



แอปพลิเคชันช่วยประเมินโรคจากอาการไอ

APPLICATION TO HELP ASSESS DISEASE FROM COUGHING

เกรียงศักดิ์ ศรีสุวรรณ

จอนแนชั่น ดอร์วิเนน

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิศวกรรมอิเล็กทรอนิกส์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

วิทยาเขตชุมพรเขตรอุดมศักดิ์ จังหวัดชุมพร

ปีการศึกษา 2564

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้คัดลอกเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ปีการศึกษา2564

หลักสูตรวิศวกรรมอิเล็กทรอนิกส์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง วิทยาเขตชุมพรเขตรอุดมศักดิ์ จังหวัดชุมพร

เรื่อง แอปพลิเคชันช่วยประเมินโรคจากอาการไอ

APPLICATION TO HELP ASSESS DISEASE FROM COUGHING

ผู้จัดทำ

1.นายเกรียงศักดิ์ ศรีสุวรรณ รหัสนักศึกษา 61511012

2.นายจอนแนซัน ดอร์วิเนน รหัสนักศึกษา 61511013


.....อาจารย์ที่ปรึกษา
(ผศ.ดร.มนตรี ไชยชาญยุทธ์)



COPYRIGHT 2021

DEPARTMENT OF ENGINEERING

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

PRINCE OF CHUMPHON

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ชื่อปริญญาบัตร	แอปพลิเคชันช่วยประเมินโรคจากอาการไอ	
นักศึกษา	นายเกรียงศักดิ์ ศรีสุวรรณ	รหัสนักศึกษา 61511012
	นายจอนแนซัน ดอร์วินน	รหัสนักศึกษา 61511013
อาจารย์ที่ปรึกษา	ผศ.ดร.มนตรี ไชยชาญยุทธ์	
หลักสูตร	วิศวกรรมศาสตรบัณฑิต	
สาขาวิชา	วิศวกรรมอิเล็กทรอนิกส์	
ปีการศึกษา	2564	

บทคัดย่อ

ปริญญาบัตรฉบับนี้ได้กล่าวถึงการใช้ค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมล เพื่อสร้างแอปพลิเคชันช่วยประเมินโรคจากอาการไอ อาศัยหลักการเปรียบเทียบค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลของเสียงที่รับเข้ากับฟิจเจอร์ โดยนำไฟล์เสียงไอคนที่เป็โรคและเสียงไอคนที่เป็ปกติสร้างฟิจเจอร์และทำการประเมินความตรงกันของเสียงที่รับเข้ากับฟิจเจอร์เป็เปอร์เซ็นต์ ซึ่งออกแบบโดยใช้ภาษาไพทอน สร้างชุดคำสั่ง และภาษาซีวี ในการสร้างหน้าแอปพลิเคชัน

การทดลองใช้แอปพลิเคชันช่วยประเมินโรคจากอาการไอ สามารถทำงานบนระบบปฏิบัติการวินโดว โดยนำไฟล์เสียงไอคนที่ติดโรคโควิด-19กับเสียงไอของคนที่เป็ปกติ ขนาดเวลาต่างกัน และจำนวนไฟล์ต่างกัน มาเป็ฟิจเจอร์ ส่งผลต่อความแม่นยำในการประเมินโรค การนำไฟล์เสียงไอขนาดมากกว่า 1 วินาที มาเป็ฟิจเจอร์ส่งผลให้ค่าความแม่นยำในการทำนายผลลดลงในการทดลองทำนายผลไฟล์เสียงไอโดยนำไฟล์ขนาดหนึ่งวินาทีไฟล์เสียงไอคนที่เป็โควิด 53 ไฟล์ ไฟล์เสียงไอคนที่เป็ปกติ 191 เป็ฟิจเจอร์ เมื่อนำไฟล์เสียงที่ไม่ใช่เสียงไอมาทดสอบได้ผลลัพธ์ว่าไม่ใช่เสียงไอ มีค่าความแม่นยำในการทำนายผลอยู่ที่ 100 เปอร์เซ็นต์ เมื่อนำไฟล์เสียงไอคนที่เป็โควิดมาผ่านการประมวลผล มีค่าความแม่นยำการทำนายผลอยู่ที่ 25 เปอร์เซ็นต์ และเมื่อนำไฟล์เสียงไอของคนที่ปกติมาใช้ในการประมวลผล มีค่าความแม่นยำอยู่ที่ 77.78 เปอร์เซ็นต์ เนื่องจากมีจำนวนไฟล์เสียงไอโควิดน้อยเกินไป ทำให้ฟิจเจอร์ที่ได้ไม่ครอบคลุมกับการทดสอบ

Project Title	APPLICATION TO HELP ASSESS DISEASE FROM COUGHING	
Student	Mr. Kriangsak Srisuwan	ID 61511012
	Mr. Jonnathan Dorvinan	ID 61511013
Advisor	Asst Prof. Dr. Montree Chichanyut	
Education Level	Bachelor of Engineering	
Program in	Electronics Engineering	
Academic Year	2021	

ABSTRACT

This thesis discusses the use of Mel frequency cepstrum coefficient. to create an application to help assess diseases from cough Based on the principle of comparing the Mel frequency cepstrum coefficient of the sound received with the features. By using the sound file of coughing people who are sick and coughing people who are normal to create a feature. and assessed the synchronization of the sound received with the feature as a percentage which is designed using Python Create a set of instructions and Kivy language to create application pages.

The trial of application to helps assess disease from coughing. It can run by bringing the audio file of Covid-19 coughing with the sound of normal coughing. Different time scale and different number of files. Become a feature affect the accuracy of disease assessment Incorporating more than 1 second. A feature resulted in lower prediction accuracy. In the AI audio file prediction experiment using 53 and normal cough 191 sound file to build a feature. When a non-cough audio file was tested, the result was non-cough. It had a prediction accuracy of 100 percent when the audio files of people who are infected with Covid-19 were processed. It has a predictive accuracy of 25 percent and when a normal human cough audio file is used for processing. It had an accuracy of 77.78 percent due to the low number of Covid cough audio files. This's causing the features that are not covered by the test.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงได้เป็นอย่างดี ด้วยความช่วยเหลือ และการสนับสนุนจากบุคคล
หลายๆท่าน ซึ่งผู้เขียนขอขอบคุณทุกๆ ท่านดังต่อไปนี้

ขอขอบพระคุณคุณพ่อและคุณแม่ ผู้ซึ่งคอยให้การอบรมสั่งสอน เลี้ยงดู สนับสนุนการศึกษา
ตลอดจนให้กำลังใจเสมอมา

