg_pH10

ในทางกลับกัน Feature ที่ส่งผลของประสิทธิภาพในการทำนายอัตราการย่อยสลายโดย pH10 เพียงอย่างเดียว เป็นส่วนที่เรียกว่า b2_sequence ซึ่งมีผลเยอะกว่า a1 sequence ตามรูปที่ 29



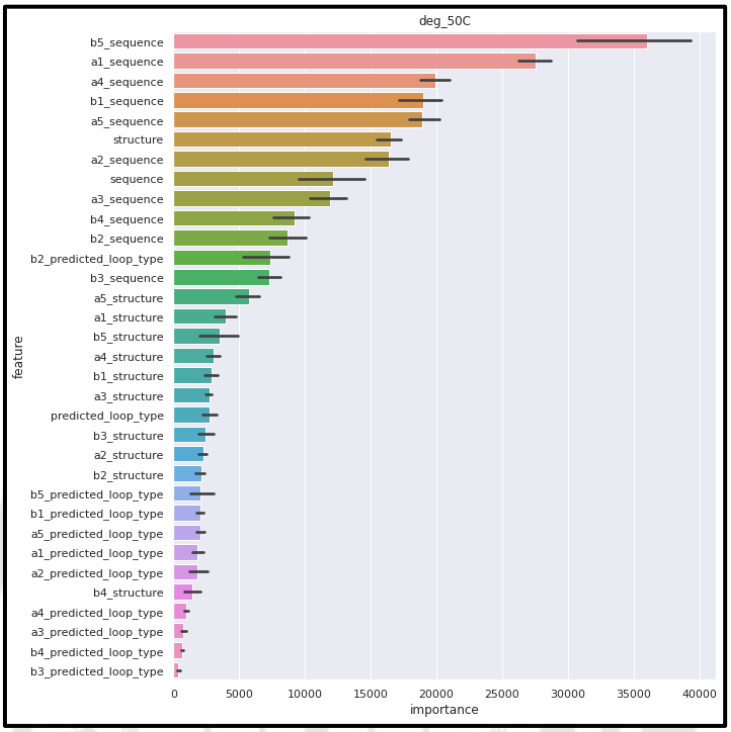
ภาพประกอบ 29 Feature importance ของค่า deg_pH10

Feature ที่ส่งผลของประสิทธิภาพในการทำนายอัตราการย่อยสลายโดยการใช้แมกนีเซียม อุณหภูมิ 50 องศาเซลเซียส คือ b5_sequence รองลงมาคือส่วนของ a1_sequence ตามรูปที่ 30



ภาพประกอบ 30 Feature importance ของค่า deg_Mg_50C

Feature ที่ส่งผลของประสิทธิภาพในการทำนายอัตราการย่อยสลายที่อุณหภูมิ 50 องศาเซลเซียส อย่างเดียว คือ b5_sequence รองลงมาคือส่วนของ a1_sequence อีกเช่นกัน ตามรูปที่ 31



ภาพประกอบ 31 Feature importance ของค่า deg_50C

บทที่ 5
สรุปผลการวิจัย อภิปรายผลและข้อเสนอแนะ

ในการวิจัยเพื่อศึกษา เพื่อศึกษาการสร้างแบบจำลองการทำนายค่าความเสถียรของวัคซีนเอ็มอาร์เอ็นเอสำหรับโรคโควิด 19 ด้วยวิธีการเรียนรู้ของเครื่องจักรทั้งหมด 4 แบบจำลอง คือ Xgboost ,Random Forest, Catboost และ LightGBM ทำนายค่าความเสถียรของวัคซีนเอ็มอาร์เอ็นเอสำหรับโรคโควิด 19 ซึ่งเป็นอัตราการย่อยสลายในสภาวะต่างๆ โดยใช้หลักการของ MCRMSE สามารถแบ่งหัวข้อในการสรุปผลได้ดังต่อไปนี้

1. สรุปผลการวิจัย และ อภิปรายผลการวิจัย
2. ข้อเสนอแนะ

5.1 สรุปผลการวิจัย และ อภิปรายผลการวิจัย

วัคซีนชนิดเอ็มอาร์เอ็นเอเป็นหนึ่งในเทคโนโลยีที่ถูกนำมาใช้ในการป้องกันการแพร่ระบาดของโรคโควิด 19 โดยวัคซีนชนิดนี้มีประสิทธิภาพในการป้องกันโรคได้สูงถึง 95% งานวิจัยนี้นำเสนอการเรียนรู้ของเครื่องจักรทั้งหมด 4 แบบ คือ XGboost, Random Forest, Catboost และ Light GBM ในการทำนายความเสถียรของการย่อยสลายของลำดับอาร์เอ็นเอ โดยจากผลการทดลองพบว่า โมเดลการเรียนรู้ของเครื่องจักรชนิด Light GBM เป็นตัวทำนายที่ดีที่สุดโดยมีค่า Mean Column-wise Root Mean Squared (MCRMSE) เท่ากับ 0.31265 เมื่อเทียบกับ best score ของ challenge อยู่ที่ 0.23162 จากการค้นคว้าเพิ่มเติมเกี่ยวกับเทคนิคของ LightGBM พบว่าถูกใช้ในงานเกี่ยวกับการ identify tissue-specific genes โดยอ้างอิงจากงานของ Zijie Wang และคณะ (Wang, 2023) ใช้ parameter ‘gain’ ในการค้นหายีนเฉพาะของพืช และเป็นเทคนิคที่เหมาะกับขนาดของข้อมูลที่มีความเยอะประมาณนึง อีกทั้งตัวของ LightGBM ยังมีอัลกอริทึม Leaf wise ซึ่งสามารถลดการสูญเสียได้มากกว่าอัลกอริทึม Level-wise ดังนั้นจึงน่าจะเป็นเหตุผลที่ LightGBM ให้ผลลัพธ์ที่ค่อนข้างดีกว่าเทคนิคอื่นๆ และเมื่อนำมาหาส่วนสำคัญที่มีผลกับประสิทธิภาพของโมเดลจะได้ตามตารางที่3

ตาราง 3 Feature Importance ที่สำคัญที่สุดของแต่ละเป้าหมาย

Target	Feature importance
Reactivity	Structure
deg_Mg_pH10	a1_sequence
deg_Mg_50C	b5_sequence
deg_pH10	b2_sequence
deg_50C	b5_sequence

การทำ Feature Importance แต่ละปัจจัยหรือเป้าหมายมีตำแหน่งของเอ็มอาร์เอ็นเอ ในการย่อยสลายที่แตกต่างกันไป จากผลที่ได้จะเห็นที่ได้ว่าที่ ปัจจัยอุณหภูมิ 50 องศาเซลเซียสและปัจจัยอุณหภูมิ 50 องศาเซลเซียสในแมกนีเซียม sequence ที่มีผลต่อประสิทธิภาพของโมเดลเป็น sequence เดียวกันคิอ b5_sequence ซึ่งเป็นสิ่งที่น่าสนใจแสดงให้เห็นว่าถ้าในห้องปฏิบัติการจริงที่อุณหภูมิสูงอาจจะมีผลต่อการย่อยสลายของ sequence ตำแหน่งดังกล่าวเป็นพิเศษ และอีกส่วนที่มองข้ามไม่ได้คือ a1_sequence ซึ่งเป็นส่วนที่มีผลอยู่ในอันดับต้นๆ ของทุกปัจจัย การใช้วิธีการเรียนรู้ของเครื่องจักรในงานวิจัยนี้มีค่า MCRMSE อยู่ประมาณ 0.31-0.48 การใช้เทคนิคดังกล่าวมีความสามารถในการเพิ่มความเร็วและประสิทธิภาพของการค้นพบวัคซีนเอ็มอาร์เอ็นเอที่เสถียรและมีผลต่อการวิจัยในสาขาอื่น ๆ ที่เกี่ยวข้อง เนื่องจากงานวิจัยนี้มุ่งเน้นสำหรับการศึกษาแบบจำลองที่ไม่ซับซ้อนมากนำมาเปรียบเทียบกัน และหาส่วนที่มีผลต่อการย่อยสลายของวัคซีนเอ็มอาร์เอ็นเอ ซึ่งเทคนิคนี้อาจถูกใช้เพื่อทดสอบคัดกรองเพื่อลบลำดับที่ไม่เสถียรออกไป ผู้วิจัยใช้ผลของ Public Leaderboard จากชุมชน Kaggle เมื่อนำไปเปรียบเทียบกับผลการทดลองอื่นๆซึ่งอาจจะเหมาะเมื่อนำไปต่อยอดด้วยการวิเคราะห์ด้านอื่นๆประกอบกัน

5.2 **ข้อเสนอแนะ**

ข้อจำกัดของงานนี้ที่พบคือความยาวของลำดับของโมเลกุลเอ็มอาร์เอ็นเอที่ใช้ความยาวที่ใช้ในงานวิจัยนี้อยู่ระหว่าง 107-130 ฐานและเป็นข้อมูลที่ถูกจำลองขึ้นในห้องปฏิบัติการเพื่อใช้ในการทดสอบ ในขณะที่วัคซีน โควิด 19 ที่แท้จริงน่าจะมีความยาวตั้งแต่ 3,000-4,000 ฐาน แต่ผลของงานนี้แสดงให้เห็นว่าอัลกอริทึมการทำนายดังกล่าวมีความเป็นไปได้และมีศักยภาพที่ช่วยย่นเวลา ในระหว่างกระบวนการวิจัยในระหว่างการระบาดของโรค ซึ่งในระยะยาวเทคนิคดังกล่าวอาจช่วยในการวิจัยได้ดีขึ้นในการทำความเข้าใจเหตุผลเบื้องหลังความเสถียรของโมเลกุลเอ็มอาร์เอ็น

เอบางชนิดและช่วยในการพัฒนาเทคโนโลยีที่เกี่ยวข้อง โดยหวังเป็นอย่างยิ่งว่างานนี้จะเป็นประโยชน์ต่อนักวิทยาศาสตร์ข้อมูลคนอื่น ๆ ในการสร้างการคาดการณ์ที่ดีขึ้นในอนาคต ในการปรับปรุงเสถียรภาพของวัคซีนเอ็มอาร์เอ็นเอจริงๆเป็นปัญหาที่ได้รับการสำรวจก่อนเกิดโรคระบาด การแก้ปัญหาโดยกระบวนการเรียนรู้ของเครื่องจะถูกใช้ในห้องปฏิบัติการ ก่อนที่จะนำเข้าสู่ภาคอุตสาหกรรม เพื่อเร่งการวิจัยวัคซีน mRNA และส่งมอบวัคซีนที่มีความเสถียรในตู้เย็นเพื่อป้องกันไวรัส SARS-CoV-2 ซึ่งเป็นไวรัสที่เป็นตัวก่อโรคโควิด-19

บรรณานุกรม

Amgad Muneer, S. M. F., Nur Arifin Akbar, David Agustriawan, Setyanto Tri Wahyudi,. (2022). iVaccine-Deep: Prediction of COVID-19 mRNA vaccine degradation using deep learning,. *Journal of King Saud University - Computer and Information Sciences,*.

Arash Keshavarzi Arshadi, J. W., Milad Salem, Emmanuel Cruz,. (2020). Artificial Intelligence for COVID-19 Drug Discovery and Vaccine Development.

Bowman, B. N., McAdam, P. R., Vivona, S., Zhang, J. X., Luong, T., Belew, R. K., . . . Woelk, C. H. (2011). Improving reverse vaccinology with a machine learning approach. *Vaccine, 29*(45), 8156-8164.

Cheng F, W. Y., Bai Y, Liang Z, Mao Q, Liu D, Wu X, Xu M. (2023). Research Advances on the Stability of mRNA Vaccines. Viruses. .

Eternagame. (2020). How to build a better vaccine from the comfort of your own web browser. สืบค้นจาก https://medium.com/eternaproject/how-to-build-a-better-vaccine-from-the-comfort-of-your-own-web-browser-233343e0210d

Florindo, H. F., Kleiner, R., Vaskovich-Koubi, D., Acúrcio, R. C., Carreira, B., Yeini, E., . . . Satchi-Fainaro, R. (2020). Immune-mediated approaches against COVID-19. *Nature Nanotechnology, 15*(8), 630-645.