ขอขอบพระคุณ ผศ.ดร.มนตรี ไชยชาญยุทธ์ อาจารย์ที่ปรึกษา ผู้ซึ่งให้คำแนะนำต่างๆ ในการทำ
โครงการ และติดตามเกี่ยวกับงานโครงการตลอดมา ผู้เขียนรู้สึกซาบซึ้งในความเมตตาของท่านจึงขอกราบ
ขอบพระคุณเป็นอย่างสูง

ขอขอบคุณ อาจารย์ณภัสรพี สิทธิวัจน์ ที่ได้ให้คำปรึกษาเกี่ยวกับการพัฒนาระบบและให้ความรู้
เกี่ยวกับพื้นฐานการเขียนโปรแกรมและให้คำแนะนำเกี่ยวกับวิธีการทำระบบของแอปพลิเคชัน

ขอขอบพระคุณคณะอาจารย์ที่เคารพทุกท่าน ที่ให้ความเอาใจใส่แนะนำ คอยช่วยเหลือเสมอมา
และต้องขอขอบคุณพี่ๆ เพื่อนๆ น้องๆ ที่คอยช่วยเหลือในการทำโครงการจนสำเร็จลุล่วงไปได้ด้วยดี

คุณค่าและประโยชน์อันพึงมีจากรายงานฉบับนี้ ผู้วิจัยขอมอบแด่ผู้มีพระคุณทุกท่าน

เกรียงศักดิ์ ศรีสุวรรณ

จอนแนชั่น ดอร์วิเนน

สารบัญ

หน้า

บทคัดย่อภาษาไทย	I
บทคัดย่อภาษาอังกฤษ.....	II
กิตติกรรมประกาศ.....	II
สารบัญ	IV
สารบัญตาราง	X
สารบัญรูป.....	XI
บทที่ 1 บทนำ.....	1
1.1 ความเป็นมาและความสำคัญของปัญหา.....	1
1.2 ความมุ่งหมายและวัตถุประสงค์.....	1
1.3 สมมุติฐานของการศึกษา.....	1
1.4 ขอบเขตของการศึกษา.....	2
1.5 ประโยชน์ที่ได้รับ.....	2
1.6 ขั้นตอนและแผนการดำเนินงาน.....	2
1.7 โครงสร้างปริญญานิพนธ์.....	4
บทที่ 2 ทฤษฎีที่เกี่ยวข้อง.....	5
2.1 โรคโควิด-19 (Covid-19).....	5
2.1.1 ระบาดวิทยาของเชื้อ.....	5
2.1.2 ลักษณะของโรค	6
2.1.3 ระยะฟักตัวของโรค.....	6
2.1.4 วิธีการแพร่ระบาดของโรค	6
2.1.5 อาการเมื่อติดเชื้อไวรัสโคโรนาสายพันธุ์ใหม่ หรือไวรัสโควิด-19.....	6
2.1.6 กลุ่มเสี่ยงติดเชื้อไวรัสโคโรนาสายพันธุ์ใหม่ หรือโควิด-19.....	7
2.1.7 หากมีอาการของโรคโควิด-19.....	7

สารบัญ(ต่อ)

	หน้า
2.1.8 วิธีป้องกันการติดเชื้อไวรัสโคโรนาสายพันธุ์ใหม่.....	7
2.1.9 ภาวะลองโควิด.....	8
2.2 การเรียนรู้เชิงลึก (Deep learning).....	8
2.3 การเรียนรู้เชิงลึกของเสียง.....	9
2.3.1 กระบวนการสร้างการเรียนรู้เชิงลึกของเสียง	9
2.4 การเรียนรู้ของเครื่อง.....	11
2.4.1 หลักการทำงานของการเรียนรู้ของเครื่อง	12
2.4.2 ข้อดีของการเรียนรู้ของเครื่อง.....	12
2.4.3 วิธีการที่ใช้ในการเรียนรู้ของเครื่อง.....	13
2.4.4 ประเภทต่างๆของการเรียนรู้ของเครื่อง.....	13
2.4.5 การเรียนรู้ของเครื่องกับแอปพลิเคชันที่ได้รับความนิยมสูงสุด.....	14
2.4.6 แอปพลิเคชันเพื่อการค้าของการเรียนรู้ของเครื่อง.....	15
2.4.7 การเรียนรู้ของเครื่องกับผู้นำเทคโนโลยี.....	15
2.5 วิชวลสตูดิโอโค้ด (Visual Studio Code).....	16
2.6 ภาษาไพทอน.....	17
2.7 เจสัน (Json)	17
2.8 ลิโบรซ่า (librosa).....	18
2.9 นัมพาย (NumPy).....	19
2.10 แมทซ์ (Math).....	19
2.11 อุปกรณ์รับเสียง (sounddevice).....	20
2.12 ซาวนด์ไฟล์ (soundfile).....	20
2.13 คีวี (Kivy).....	21
บทที่ 3 การออกแบบ	23
3.1 บล็อกไดอะแกรม	23

สารบัญ(ต่อ)

หน้า

3.1.1 บล็อกไดอะแกรมการออกแบบแอปพลิเคชันช่วยประเมินโรคจากอาการไอ	23
3.2 แผนผังการทำงานแอปพลิเคชัน.....	24
3.3 การออกแบบสร้างชุดข้อมูลสำหรับเทรน	26
3.3.1 สร้างโฟลเดอร์ของแอปพลิเคชัน	26
3.3.2 สร้างโฟลเดอร์สำหรับเสียงไอ.....	26
3.3.3 เขียนโปรแกรมสร้างชุดข้อมูลที่ใช้ในการเทรนด้วยภาษาไพทอน.....	28
3.4 การออกแบบชุดโปรแกรมสำหรับสร้างฟิจเจอร์.....	32
3.4.1 สร้างไฟล์สเกลไพทอนของโปรแกรมสร้างฟิจเจอร์.....	33
3.4.2 โลบาร์ที่จำเป็นในการสร้างฟิจเจอร์และชื่อไฟล์ที่ต้องการอ่านแล้วบันทึก	33
3.4.3 ชุดคำสั่งหลักในการสร้างฟิจเจอร์	34
3.4.4 ชุดคำสั่งอ่านค่าข้อมูลในชุดข้อมูล.....	35
3.4.5 ชุดคำสั่งสร้างขอบเขตของชุดข้อมูล.....	35
3.4.6 ชุดคำสั่งสร้างขีดจำกัดจากขอบเขตของชุดข้อมูล	36
3.4.7 ชุดคำสั่งหาความแม่นยำของฟิจเจอร์	39
3.4.8 ชุดคำสั่งบันทึกฟิจเจอร์.....	40
3.4.9 ชุดคำสั่งพล็อตกราฟฟิจเจอร์	41
3.5 การออกแบบแอปพลิเคชันและการทำงาน	42
3.5.1 สร้างไฟล์สเกลไพทอนของแอปพลิเคชัน.....	43
3.5.2 โลบาร์ที่จำเป็นในการสร้างแอปพลิเคชันและค่าที่ต้องกำหนด	43
3.5.3 คำสั่งหลักในการทำงานของแอปพลิเคชัน	44
3.5.4 ออกแบบสร้างหน้าต่างแอปพลิเคชัน.....	46
3.5.5 คำสั่งอ่านข้อมูลฟิจเจอร์ที่ได้เตรียมไว้.....	48
3.5.6 คำสั่งบันทึกเสียง.....	48
3.5.7 คำสั่งอ่านไฟล์เสียงที่ได้บันทึกเป็นไฟล์สกุล wav.....	49
3.5.8 คำสั่งประมวลผลเปรียบเทียบเสียงที่รับเข้ากับฟิจเจอร์	51