Mackenzie, R. J. (2020). DNA vs. RNA – 5 Key Differences and Comparison. https://www.technologynetworks.com/genomics/articles/what-are-the-key-differences-between-dna-and-rna-296719

Ong, E., Wong, M. U., Huffman, A., และ He, Y. (2020). COVID-19 coronavirus vaccine design using reverse vaccinology and machine learning. *bioRxiv*.

Pardi, N., Hogan, M. J., Porter, F. W., และ Weissman, D. (2018). mRNA vaccines - a new era in vaccinology. *Nat Rev Drug Discov, 17*(4), 261-279.

S. Asif Imran, M. T. I., C. Shahnaz, M. Tafhimul Islam, O. Tawhid Imam and M. Haque, . (2020). COVID-19 mRNA Vaccine Degradation Prediction using Regularized LSTM Model,. *2020 IEEE International Women in Engineering (WIE) Conference on*

Electrical and Computer Engineering (WIECON-ECE).

S. H. Ing, A. A. A. a. S. K. (2021). Development of COVID-19 mRNA Vaccine Degradation Prediction System. *IEEE*.

Schlake, T., Thess, A., Fotin-Mleczek, M., และ Kallen, K. J. (2012). Developing mRNA-vaccine technologies. *RNA Biol, 9*(11), 1319-1330.

Stanford. (2020). OpenVaccine: COVID-19 mRNA Vaccine Degradation Prediction. สืบค้นจาก https://www.kaggle.com/c/stanford-covid-vaccine/data

Uddin MN, R. M. (2021). Challenges of Storage and Stability of mRNA-Based COVID-19 Vaccines. *Vaccines (Basel)*.

Wadhwa A, A. A., Lokras A, Foged C, Thakur A. . (2020). Opportunities and Challenges in the Delivery of mRNA-based Vaccines. Pharmaceutics.

Wang, Z. (2023). Comparative analysis of tissue-specific genes in maize based on machine learning models: CNNperforms technicallybest, LightGBM performs biologically soundest. *Frontiers in Genetics*.

Wayment-Steele, H. K., Kladwang, W., Watkins, A.M. et al. (2022). Deep learning models for predicting RNA degradation via dual crowdsourcing.

Wurtmann, E. J., และ Wolin, S. L. (2009). RNA under attack: cellular handling of RNA damage. *Crit Rev Biochem Mol Biol, 44*(1), 34-49.

เจาะลึกระบบสุขภาพ, ส. H. (2023). กรมควบคุมโรค เผยเคสโควิดยังเพิ่ม ผู้เชี่ยวชาญห่วงโควิดรุนแรงในเด็กอาจส่งผลต่อพัฒนาการ ย้ำวัคซีนจำเป็นในเด็กเล็ก. https://www.hfocus.org/content/2023/06/27769

ช้างแก้ว, น. พ. (2021). ตอบทุกข้อสงสัย “วัคซีนโควิด-19”. https://thainakarin.co.th/covid-19-vaccine-knowledge/

ไทยโพสต์. (2020). ข่าวดีมาก วัคซีน'โมเดอร์นา'ป้องกันโควิดได้ผลเกือบ95%. สืบค้นจาก https://www.thaipost.net/main/detail/84053

มหาวิทยาลัยมหิดล, ค. (2020). วัคซีน คืออะไร...แล้วทำไมเราถึงต้องฉีดวัคซีน ? สืบค้นจาก https://www.gj.mahidol.ac.th/main/knowledge-2/what-is-vaccine/

โมโลกุลอาร์เอ็นเอ (RNA Molecules)).

รัตนโรจน์พงศ์, ไ. (2021). โคโรน่าไวรัสสายพันธุ์ใหม่ (โควิด-19): องค์ความรู้ด้านงานวิจัยไวรัสวิทยา

การศึกษาข้อมูลด้านการตรวจวินิจฉัย การรักษา และการพัฒนาวัคซีน.

โรงพยาบาลศิครินทร์. วัคซีนโควิดมีกี่ชนิด?

https://www.sikarin.com/health/%E0%B8%A7%E0%B8%B1%E0%B8%84%E0%B8%8B%E0%B8%B5%E0%B8%99%E0%B9%82%E0%B8%84%E0%B8%A7%E0%B8%B4%E0%B8%94%E0%B8%A1%E0%B8%B5%E0%B8%81%E0%B8%B5%E0%B9%88%E0%B8%8A%E0%B8%99%E0%B8%B4%E0%B8%94

ศักดิ์ทองจีน, พ. ร. (2021). วัคซีนโควิด-19 มีกี่ชนิด อะไรบ้าง.

https://www.synphaet.co.th/%E0%B8%A7%E0%B8%B1%E0%B8%84%E0%B8%8B%E0%B8%B5%E0%B8%99%E0%B9%82%E0%B8%84%E0%B8%A7%E0%B8%B4%E0%B8%94-19-%E0%B8%A1%E0%B8%B5%E0%B8%81%E0%B8%B5%E0%B9%88%E0%B8%8A%E0%B8%99%E0%B8%B4%E0%B8%94-%E0%B8%AD/

ภาคผนวก

รายละเอียดโค้ดการตรวจสอบข้อมูลและสำรวจข้อมูลเบื้องต้น (Exploratory Data Analysis)

```
from google.colab import drive
drive.mount('/content/drive')

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import json
from collections import Counter
warnings.simplefilter(action='ignore')
pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)
plt.rcParams.update({'figure.max_open_warning': 0})

train = pd.read_json('/content/drive/My Drive/IS/train.json', lines=True)
test = pd.read_json('/content/drive/My Drive/IS/test.json', lines=True)

print('train shapes: ', train.shape)
print('Test shapes: ', test.shape)

train.head(5)
test.head(5)

train.head(3)
train.dtypes.value_counts().plot.bar()

train.info()

train.describe()

test.info()

train.isna().sum()
print('train shape :',train.shape)
print('test shape :',test.shape)

train.head()
test.head()
train.describe()

fig, axe = plt.subplots(1, 2, figsize=(10, 5))
sns.countplot(train['SN_filter'], palette='magma',ax=axe[1])
sns.distplot(train['signal_to_noise'],color='pink',ax=axe[0])


axe[0].set_title('Signal/Noise ')
axe[1].set_title('SN_Filter');


Run Cell | Run Below | Debug cell
# %%
plt.figure(figsize=(10,4))
sns.countplot(test['seq_length'],color='blue')


Run Cell | Run Above | Debug cell
# %%
plt.figure(figsize=(15,2))
sns.boxplot(train['signal_to_noise'],color='yellow',x='signal_to_noise')

plt.show()


Run Cell | Run Above | Debug cell
# %%
avg_reactivity = np.array(list(map(np.array,train.reactivity))).mean(axis=0)
avg_deg_50C = np.array(list(map(np.array,train.deg_50C))).mean(axis=0)
avg_deg_pH10 = np.array(list(map(np.array,train.deg_pH10))).mean(axis=0)

avg_deg_Mg_50C = np.array(list(map(np.array,train.deg_Mg_50C))).mean(axis=0)
avg_deg_Mg_pH10 = np.array(list(map(np.array,train.deg_Mg_pH10))).mean(axis=0)
```

```
fig, ax = plt.subplots(1,3,figsize=(15,4))

# Distribution of Reactivity Averaged over position
sns.kdeplot(avg_reactivity,color='red',ax=ax[0])
ax[0].set_title('Average Reactivity',size=10)

# Distribution of deg_50C Averaged over position
sns.kdeplot(avg_deg_50C,color='blue',ax=ax[1])
ax[1].set_title('Average deg_50C',size=10)

# Distribution of deg_pH10 Averaged over position
sns.kdeplot(avg_deg_pH10,color='green',ax=ax[2])
ax[2].set_title('Average deg_pH10',size=10)


plt.show()


Run Cell | Run Above | Debug cell
# %%
plt.figure(figsize=(15,8))

sns.lineplot(x=range(68),color='red',y=avg_reactivity,label='reactivity')
sns.lineplot(x=range(68),color='blue',y=avg_deg_50C,label='deg_50C')
sns.lineplot(x=range(68),color='green',y=avg_deg_pH10,label='deg_ph10')

plt.xlabel('Positions')
plt.xticks(range(0,68,2))
plt.ylabel('Values')
plt.title('Average Target Values (without Mg) and Positions')

plt.show()
```

```
plt.figure(figsize=(15,5))

sns.lineplot(x=range(68),y=avg_deg_Mg_50C,label='deg_Mg_50C')
sns.lineplot(x=range(68),y=avg_deg_Mg_pH10,label='deg_Mg_pH10')

plt.xlabel('Positions')
plt.xticks(range(0,68,2))
plt.ylabel('Values')
plt.title('Average Target Values (w Mg) and Positions')

plt.show()


Run Cell | Run Above | Debug cell
# %%
One_sample = train.iloc[0]
One_sequence = One_sample["sequence"]
One_structure = One_sample["structure"]
One_predicted_loop_type = One_sample["predicted_loop_type"]
One_reactivity = One_sample["reactivity"]


Run Cell | Run Above | Debug cell
# %%
print("One_sequence :",One_sequence)
print("One_structure :",One_structure)
print("One_predicted_loop_type :",One_predicted_loop_type)
print("One_reactivity :", One_reactivity)
```

```
Counter(One_sequence)


Run Cell | Run Above | Debug cell
# %%
sum_one_sequence = sum(Counter(One_sequence).values())
sum_one_sequence


Run Cell | Run Above | Debug cell
# %%
Counter(One_structure)


Run Cell | Run Above | Debug cell
# %%
sum_one_structure = sum(Counter(One_structure).values())
sum_one_structure


Run Cell | Run Above | Debug cell
# %%
Counter(One_predicted_loop_type)


Run Cell | Run Above | Debug cell
# %%
sum_one_predicted_loop_type = sum(Counter(One_predicted_loop_type).values())
sum_one_predicted_loop_type


Run Cell | Run Above | Debug cell
# %%
print(f"Samples with signal_to_noise greater than 1: {len(train.loc[(train['signal_to_noise'] > 1 )])}")
print(f"Samples with SN_filter = 1: {len(train.loc[(train['SN_filter'] == 1 )])}")
print(f"Samples with signal_to_noise greater than 1, but SN_filter == 0: {len(train.loc[(train['signal_to_noise'] > 1) &
```