สารบัญ(ต่อ)

	หน้า
3.5.9 คำสั่งทำนายผล.....	52
3.5.10 คำสั่งการพล็อตกราฟเปรียบเทียบเสียงกับพีจเจอร์.....	53
3.5.11 เริ่มการทำงานแอปพลิเคชัน.....	53
บทที่ 4 การทดลองและผลการทดลอง.....	58
4.1 การทดลองแอปพลิเคชันบนคอมพิวเตอร์ระบบปฏิบัติการวินโดวส์ 11	58
4.1.1 วัตถุประสงค์	58
4.1.2 อุปกรณ์ในการทดลอง.....	58
4.1.3 วิธีการทดลอง	58
4.1.4 ผลการทดลอง.....	59
4.2 การทดลองหาความแม่นยำของพีจเจอร์จากการเทรน	62
4.2.1 วัตถุประสงค์	62
4.2.2 อุปกรณ์ในการทดลอง.....	62
4.2.3 วิธีการทดลอง	63
4.2.4 ผลการทดลอง.....	63
4.3 การทดลองเสียงที่มีลักษณะการไต่ต่างกัน	65
4.3.1 วัตถุประสงค์	65
4.3.2 อุปกรณ์ในการทดลอง.....	65
4.3.3 วิธีการทดลอง	65
4.3.4 ผลการทดลอง.....	65
4.4 การทดลองเสียงไอที่มีจำนวนลูกคลื่นต่างกัน	68
4.4.1 วัตถุประสงค์	68
4.4.2 อุปกรณ์ในการทดลอง.....	69
4.4.3 วิธีการทดลอง	69
4.4.4 ผลการทดลอง.....	69

สารบัญ(ต่อ)

	หน้า
4.5 การทดลองเสียงโอบแบบมีตัวกรองเสียงและไม่มีตัวกรองเสียง.....	73
4.5.1 วัตถุประสงค์	73
4.5.2 อุปกรณ์ในการทดลอง.....	73
4.5.3 วิธีการทดลอง	73
4.5.4 ผลการทดลอง.....	73
4.6 การทดลองทำนายผลของแอฟฟลิเคชันด้วยไฟล์เสียง	76
4.6.1 วัตถุประสงค์	76
4.6.2 อุปกรณ์ในการทดลอง.....	76
4.6.3 วิธีการทดลอง	77
4.6.4 ผลการทดลอง.....	77
บทที่ 5 บทสรุปและข้อเสนอแนะ	82
5.1 สรุปผลการทดลอง	82
5.1.1 สรุปผลการทดลองแอฟฟลิเคชันบนคอมพิวเตอร์ระบบปฏิบัติการวินโดวส์ 11	82
5.1.2 สรุปผลการทดลองหาความแม่นยำของฟิจเจอร์จากการเทรน.....	82
5.1.3 สรุปผลการทดลองเสียงที่มีลักษณะการไอบต่างกัน	82
5.1.4 สรุปผลการทดลองเสียงโอบที่มีจำนวนลูกคลื่นต่างกัน	82
5.1.5 สรุปผลการทดลองเสียงโอบแบบมีตัวกรองเสียงและไม่มีตัวกรองเสียง	82
5.1.6 สรุปผลการทดลองทำนายผลของแอฟฟลิเคชันด้วยไฟล์เสียง.....	83
5.2 ปัญหาและอุปสรรค.....	83
5.2.1 ความแม่นยำของฟิจเจอร์	83
5.2.2 ไฟล์เสียงที่ใช้ในการสร้างฟิจเจอร์	83
5.3 แนวทางแก้ไข.....	83
5.3.1 ความแม่นยำของฟิจเจอร์.....	83
5.3.2 ไฟล์เสียงที่ใช้ในการสร้างฟิจเจอร์	84

สารบัญ(ต่อ)

	หน้า
เอกสารอ้างอิง.....	85
ภาคผนวก ก โปรแกรมการทำงาน	87
ภาคผนวก ข คู่มือการใช้งานแอปพลิเคชันช่วยประเมินโรคจากอาการไอ.....	114
ประวัติผู้เขียน	121



สารบัญตาราง

ตารางที่	หน้า
1.1 ขั้นตอนการดำเนินงานของโครงการภาคเรียนที่ 1.....	3
1.2 ขั้นตอนการดำเนินงานของโครงการภาคเรียนที่ 2.....	4
4.1 การทดลองหาค่าความแม่นยำของพีเจอร์จากการเทรนจำนวนข้อมูลต่างกัน	63
4.2 แสดงค่าความแตกต่างที่มีลักษณะการโอที่ต่างกันของขอบเขตบน.....	66
4.3 แสดงค่าความแตกต่างที่มีลักษณะการโอที่ต่างกันของขอบเขตล่าง.....	66
4.4 การทดลองเสียงโอที่มีจำนวนลูกคลื่นต่างกัน	69
4.5 การทดลองเสียงโอแบบมีตัวกรองเสียงและไม่มีตัวกรองเสียงในอุปกรณ์รับเสียง	74
4.6 การทดลองทำนายผลลัพธ์ของแอปพลิเคชันด้วยไฟล์เสียง	78