รายละเอียดการทำแบบจำลอง XG boost

```
from google.colab import drive
drive.mount('/content/drive')

import json

import numpy as np
import pandas as pd

from sklearn.multioutput import MultiOutputRegressor
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import make_scorer

from xgboost import XGBRegressor


Run Cell | Run Below | Debug cell
# %%
train_df = pd.read_json('/content/drive/My Drive/IS/train.json', lines=True)
test_df = pd.read_json('/content/drive/My Drive/IS/test.json', lines=True)
sample_sub_df = pd.read_csv('/content/drive/My Drive/IS/sample_submission.csv')


Run Cell | Run Above | Debug cell
# %%
print(train_df.shape)
print(test_df.shape)
print(sample_sub_df.shape)


Run Cell | Run Above | Debug cell
# %%
train_df.head(3)


Run Cell | Run Above | Debug cell
# %%
test_df.head(3)
test_df.head(3)


Run Cell | Run Above | Debug cell
# %%
sample_sub_df.head(3)


Run Cell | Run Above | Debug cell
# %%
# Calculate Means of targets
train_df['reactivity'] = train_df['reactivity'].apply(lambda x: np.mean(x))
train_df['deg_Mg_pH10'] = train_df['deg_Mg_pH10'].apply(lambda x: np.mean(x))
train_df['deg_pH10'] = train_df['deg_pH10'].apply(lambda x: np.mean(x))
train_df['deg_Mg_50C'] = train_df['deg_Mg_50C'].apply(lambda x: np.mean(x))
train_df['deg_50C'] = train_df['deg_50C'].apply(lambda x: np.mean(x))


Run Cell | Run Above | Debug cell
# %%
train_df.head()


Run Cell | Run Above | Debug cell
# %%
# Drop unnecessary columns for now
train_df = train_df.drop(['id', 'index', 'reactivity_error', 'deg_error_Mg_pH10', 'deg_error_pH10', 'deg_error_Mg_50C', 'deg_error_50C',
train_df.head()


Run Cell | Run Above | Debug cell
# %%
# Split data in features and labels
X_train = train_df.drop(['reactivity', 'deg_Mg_pH10', 'deg_Mg_50C'], axis=1)
Y_train = train_df[['reactivity', 'deg_Mg_pH10', 'deg_Mg_50C']]
```

```
X_train, X_test, Y_train, Y_test = train_test_split(X_train, Y_train, test_size=0.15)
X_train.shape, X_test.shape, Y_train.shape, Y_test.shape


Run Cell | Run Above | Debug cell
# %%
def featurize(df):

    df['total_A_count'] = df['sequence'].apply(lambda s: s.count('A'))
    df['total_G_count'] = df['sequence'].apply(lambda s: s.count('G'))
    df['total_U_count'] = df['sequence'].apply(lambda s: s.count('U'))
    df['total_C_count'] = df['sequence'].apply(lambda s: s.count('C'))

    df['total_dot_count'] = df['structure'].apply(lambda s: s.count('.'))
    df['total_ob_count'] = df['structure'].apply(lambda s: s.count('('))
    df['total_cb_count'] = df['structure'].apply(lambda s: s.count(')'))

    return df


Run Cell | Run Above | Debug cell
# %%
X_train = featurize(X_train)
X_test = featurize(X_test)


Run Cell | Run Above | Debug cell
# %%
X_train = X_train.drop(['sequence', 'structure', 'predicted_loop_type'], axis=1)
X_test = X_test.drop(['sequence', 'structure', 'predicted_loop_type'], axis=1)


Run Cell | Run Above | Debug cell
# %%
X_train.head()
 scaler = StandardScaler()
 scaler.fit(X_train)

 X_train = scaler.transform(X_train)
 X_test = scaler.transform(X_test)


Run Cell | Run Above | Debug cell
# %%
def mcrmse_loss(y_true, y_pred, N=3):
    """
    Calculates competition eval metric
    """
    assert len(y_true) == len(y_pred)
    n = len(y_true)
    return np.sum(np.sqrt(np.sum((y_true - y_pred)**2, axis=0)/n)) / N

custom_scorer = make_scorer(mcrmse_loss, greater_is_better=False)


Run Cell | Run Above | Debug cell
# %%
# Hyperparameter tune multioutput XGBoost Regressor
xgb = XGBRegressor(
    subsample=0.8,
    colsample_bytree=0.75,
    reg_lambda=2,
    reg_alpha=1,
    random_state=28
)

params = {
    'estimator__n_estimators': [ 800, 900, 1000],
    'estimator__learning_rate': [0.1, 0.25, 0.30],
    'estimator__max_depth': [3, 4, 5]
}
```

```
reg = MultiOutputRegressor(xgb)

# Perform 5 fold cross validation on set 1 features
gs = GridSearchCV(reg, param_grid=params, cv=5, return_train_score=True , n_jobs=-1, sc
gs.fit(X_train, Y_train)


Run Cell | Run Above | Debug cell
# %%
gs.best_params_


Run Cell | Run Above | Debug cell
# %%
# Train using best parameters
xgb = XGBRegressor(
    max_depth=gs.best_params_['estimator__max_depth'],
    subsample=0.8,
    colsample_bytree=0.75,
    reg_lambda=2,
    reg_alpha=1,
    n_estimators=gs.best_params_['estimator__n_estimators'],
    learning_rate=gs.best_params_['estimator__learning_rate'],
    random_state=28
)

reg = MultiOutputRegressor(xgb)
reg.fit(X_train, Y_train)


Run Cell | Run Above | Debug cell
# %%
# Train score
mcrmse_loss(reg.predict(X_train), np.array(Y_train))


# Validation score
mcrmse_loss(reg.predict(X_test), np.array(Y_test))


Run Cell | Run Above | Debug cell
# %%
test = featurize(test_df.drop(['index', 'id'], axis=1))
test = test.drop(['sequence', 'structure', 'predicted_loop_type'], axis=1)
test = scaler.transform(test)


Run Cell | Run Above | Debug cell
# %%
# Predict
preds = pd.DataFrame(reg.predict(test))


Run Cell | Run Above | Debug cell
# %%
# Create submission csv
submission_df = preds.loc[preds.index.repeat(list(test_df['seq_length']))].reset_index(drop=True)
submission_df = submission_df.rename(columns={0: 'reactivity', 1: 'deg_Mg_pH10', 2: 'deg_Mg_50C'})
submission_df['id_seqpos'] = sample_sub_df['id_seqpos']
submission_df['deg_pH10'] = 0.0
submission_df['deg_50C'] = 0.0
submission_df = submission_df[['id_seqpos', 'reactivity', 'deg_Mg_pH10', 'deg_pH10', 'deg_Mg_50C', 'deg_50C']]


Run Cell | Run Above | Debug cell
# %%
submission_df.to_csv('base.csv', index=False)
```

รายละเอียดการทำแบบจำลอง Catboost

```python
from google.colab import drive
drive.mount('/content/drive')
pip install catboost
import json

import numpy as np
import pandas as pd

from sklearn.multioutput import MultiOutputRegressor
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import make_scorer

from catboost import CatBoostRegressor

train_df= pd.read_json('/content/drive/My Drive/IS/train.json', lines=True)
test_df= pd.read_json('/content/drive/My Drive/IS/test.json', lines=True)
sample_sub_df= pd.read_csv('/content/drive/My Drive/IS/sample_submission.csv')
print(train_df.shape)
print(test_df.shape)
print(sample_sub_df.shape)
train_df.head(3)
test_df.head(3)
sample_sub_df.head(3)
# Calculate Means of targets
train_df['reactivity'] = train_df['reactivity'].apply(lambda x: np.mean(x))
train_df['deg_Mg_pH10'] = train_df['deg_Mg_pH10'].apply(lambda x: np.mean(x))
train_df['deg_pH10'] = train_df['deg_pH10'].apply(lambda x: np.mean(x))
train_df['deg_Mg_50C'] = train_df['deg_Mg_50C'].apply(lambda x: np.mean(x))
train_df['deg_50C'] = train_df['deg_50C'].apply(lambda x: np.mean(x))
train_df.head()
# Drop unnecessary columns for now
train_df = train_df.drop(['id', 'index', 'reactivity_error', 'deg_error_Mg_pH10', 'deg_error_pH10', 'deg_error_Mg_50C', 'deg_error_50C', 'SN_filter', 'signal_to_noise', '
train_df.head()
# Split data in features and labels
X_train = train_df.drop(['reactivity', 'deg_Mg_pH10', 'deg_Mg_50C'], axis=1)
Y_train = train_df[['reactivity', 'deg_Mg_pH10', 'deg_Mg_50C']]
X_train, X_test, Y_train, Y_test = train_test_split(X_train, Y_train, test_size=0.15)
X_train.shape, X_test.shape, Y_train.shape, Y_test.shape

def featurize(df):

    df['total_A_count'] = df['sequence'].apply(lambda s: s.count('A'))
    df['total_G_count'] = df['sequence'].apply(lambda s: s.count('G'))
    df['total_U_count'] = df['sequence'].apply(lambda s: s.count('U'))
    df['total_C_count'] = df['sequence'].apply(lambda s: s.count('C'))
```

```
    df['total_dot_count'] = df['structure'].apply(lambda s: s.count('.'))
    df['total_ob_count'] = df['structure'].apply(lambda s: s.count('('))
    df['total_cb_count'] = df['structure'].apply(lambda s: s.count(')'))

    return df

X_train = featurize(X_train)
X_test = featurize(X_test)

X_train = X_train.drop(['sequence', 'structure', 'predicted_loop_type'], axis=1)
X_test = X_test.drop(['sequence', 'structure', 'predicted_loop_type'], axis=1)

X_train.head()

scaler = StandardScaler()
scaler.fit(X_train)

X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)

def mcrmse_loss(y_true, y_pred, N=3):
    """
    Calculates competition eval metric
    """
    assert len(y_true) == len(y_pred)
    n = len(y_true)
    return np.sum(np.sqrt(np.sum((y_true - y_pred)**2, axis=0)/n)) / N

custom_scorer = make_scorer(mcrmse_loss, greater_is_better=False)

cbr = CatBoostRegressor(random_state=28)
params = {
    'estimator__n_estimators': [ 300, 500, 800],
    'estimator__learning_rate': [0.02, 0.05, 0.1],
    'estimator__max_depth': [3, 6, 8]
}
reg = MultiOutputRegressor(cbr)
# Perform 5 fold cross validation on set 1 features
cb = GridSearchCV(estimator=reg, param_grid = params, cv = 5, n_jobs=-1, scoring=custom_scorer)

cb.fit(X_train,Y_train)

cb.best_params_

 # Train using best parameters
 cbr = CatBoostRegressor(
     depth=cb.best_params_['estimator__max_depth'],
     n_estimators=cb.best_params_['estimator__n_estimators'],
     learning_rate=cb.best_params_['estimator__learning_rate'],
     random_state=28
 )
 reg = MultiOutputRegressor(cbr)
 reg.fit(X_train, Y_train)

 # Train score
 mcrmse_loss(reg.predict(X_train), np.array(Y_train))

 # Validation score
 mcrmse_loss(reg.predict(X_test), np.array(Y_test))

 test = featurize(test_df.drop(['index', 'id'], axis=1))
 test = test.drop(['sequence', 'structure', 'predicted_loop_type'], axis=1)
 test = scaler.transform(test)
 # Predict
 preds = pd.DataFrame(reg.predict(test))
```

รายละเอียดการทำแบบจำลอง Random Forest

```python
from google.colab import drive
drive.mount('/content/drive')

import json

import numpy as np
import pandas as pd

from sklearn.multioutput import MultiOutputRegressor
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import make_scorer

from sklearn.ensemble import RandomForestRegressor

train_df = pd.read_json('/content/drive/My Drive/IS/train.json', lines=True)
test_df = pd.read_json('/content/drive/My Drive/IS/test.json', lines=True)
sample_sub_df = pd.read_csv('/content/drive/My Drive/IS/sample_submission.csv')

print(train_df.shape)
print(test_df.shape)
print(sample_sub_df.shape)

train_df.head(3)

test_df.head(3)

# Calculate Means of targets
train_df['reactivity'] = train_df['reactivity'].apply(lambda x: np.mean(x))
train_df['deg_Mg_pH10'] = train_df['deg_Mg_pH10'].apply(lambda x: np.mean(x))
train_df['deg_pH10'] = train_df['deg_pH10'].apply(lambda x: np.mean(x))
train_df['deg_Mg_50C'] = train_df['deg_Mg_50C'].apply(lambda x: np.mean(x))
train_df['deg_50C'] = train_df['deg_50C'].apply(lambda x: np.mean(x))

train_df.head(3)

# Drop unnecessary columns for now
train_df = train_df.drop(['id', 'index', 'reactivity_error', 'deg_error_Mg_pH10', 'deg_error_pH10', 'deg_error_Mg_50C', 'deg_error_50C', 'SN_filter', 'signal_to_noise', 'deg_
train_df.head()

# Split data in features and labels
X_train = train_df.drop(['reactivity', 'deg_Mg_pH10', 'deg_Mg_50C'], axis=1)
Y_train = train_df[['reactivity', 'deg_Mg_pH10', 'deg_Mg_50C']]

X_train, X_test, Y_train, Y_test = train_test_split(X_train, Y_train, test_size=0.15)
X_train.shape, X_test.shape, Y_train.shape, Y_test.shape
```

```
def featurize(df):

    df['total_A_count'] = df['sequence'].apply(lambda s: s.count('A'))
    df['total_G_count'] = df['sequence'].apply(lambda s: s.count('G'))
    df['total_U_count'] = df['sequence'].apply(lambda s: s.count('U'))
    df['total_C_count'] = df['sequence'].apply(lambda s: s.count('C'))

    df['total_dot_count'] = df['structure'].apply(lambda s: s.count('.'))
    df['total_ob_count'] = df['structure'].apply(lambda s: s.count('('))
    df['total_cb_count'] = df['structure'].apply(lambda s: s.count(')'))

    return df

X_train = featurize(X_train)
X_test = featurize(X_test)

X_train = X_train.drop(['sequence', 'structure', 'predicted_loop_type'], axis=1)
X_test = X_test.drop(['sequence', 'structure', 'predicted_loop_type'], axis=1)

X_train.head()

scaler = StandardScaler()
scaler.fit(X_train)

X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)

def mcrmse_loss(y_true, y_pred, N=3):
    """
    Calculates competition eval metric
    """
    assert len(y_true) == len(y_pred)
    n = len(y_true)
    return np.sum(np.sqrt(np.sum((y_true - y_pred)**2, axis=0)/n)) / N

custom_scorer = make_scorer(mcrmse_loss, greater_is_better=False)

rf = RandomForestRegressor(random_state=42,verbose=1)
params = {
    'estimator__n_estimators': [ 100, 200, 300],
    'estimator__max_depth': [8, 9, 10],

}
reg = MultiOutputRegressor(rf)
# Perform 5 fold cross validation on set 1 features
rfr = GridSearchCV(estimator=reg, param_grid = params, cv = 5, n_jobs=-1, scoring=custom_scorer)
```

```
rfr.fit(X_train,Y_train)

rfr.best_params_

# Train using best parameters
rf = RandomForestRegressor(n_jobs=-1, random_state=28,verbose=1, n_estimators=500, max_depth=13)

rfr = MultiOutputRegressor(rf)
rfr.fit(X_train, Y_train)

# Train score
mcrmse_loss(rfr.predict(X_train), np.array(Y_train))

# Validation score
mcrmse_loss(rfr.predict(X_test), np.array(Y_test))

test = featurize(test_df.drop(['index', 'id'], axis=1))
test = test.drop(['sequence', 'structure', 'predicted_loop_type'], axis=1)
test = scaler.transform(test)

# Predict
preds = pd.DataFrame(rfr.predict(test))
```