สารบัญรูป

รูปที่	หน้า
2.1 โรคโควิด-19.....	5
2.2 การเรียนรู้เชิงลึก	8
2.3 การเรียนรู้เชิงลึกของเสียง.....	9
2.4 การเรียนรู้ของเครื่อง.....	11
2.5 สัญลักษณ์วิชชวลสตูดิโอโค้ด.....	16
2.6 สัญลักษณ์ภาษาไพทอน	17
2.7 Json.....	18
2.8 ลิโบร่า.....	18
2.9 นัมพาย.....	19
2.10 แมทซ์.....	19
2.11 อุปกรณ์รับเสียง	20
2.12 สัญลักษณ์ของชานีไฟล์.....	21
2.13 สัญลักษณ์ของกีวี.....	21
3.1 บล็อกไดอะแกรมการออกแบบแอปพลิเคชันช่วยประเมินโรคจากอาการไอ.....	23
3.2 เป็นรูปแผนผังการทำงานของแอปพลิเคชัน	25
3.3 โฟลเดอร์ที่สร้างเพื่อเป็นที่อยู่ของแอปพลิเคชัน	26
3.4 โฟลเดอร์ที่สร้างเพื่อเป็นที่อยู่ของเสียงไอ.....	27
3.5 โฟลเดอร์ที่สร้างเพื่อเป็นที่อยู่ของเสียงไอแต่ละประเภท	27
3.6 โฟลเดอร์ที่สร้างเพื่อเป็นที่อยู่ของชุดข้อมูลที่ต้องการเทรน.....	28
3.7 ไฟล์สคริปต์ไพทอนที่ใช้เขียนโปรแกรมสร้างชุดข้อมูลสำหรับการเทรน.....	29
3.8 โลบารี่ ที่จำเป็นในการใช้งานสร้างชุดข้อมูล	29
3.9 ชุดคำสั่งต่าง ๆ ที่ใช้สร้างชุดข้อมูลสำหรับการเทรน.....	30
3.10 การรันโปรแกรม	31
3.11 ชุดข้อมูลสำหรับการเทรน.....	31
3.12 ตัวอย่างชุดข้อมูลสำหรับการเทรน.....	32

สารบัญรูป(ต่อ)

รูปที่

หน้า

3.13	ไฟล์โปรแกรมสร้างฟิจเจอร์	33
3.14	ไลบรารีและชื่อไฟล์ที่ต้องการอ่านและบันทึก	33
3.15	ชุดคำสั่งหลักในการสร้างฟิจเจอร์	34
3.16	ชุดคำสั่งอ่านค่าข้อมูลใน ชุดข้อมูล	35
3.17	ชุดคำสั่งสร้างขอบเขตบนของชุดข้อมูล	35
3.18	ชุดคำสั่งสร้างขอบเขตล่างของชุดข้อมูล	36
3.19	ชุดคำสั่งสร้างขีดจำกัดมากที่สุดขอบเขตบนของชุดข้อมูล	36
3.20	ชุดคำสั่งสร้างขีดจำกัดน้อยที่สุดขอบเขตบนของชุดข้อมูล	37
3.21	ชุดคำสั่งสร้างขีดจำกัดมากที่สุดขอบเขตล่างของชุดข้อมูล	37
3.22	ชุดคำสั่งสร้างขีดจำกัดน้อยที่สุดขอบเขตล่างของชุดข้อมูล	38
3.23	ชุดคำสั่งหาความแม่นยำของฟิจเจอร์	39
3.24	ชุดคำสั่งบันทึกฟิจเจอร์เป็นสกุลไฟล์เจสัน	40
3.25	ตัวอย่างฟิจเจอร์สกุลไฟล์เจสัน	40
3.26	ชุดคำสั่งพล็อตกราฟ ฟิจเจอร์	41
3.27	กราฟฟิจเจอร์ที่ได้จากการเทรน	42
3.28	ไฟล์แอปพลิเคชัน	43
3.29	ไลบรารีที่จำเป็นและค่าที่ต้องกำหนด	44
3.30	คำสั่งหลักในการทำงานของแอปพลิเคชันส่วนที่ 1	44
3.31	คำสั่งหลักในการทำงานของแอปพลิเคชันส่วนที่ 2	45
3.32	คำสั่งออกแบบหน้าต่างแอปพลิเคชัน	46
3.33	ไฟล์ “ahadc.kv” ที่ได้หลังจากการบันทึกลงโฟลเดอร์เดียวกับแอปพลิเคชัน	47
3.34	หน้าต่างแอปพลิเคชันหลังการออกแบบ	47
3.35	คำสั่งอ่านข้อมูลฟิจเจอร์ที่ได้เตรียมไว้	48
3.36	คำสั่งบันทึกเสียง	48
3.37	คำสั่งอ่านไฟล์เสียงที่ได้บันทึกเป็นไฟล์สกุล .wav	49

สารบัญรูป(ต่อ)

รูปที่

หน้า

3.38 คำสั่งหาขอบเขตบนของไฟล์เสียงที่ได้บันทึกเป็นไฟล์สกุล wav ในแต่ละส่วน	49
3.39 คำสั่งหาขอบเขตล่างของไฟล์เสียงที่ได้บันทึกเป็นไฟล์สกุล wav ในแต่ละส่วน	50
3.40 คำสั่งประมวลผลเปรียบเทียบเสียงที่รับเข้ากับฟิจเจอร์	51
3.41 คำสั่งทำนายผล.....	52
3.42 คำสั่งการพล็อตกราฟเปรียบเทียบเสียงกับฟิจเจอร์	53
3.43 หน้าต่างเทอร์มินอลที่ทำงานเมื่อรันแอปพลิเคชัน	53
3.44 หน้าต่างแอปพลิเคชันเมื่อรันแอปพลิเคชัน.....	54
3.45 หน้าต่างเทอร์มินอลเมื่อบันทึกเสียงแล้วผ่านการประมวลผลและทำนายผล	54
3.46 กราฟการเปรียบเทียบเสียงไอและฟิจเจอร์.....	55
3.47 ภาพรวมการทำงานของแอปพลิเคชัน	56
3.48 โพลเดอร์หลังจากการทำงานของแอปพลิเคชัน	57
4.1 หน้าต่างแสดงข้อความพร้อมที่จะเริ่มบันทึกเสียงไอ	59
4.2 หน้าต่างแสดงข้อความกำลังบันทึกเสียง	60
4.3 หน้าต่างแสดงผลการประมวลผลเสียงไอปกติ	61
4.4 หน้าต่างแสดงผลการประมวลผลเสียงติดเชื้อโควิด19	61
4.5 หน้าต่างแสดงผลการประมวลผลเสียงที่ไม่ใช่เสียงไอ.....	62
4.6 กราฟฟิจเจอร์เสียงไอคนเดียว 1 ลูกคลื่น ขนาดไฟล์ 1 วินาที ในลักษณะการไต่ต่างกัน	68
4.7 เสียงไอคนเดียวขนาดไฟล์หนึ่งวินาทีแบบหนึ่งและสองลูกคลื่น	71
4.8 เสียงไอคนเดียวขนาดไฟล์หนึ่งวินาทีแบบสามและสี่ลูกคลื่น.....	72
4.9 เสียงไอที่มีและไม่มีตัวกรองเสียงในอุปกรณ์รับเสียง	76

บทที่ 1

บทนำ

ในบทนี้จะกล่าวถึง ที่มาและความสำคัญของปัญหา วัตถุประสงค์ของการศึกษา สมมุติฐาน การศึกษา ขอบเขตของการศึกษา ประโยชน์ที่คาดว่าจะได้รับ ขั้นตอนการดำเนินงาน รวมถึงโครงสร้างของปฏิญานิพนธ์ ซึ่งมีรายละเอียดดังนี้

1.1 ความเป็นมาและความสำคัญของปัญหา

เนื่องจากในปัจจุบันได้มีโรคโควิด-19 ขึ้นซึ่งเป็นโรคยอดนิยมที่ระบาดหนักทั่วโลก โดยผู้ป่วยเหล่านี้จะมีการไอหรือจามเหมือนกับโรคไข้หวัดธรรมดา

โดยอาการไอหรือจามนั้นผู้คนส่วนใหญ่อาจจะคิดว่าตนเองป่วยเป็นแค่ไข้หวัดธรรมดาแล้วอาจจะละเลยการตรวจคัดกรองและใช้ชีวิตปะปนกับคนทั่วไปและนี่คือสาเหตุหลักที่ทางผู้พัฒนาได้จัดทำโครงการฉบับนี้ขึ้นมาเพื่อศึกษาวิจัยเกี่ยวกับสัญญาณคลื่นเสียงไอเพื่อนำมาพัฒนาแอปพลิเคชัน

1.2 ความมุ่งหมายและวัตถุประสงค์

1. เพื่อศึกษาค้นสัญญาณเสียงจากอาการไอ
2. เพื่อศึกษาการประเมินโรคจากอาการไอ
3. เพื่อศึกษาและออกแบบแอปพลิเคชันบนระบบปฏิบัติการวินโดวส์
4. เพื่อศึกษาการประมวลผลคลื่นเสียงของแอปพลิเคชันบนระบบปฏิบัติการวินโดวส์
5. เพื่อศึกษางานวิจัยคลื่นเสียงที่เกี่ยวข้องจากหลากหลายแหล่งที่มา

1.3 สมมุติฐานของการศึกษา

โรคโควิด-19 นั้นเป็นเชื้อไวรัสที่สามารถแพร่กระจายได้ในอากาศ ทำให้ผู้คนสามารถติดเชื้อได้ง่ายและเมื่อติดเชื้อแล้วจะทำให้ผู้ป่วยมีไข้และมีอาการไอ เพราะว่าเชื้อโควิดจะไปทำลายปอดของผู้ที่ติดเชื้อ ส่งผลให้ผู้ติดเชื้อมีเสียงที่เปลี่ยนไป ถ้าหากสามารถวิเคราะห์เสียงไอที่เกิดจากโรคโควิด-19 ได้ จะสามารถทำให้จำแนกเสียงและวิเคราะห์เสียงได้ แล้วนำมาสร้างเป็นแอปพลิเคชัน เป็นแอปพลิเคชันตรวจสอบเสียงไอ เป็นต้น

1.4 ขอบเขตของการศึกษา

1. เขียนอัลกอริทึมในการช่วยประเมินโรคจากเสียงไอได้อย่างน้อย 1 โรค
2. แอปพลิเคชันสามารถทำงานบนคอมพิวเตอร์ระบบปฏิบัติการวินโดวส์
3. สามารถรับข้อมูลเสียงไอเป็นระยะเวลาสูงสุด 15 วินาที
4. ผู้ใช้งานสามารถดาวน์โหลดแล้วติดตั้งแอปพลิเคชันได้ด้วยตนเอง
5. สามารถใช้งานแอปพลิเคชันในบริเวณที่ไร้เสียงรบกวน
6. สามารถแสดงหน้าต่างการทำนายผลได้

1.5 ประโยชน์ที่ได้รับ

1. ได้รับความรู้จากการศึกษาค้นคว้าสัญญาณเสียงจากอาการไอ
2. ได้รับความรู้การประเมินโรคจากอาการไอ
3. ได้รับความรู้จากการศึกษาและออกแบบโปรแกรมบนระบบปฏิบัติการวินโดวส์
4. ได้รับความรู้จากการประมวลผลคลื่นเสียงของโปรแกรมบนระบบปฏิบัติการวินโดวส์
5. ได้รับความรู้จากการศึกษางานวิจัยคลื่นเสียงที่เกี่ยวข้องจากหลากหลายแหล่งที่มา

1.6 ขั้นตอนและแผนการดำเนินงาน

ขั้นตอนการดำเนินงานที่ผู้จัดทำได้วางแผนไว้ เริ่มจากการศึกษาค้นคว้ารวบรวมข้อมูลของโรคโควิด-19 ที่ใช้ในการทำโครงงาน ออกแบบสร้างแอปพลิเคชัน ศึกษาการเขียนโค้ดต่างๆที่ใช้ภาษาไพทอน ในการสร้างแอปพลิเคชันและใช้ในการสร้างระบบการเรียนรู้เชิงลึกของเสียง จัดทำโครงงานทดลองรันแอปพลิเคชันบนระบบจำลอง (Emulator) บนคอมพิวเตอร์ (computer) ทดสอบการทำงานทั้งหมด เก็บผลการทดลอง การแก้ไข จัดทำรายงาน และเตรียมการนำเสนอ จะแบ่งตามภาคเรียนที่ 1 และ 2 ซึ่งได้ชี้แจง รายละเอียดในตารางที่ 1.1 และ 1.2 ดังนี้

ตารางที่ 1.1 ขั้นตอนการดำเนินงานของโครงการภาคเรียนที่ 1

รายละเอียด	ช่วงเวลาดำเนินงาน (สิงหาคม - พฤศจิกายน 2564)															
	สิงหาคม				กันยายน				ตุลาคม				พฤศจิกายน			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1.ศึกษาและรวบรวมข้อมูลเกี่ยวกับโครงการ	←		→													
2.เข้าพบอาจารย์ที่ปรึกษา			←	→												
3.นำเสนอหัวข้อโครงการต่อคณาจารย์				↔												
4.ค้นคว้าหาข้อมูลของโรคที่ทำการศึกษา				←				→								
5.ค้นคว้าหาข้อมูลของการออกแบบแอปพลิเคชัน						←				→						
6.ค้นคว้าหาข้อมูลเกี่ยวกับการเขียนโค้ดในการสร้างแอปพลิเคชัน									←			→				
7.ค้นคว้าและศึกษาเกี่ยวกับการทำระบบการเรียนรู้เชิงลึกของเสียง								←								→
8.ทดลองสร้างและออกแบบแอปพลิเคชัน								←								→
9.ทดลองเขียนระบบการเรียนรู้เชิงลึกของเสียง ที่ใช้เสียงในการแยกโรค								←								→
10.ทดสอบแอปพลิเคชันและระบบของแอปพลิเคชันที่สร้างขึ้น									←							→
11.สรุปผลการทดลอง															←	→
12.จัดทำรายงานและนำเสนอ															←	→