รายละเอียดการทำแบบจำลอง Ligth GBM

```
from google.colab import drive
drive.mount('/content/drive')

import gc
import os
import random

import lightgbm as lgb
import numpy as np
import pandas as pd
import seaborn as sns
import itertools

from matplotlib import pyplot as plt
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import StratifiedKFold, KFold, GroupKFold
from sklearn.cluster import KMeans

sns.set(style='darkgrid')
SEEDS = 28
Run Cell | Run Below | Debug cell
# %%
def rmse(y_true, y_pred):
    return (mean_squared_error(y_true, y_pred))** .5

# treemodel_wrapper
class TreeModel:
    def __init__(self, model_type):
        self.model_type = model_type
        self.tr_data = None
        self.vl_data = None
        self.model = None

    def train(self, params, train_x, train_y, valid_x=None, valid_y=None, num_round=None, early_stopping=None, verbose=None):
        if self.model_type == 'lgb':
            self.tr_data = lgb.Dataset(train_x, label=train_y)
            self.vl_data = lgb.Dataset(valid_x, label=valid_y)
            self.model = lgb.train(params, self.tr_data, valid_sets=[self.tr_data, self.vl_data],
                                   num_boost_round=num_round, early_stopping_rounds=early_stopping,verbose_eval=verbose)

        if self.model_type == 'rf_reg':
            self.train_x = train_x
            self.train_y = train_y
            self.model = RandomForestRegressor(**params).fit(self.train_x, self.train_y)
      if self.model_type == 'xgb':
          self.tr_data = xgb.DMatrix(train_x, train_y)
          self.vl_data = xgb.DMatrix(valid_x, valid_y)
          self.model = xgb.train(params, self.tr_data, num_boost_round=num_round,
                                 evals=[(self.tr_data, 'train'), (self.vl_data, 'val')],
                                 verbose_eval=verbose, early_stopping_rounds=early_stopping)

      if self.model_type == 'cat':
          params['num_boost_round'] = num_round
          self.cat_cols = list(train_x.select_dtypes(include='object').columns)
          self.tr_data = Pool(train_x, train_y, cat_features=self.cat_cols)
          self.vl_data = Pool(valid_x, valid_y, cat_features=self.cat_cols)
          self.model = CatBoost(params).fit(self.tr_data, eval_set=self.vl_data,
                                            early_stopping_rounds=early_stopping, verbose=verbose, use_best_model=True)

          return self.model


  def predict(self,X):
      if self.model_type == 'lgb':
          return self.model.predict(X, num_iteration=self.model.best_iteration)

      if self.model_type == 'rf_reg':
          return self.model.predict(X)

      if self.model_type == 'xgb':
          X_DM = xgb.DMatrix(X)
          return self.model.predict(X_DM)

      if self.model_type == 'cat':
          X_pool = Pool(X, cat_features=self.cat_cols)
          return self.model.predict(X_pool)

  @property
  def feature_names_(self):
      if self.model_type == 'lgb':
          return self.model.feature_name()

      if self.model_type == 'rf_reg':
          return self.train_x.columns

      if self.model_type == 'xgb':
          return list(self.model.get_score(importance_type='gain').keys())

      if self.model_type == 'cat':
          return self.model.feature_names_
```

```
    def feature_importances_(self):
        if self.model_type == 'lgb':
            return self.model.feature_importance(importance_type='gain')

        if self.model_type == 'rf_reg':
            return self.model.feature_importances_

        if self.model_type == 'xgb':
            return list(self.model.get_score(importance_type='gain').values())

        if self.model_type == 'cat':
            return self.model.feature_importances_


train= pd.read_json('/content/drive/My Drive/IS/train.json', lines=True)
test= pd.read_json('/content/drive/My Drive/IS/test.json', lines=True)
submission= pd.read_csv('/content/drive/My Drive/IS/sample_submission.csv')

train_data = []
for mol_id in train['id'].unique():
    sample_data = train.loc[train['id'] == mol_id]
    sample_seq_length = sample_data.seq_length.values[0]

    for i in range(68):
        sample_dict = {'id' : sample_data['id'].values[0],
                       'id_seqpos' : sample_data['id'].values[0] + '_' + str(i),
                       'sequence' : sample_data['sequence'].values[0][i],
                       'structure' : sample_data['structure'].values[0][i],
                       'predicted_loop_type' : sample_data['predicted_loop_type'].values[0][i],
                       'reactivity' : sample_data['reactivity'].values[0][i],
                       'reactivity_error' : sample_data['reactivity_error'].values[0][i],
                       'deg_Mg_pH10' : sample_data['deg_Mg_pH10'].values[0][i],
                       'deg_error_Mg_pH10' : sample_data['deg_error_Mg_pH10'].values[0][i],
                       'deg_pH10' : sample_data['deg_pH10'].values[0][i],
                       'deg_error_pH10' : sample_data['deg_error_pH10'].values[0][i],
                       'deg_Mg_50C' : sample_data['deg_Mg_50C'].values[0][i],