ตารางที่ 1.2 ขั้นตอนการดำเนินงานของโครงการภาคเรียนที่ 2

รายละเอียด	ช่วงเวลาดำเนินงาน (มกราคม - พฤษภาคม 2565)															
	มกราคม				กุมภาพันธ์				มีนาคม				เมษายน			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1.หาข้อมูลและศึกษาวิธีการเขียนระบบการเรียนรู้เชิงลึกของเสียง																
2.นำข้อมูลมาเข้ากระบวนการทำระบบการเรียนรู้เชิงลึกของเสียง																
3.พัฒนาโปรแกรมให้ดีขึ้น																
4.ทดสอบแอปพลิเคชัน																
5. จัดทำเล่มปริญญานิพนธ์																
6.นำเสนอโครงการ																

1.7 โครงสร้างปริญญานิพนธ์

ปริญญานิพนธ์ฉบับนี้จะอธิบายเนื้อหาที่เกี่ยวข้องกับแอปพลิเคชันช่วยประเมินโรคจากอาการไอ แบ่งออกเป็น 5 บทแต่ละบทประกอบไปด้วยเนื้อหาดังต่อไปนี้

บทที่1ในบทนี้จะกล่าวถึงที่มาและความสำคัญของปัญหา วัตถุประสงค์ของการศึกษา ขอบเขตของปริญญานิพนธ์ ประโยชน์ที่คาดว่าจะได้รับ แผนขั้นตอนดำเนินงาน และโครงสร้างของปริญญานิพนธ์

บทที่2 ในบทนี้จะกล่าวถึง2ทฤษฎีที่เกี่ยวข้องกับการวิเคราะห์คลื่นเสียงและการสร้างแอปพลิเคชันและการสร้างระบบการเรียนรู้เชิงลึกของเสียงที่ใช้ในการทำโครงการซึ่งประกอบไปด้วยแอปพลิเคชันต่างๆที่ใช้ในการสร้างแอปพลิเคชัน เช่น วิชชอนสตูดิโอโค้ด , ภาษาไพทอน , กวี , การสร้างระบบการเรียนรู้เชิงลึกของเสียง เป็นต้น

บทที่3 ในบทนี้จะกล่าวถึงบล็อกไดอะแกรมของตัวแอปพลิเคชันและการทำงานต่างๆในแอปพลิเคชัน

บทที่4 ในบทนี้จะกล่าวถึงการทดลองและผลการทดลองแอปพลิเคชัน

บทที่5 วิเคราะห์และสรุปผลการทดลอง ในบทนี้จะกล่าวถึงการสรุปผลการทำงานของแอปพลิเคชัน ปัญหาและอุปสรรค วิธีแก้ไขปัญหา และข้อเสนอแนะ

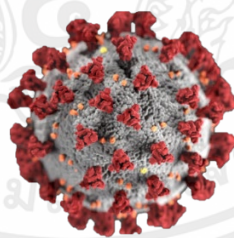
บทที่ 2

ทฤษฎีที่เกี่ยวข้อง

โครงการแอปพลิเคชันช่วยประเมินโรคจากอาการไอโดยมีระบบ AI (การเรียนรู้เชิงลึกของเสียง) มาช่วยในการประมวลผลวิเคราะห์โรคให้แม่นยำขึ้นและสามารถประเมินโรคเบื้องต้นได้นั้น มีรายละเอียดดังต่อไปนี้

2.1 โรคโควิด-19 (Covid-19)

โรคโควิด-19 [1] เป็นไวรัสชนิดอาร์เอ็นเอสายเดี่ยว เป็นไวรัสที่ถูกพบครั้งแรกในปี 1960 และมีรายงานการพบเชื้อในช่วงปี ค.ศ. 1965 โดยสามารถแพร่เชื้อได้ทั้งในคนและสัตว์ ปัจจุบันมีการค้นพบไวรัสสายพันธุ์นี้แล้วทั้งหมด 6 สายพันธุ์ ส่วนสายพันธุ์ที่กำลังแพร่ระบาดหนักทั่วโลกตอนนี้เป็นสายพันธุ์ที่ยังไม่เคยพบมาก่อน คือ สายพันธุ์ที่ 7 จึงถูกเรียกว่าเป็น “ไวรัสโคโรนาสายพันธุ์ใหม่” และในภายหลังถูกตั้งชื่ออย่างเป็นทางการว่า “โควิด-19” โดยไวรัสโคโรนานั้นจะสามารถแพร่กระจายได้อย่างรวดเร็วผ่านทางอากาศซึ่งทำให้ผู้คนส่วนใหญ่ติดเชื้อกันแบบไม่รู้ตัว ซึ่งเมื่อติดเชื้อแล้วจะทำให้เกิดโรคโควิด-19 และเมื่อติดเชื้อแล้วจะทำให้แสดงอาการในระบบต่างๆ เช่น ระบบหายใจ ระบบย่อยอาหาร ระบบทางเดินอาหาร ระบบประสาท และระบบอื่นๆ ดังรูปที่ 2.1



รูปที่ 2.1 โรคโควิด-19

(ที่มา : <https://www.safesiri.com/covid19>)

2.1.1 ระบาดวิทยาของเชื้อ

เชื้อไวรัสโคโรนา พบได้ทั่วโลกโดยในเขตอบอุ่นมักจะพบเชื้อไวรัสโคโรนาในช่วงฤดูหนาว และในช่วงฤดูใบไม้ผลิ ส่วนใหญ่เชื้อไวรัสโคโรนาจะแพร่กระจายอย่างรวดเร็วในพื้นที่ที่อากาศหนาวเย็น

การติดเชื้อโคโรนาไวรัสอาจทำให้เกิดอาการในระบบทางเดินหายใจส่วนบนได้ถึงร้อยละ 35 และอาจเกิดไข้หวัดจากการติดเชื้อไวรัสโคโรนาสูงถึงร้อยละ 15 ซึ่งการติดเชื้อนั้นพบได้ในทุกกลุ่มอายุ แต่พบมากในเด็กและผู้สูงอายุและอาจพบมีการติดเชื้อซ้ำได้ เนื่องจากระดับภูมิคุ้มกันจะลดลงอย่างรวดเร็วภายหลังการติดเชื้อ สำหรับการติดเชื้อทางเดินหายใจเฉียบพลันรุนแรง พบการระบาดในปี พ.ศ. 2546 โดยพบเริ่มจากประเทศจีนแล้วแพร่กระจายไปทั่วโลก พบรายงานผู้ป่วยโรคซาร์สทั้งสิ้นมากกว่า 8,000 ราย และเสียชีวิตมากกว่า 750 ราย