                       'deg_error_Mg_50C' : sample_data['deg_error_Mg_50C'].values[0][i],
                       'deg_50C' : sample_data['deg_50C'].values[0][i],
                       'deg_error_50C' : sample_data['deg_error_50C'].values[0][i]}

        shifts = [1,2,3,4,5]
        shift_cols = ['sequence', 'structure', 'predicted_loop_type']
        for shift,col in itertools.product(shifts, shift_cols):
            if i - shift >= 0:
                sample_dict['b'+str(shift)+'_'+col] = sample_data[col].values[0][i-shift]
            else:
                sample_dict['b'+str(shift)+'_'+col] = -1

            if i + shift <= sample_seq_length - 1:
                sample_dict['a'+str(shift)+'_'+col] = sample_data[col].values[0][i+shift]
            else:
                sample_dict['a'+str(shift)+'_'+col] = -1


        train_data.append(sample_dict)
train_data = pd.DataFrame(train_data)
train_data.head()

test_data = []
for mol_id in test['id'].unique():
    sample_data = test.loc[test['id'] == mol_id]
    sample_seq_length = sample_data.seq_length.values[0]
    for i in range(sample_seq_length):
        sample_dict = {'id' : sample_data['id'].values[0],
                       'id_seqpos' : sample_data['id'].values[0] + '_' + str(i),
                       'sequence' : sample_data['sequence'].values[0][i],
                       'structure' : sample_data['structure'].values[0][i],
                       'predicted_loop_type' : sample_data['predicted_loop_type'].values[0][i]}

        shifts = [1,2,3,4,5]
        shift_cols = ['sequence', 'structure', 'predicted_loop_type']
        for shift,col in itertools.product(shifts, shift_cols):
            if i - shift >= 0:
                sample_dict['b'+str(shift)+'_'+col] = sample_data[col].values[0][i-shift]
            else:
                sample_dict['b'+str(shift)+'_'+col] = -1

            if i + shift <= sample_seq_length - 1:
                sample_dict['a'+str(shift)+'_'+col] = sample_data[col].values[0][i+shift]
            else:
                sample_dict['a'+str(shift)+'_'+col] = -1

        test_data.append(sample_dict)
test_data = pd.DataFrame(test_data)
test_data.head()
```

```
# label_encoding
sequence_encmap = {'A': 0, 'G' : 1, 'C' : 2, 'U' : 3}
structure_encmap = {'.' : 0, '(' : 1, ')' : 2}
looptype_encmap = {'S':0, 'E':1, 'H':2, 'I':3, 'X':4, 'M':5, 'B':6}

enc_targets = ['sequence', 'structure', 'predicted_loop_type']
enc_maps = [sequence_encmap, structure_encmap, looptype_encmap]

for t,m in zip(enc_targets, enc_maps):
    for c in [c for c in train_data.columns if t in c]:
        train_data[c] = train_data[c].replace(m)
        test_data[c] = test_data[c].replace(m)


not_use_cols = ['id', 'id_seqpos']
features = [c for c in test_data.columns if c not in not_use_cols]
targets = ['reactivity', 'deg_Mg_pH10', 'deg_pH10', 'deg_Mg_50C', 'deg_50C']

FOLD_N = 5
gkf = GroupKFold(n_splits=FOLD_N)

params = {'objective': 'regression',
          'boosting': 'gbdt',
          'metric': 'rmse',
          'learning_rate': 0.05,
          'seed' : SEEDS}

feature_importances = pd.DataFrame()
result = {}
oof_df = pd.DataFrame(train_data.id_seqpos)

for target in targets:
    oof = pd.DataFrame()
    preds = np.zeros(len(test_data))
    scores = 0.0

    for n, (tr_idx, vl_idx) in enumerate(gkf.split(train_data[features], train_data['reactivity'], train_data['id'])):
        tr_x, tr_y = train_data[features].iloc[tr_idx], train_data[target].iloc[tr_idx]
        vl_x, vl_y = train_data[features].iloc[vl_idx], train_data[target].iloc[vl_idx]
        vl_id = train_data['id_seqpos'].iloc[vl_idx]

        model = TreeModel(model_type='lgb')
        model.train(params, tr_x, tr_y, vl_x, vl_y,
                    num_round=20000, early_stopping=100,verbose=0)

        fi_tmp = pd.DataFrame()
        fi_tmp['feature'] = model.feature_names_
        fi_tmp['importance'] = model.feature_importances_
        fi_tmp['fold'] = n
        fi_tmp['target'] = target
        feature_importances = feature_importances.append(fi_tmp)

        vl_pred = model.predict(vl_x)
        score = rmse(vl_y, vl_pred)
        scores += score / FOLD_N
        print(f'score : {score}')

        oof = oof.append(pd.DataFrame({'id_seqpos':vl_id, target:vl_pred}))

        pred = model.predict(test_data[features])
        preds += pred / FOLD_N

    oof_df = oof_df.merge(oof, on='id_seqpos', how='inner')
    submission[target] = preds

    print(f'{target}_rmse : {scores}')
    result[target] = scores

display(result)
display(f'total : {np.mean(list(result.values()))}')

# feature_importances
for target in targets:
    tmp = feature_importances[feature_importances.target==target]
    order = list(tmp.groupby('feature').mean().sort_values('importance', ascending=False).index)

    plt.figure(figsize=(10, 10))
    sns.barplot(x="importance", y="feature", data=tmp, order=order)
    plt.title(target)
    plt.tight_layout()
```

ประวัติผู้เขียน

ชื่อ-สกุล	ณัฐนรี พอสูงเนิน
วัน เดือน ปี เกิด	01 เมษายน 2533
สถานที่เกิด	กรุงเทพมหานคร
วุฒิการศึกษา	พ.ศ.2555 วิทยาศาสตรบัณฑิต สาขาวิชาเคมี จากมหาวิทยาลัยธรรมศาสตร์