2.1.2 ลักษณะของโรค

การติดเชื้อไวรัสโคโรนาในระบบทางเดินหายใจ อาจทำให้เกิดอาการไข้ อ่อนเพลีย ปวดศีรษะ มีน้ำมูก เจ็บคอ ไอ โดยในทารกที่มีอาการรุนแรง อาจมีลักษณะของปอดอักเสบ หรือ หลอดลมฝอยอักเสบในเด็กโตอาจมีอาการของหอบหืดส่วนในผู้ใหญ่ อาจพบลักษณะปอดอักเสบ หลอดลมอักเสบเรื้อรัง หรือการกลับเป็นซ้ำของโรคหอบหืดได้ และอาจทำให้เกิดอาการรุนแรงได้มากในผู้สูงอายุหรือผู้ที่ภูมิคุ้มกันบกพร่อง โดยพบการติดเชื้อแบบไม่แสดงอาการได้ในทุกอายุ และหากแสดงอาการมักพบร่วมกับการติดเชื้อในระบบทางเดินหายใจอื่นๆ การติดเชื้อทางเดินหายใจเฉียบพลันรุนแรง หรือซาร์ส จะพบมีอาการไข้ ปวดศีรษะ อ่อนเพลีย หรืออาการคล้ายไข้หวัดใหญ่ แล้วมีอาการไอและหอบเหนื่อยอย่างรวดเร็ว ซึ่งอัตราตายจะสูงขึ้นในผู้ป่วยสูงอายุ หรือมีโรคประจำตัว การติดเชื้อโคโรนาไวรัสในระบบทางเดินอาหาร มักพบบ่อยในเด็กแรกเกิด และทารกอายุน้อยกว่า 1 ปี หรืออาจพบในผู้ใหญ่ที่มีภูมิคุ้มกันบกพร่อง โดยพบเชื้อได้แม้ผู้ป่วยไม่แสดงอาการ และไม่มีฤดูกาลการเกิดโรคที่แน่นอน

2.1.3 ระยะฟักตัวของโรค

โดยเฉลี่ยประมาณ 2 วัน (อาจมีระยะฟักตัวนานถึง 3 – 4 วัน)

2.1.4 วิธีการแพร่ระบาดของโรค

แพร่กระจายเชื้อจากการสัมผัสกับสารคัดหลั่งหลังจากทางเดินหายใจ หรือ แพร่กระจายเชื้อจากฝอยละอองน้ำมูก น้ำลายจากผู้ป่วยที่มีเชื้อโดยการ ไอ หรือจาม (ลักษณะการแพร่กระจายของเชื้อโรคจะคล้ายๆกับการแพร่กระจายเชื้อของโรคไข้หวัด)

2.1.5 อาการเมื่อติดเชื้อไวรัสโคโรนาสายพันธุ์ใหม่ หรือไวรัสโควิด-19

อาการของไวรัสโควิด-19 ที่สังเกตได้ง่าย ๆ ด้วยตัวเอง ดังนี้

1. มีไข้
2. เจ็บคอ
3. ไอแห้ง ๆ
4. น้ำมูกไหล

5. หายใจเหนื่อยหอบ

2.1.6 กลุ่มเสี่ยงติดเชื้อไวรัสโคโรนาสายพันธุ์ใหม่ หรือโควิด-19

1. เด็กเล็ก (จะมีอาการรุนแรงเท่าผู้สูงอายุ)
2. ผู้สูงอายุ
3. คนที่มีโรคประจำตัวอยู่
4. คนที่ภูมิคุ้มกันผิดปกติ หรือ มีการใช้ยากดภูมิต้านทานโรคอยู่
5. คนที่มีน้ำหนักเกินมาตรฐานมาก
6. ผู้ที่เดินทางเข้า-ออกต่างประเทศ
7. ผู้ที่ต้องทำงานใกล้ชิดกับผู้ป่วยโรค โควิด-19
8. ผู้ที่ทำอาชีพที่ต้องพบปะชาวต่างชาติจำนวนมาก เช่น เจ้าหน้าที่ในโรงพยาบาล

ลูกเรือสายการบินต่าง ๆ เป็นต้น

2.1.7 หากมีอาการของโรคโควิด-19

1. หากมีอาการของโรคที่เกิดขึ้นตาม 5 ข้อดังกล่าว ควรพบแพทย์เพื่อทำการตรวจอย่างละเอียดและเมื่อแพทย์ซักถามควรตอบตามความเป็นจริง ไม่ปิดบัง ไม่บิดเบือนข้อมูลใด ๆ เพราะจะเป็นประโยชน์ต่อการวินิจฉัยโรคอย่างถูกต้องมากที่สุด
2. หากเพิ่งเดินทางกลับจากพื้นที่เสี่ยง ควรกักตัวเองอยู่แต่ในบ้าน ไม่ออกไปข้างนอกเป็นเวลา 14-27 วัน เพื่อให้ผ่านช่วงเชื้อฟักตัว (ให้แน่ใจจริง ๆ ว่าไม่ติดเชื้อ)

2.1.8 วิธีป้องกันการติดเชื้อไวรัสโคโรนาสายพันธุ์ใหม่

1. หลีกเลี่ยงการใกล้ชิดกับผู้ป่วยที่มีอาการไอ จาม น้ำมูกไหล เหนื่อยหอบ เจ็บคอ
2. หลีกเลี่ยงการเดินทางไปในพื้นที่เสี่ยง
3. สวมหน้ากากอนามัยทุกครั้งเมื่ออยู่ในที่สาธารณะ
4. ระมัดระวังการสัมผัสพื้นผิวที่ไม่สะอาด และอาจมีเชื้อโรคเกาะอยู่ รวมถึงสิ่งที่คนจับบ่อยครั้ง เช่น ที่จับบนรถไฟฟ้าและรถไฟใต้ดินที่เปิด-ปิดประตูในรถ กลอนประตูต่าง ๆ ก๊อกน้ำ ราวบันได เมื่อจับแล้วอย่าเอามือสัมผัสหน้า และข้าวของเครื่องใช้ส่วนตัวต่าง ๆ เช่น โทรศัพท์มือถือ กระเป๋า เป็นต้น
5. ล้างมือให้สะอาดด้วยสบู่ หรือแอลกอฮอล์เจลอย่างน้อย 20 วินาที ความเข้มข้นของแอลกอฮอล์ไม่ต่ำกว่า 70% (ไม่ผสมน้ำ)
6. งดจับตา จมูก ปากขณะที่ไม่ได้ล้างมือ
7. หลีกเลี่ยงการใกล้ชิด สัมผัสสัตว์ต่าง ๆ โดยที่ไม่มีการป้องกัน

8. รับประทานอาหารสุก สะอาด ไม่ทานอาหารที่ทำจากสัตว์หายาก

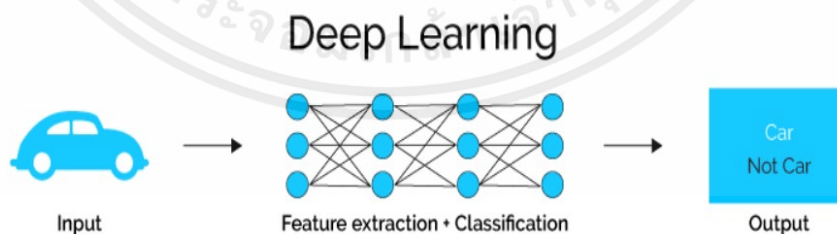
9. สำหรับบุคลากรทางการแพทย์หรือผู้ที่ต้องดูแลผู้ป่วยที่ติดเชื้อไวรัสโคโรนาสายพันธุ์ใหม่ หรือโควิด-19 โดยตรง ควรใส่หน้ากากอนามัย หรือใส่แว่นตานิรภัย เพื่อป้องกันเชื้อในละอองฝอยจากเสมหะหรือสารคัดหลั่งเข้าตา

2.1.9 ภาวะลองโควิด

ภาวะลองโควิดหรืออาการหลงเหลือหลังติดเชื้อโควิด-19 การติดเชื้อโควิด-19 ส่งผลต่อร่างกายในแต่ละคน มากน้อยแตกต่างกันไป บางรายอาจมีภาวะแทรกซ้อนรุนแรงขณะรับการรักษา บางรายหลังรักษาหายแล้วอาจยังมีอาการ หรือความผิดปกติต่างๆ หลงเหลืออยู่ และอาการต่างๆ อาจเป็นอยู่นานมากถึง 3 เดือนหลังการรักษาอาการต่างๆ เหล่านี้ เรียกว่า ลองโควิด (Long Covid) หรือโพสต์โควิดซินโดรม (Post Covid Syndrome) ซึ่งพบได้บ่อย 30% – 80% ในผู้ป่วยที่หายจากการติดเชื้อโควิดแล้ว

2.2 การเรียนรู้เชิงลึก (Deep learning)

การเรียนรู้เชิงลึก [2] คือ อัลกอริทึมที่มีหลักการทำงานคล้ายกับการทำงานของระบบโครงข่ายประสาท (neurons) ในสมองของมนุษย์ ซึ่งการเรียนรู้เชิงลึก เป็นอัลกอริทึมที่ใช้ระบบโครงข่ายประสาทเทียมในการวิเคราะห์ข้อมูลที่มีความเชื่อมโยงกัน โดยส่วนใหญ่จะใช้การเรียนรู้เชิงลึก ในการวิเคราะห์ตรวจจำวัตถุ เช่น การวิเคราะห์รูปภาพ (image processing) ตลอดจนไปถึงการวิเคราะห์เสียง เช่น การจดจำเสียงพูดเพื่อใช้เป็นคำสั่งเสียงเพื่อสั่งงานอุปกรณ์บางชนิดการเรียนรู้เชิงลึก นั้นเป็นระบบปัญญาประดิษฐ์ (AI : Artificial Intelligence) ขนาดย่อมๆ ที่ถือเป็นขั้นขั้นต้นของ การเรียนรู้ของเครื่อง (machine learning) และมีกระบวนการสร้าง ที่คล้ายคลึงกับการเรียนรู้ของเครื่อง ดังรูปที่ 2.2

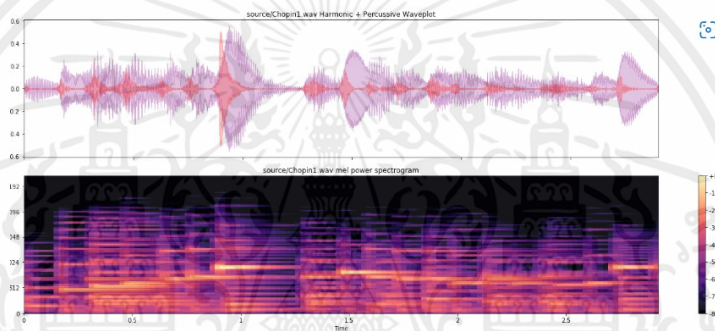


รูปที่ 2.2 การเรียนรู้เชิงลึก

(ที่มา : <https://semiengineering.com/deep-learning-spreads>)

2.3 การเรียนรู้เชิงลึกของเสียง

การเรียนรู้เชิงลึกของเสียง [3] คือ อัลกอริทึมที่ใช้ในการเรียนรู้ จัดจำ จำแนกเสียง โดยการเรียนรู้เชิงลึกของเสียง นั้นเป็นการนำเอา กระบวนการของการสร้างการเรียนรู้เชิงลึก มาประยุกต์ใช้ เพื่อวิเคราะห์ข้อมูลเชิงลึกของเสียง เพื่อจำแนกคุณสมบัติของเสียงในรูปแบบของการจำแนกความถี่ของคลื่นเสียงที่อยู่ในรูปแบบของสัญญาณ (.wav) และนำเอาไปประยุกต์ใช้ในชีวิตประจำวัน เช่น การนำเอา Audio deep learn มาสร้างอยู่ในรูปแบบของแอปพลิเคชัน โปรแกรมต่างๆ ยกตัวอย่างเช่น แอปพจนเนอร์กีตาร์ที่ใช้ในการตั้งสายกีตาร์ (Guitar Tuna) แอปค้นหาเพลงที่เมื่อเปิดแอปขึ้นมาแล้วให้มันฟังเพลง มันจะสามารถระบุเพลงหรือประเภทของเพลงขึ้นมาได้ทันทีที่ได้ทันที (shazam) ดังรูปที่ 2.3



รูปที่ 2.3 การเรียนรู้เชิงลึกของเสียง

(ที่มา : medium.com/olarik/โครงข่ายประสาทเทียมแบบคอนโวลูชัน-4e9e3a7a39bf)

2.3.1 กระบวนการสร้างการเรียนรู้เชิงลึกของเสียง

เป็นกระบวนการสร้างการเรียนรู้เชิงลึกของเสียง เพื่อเรียนรู้ จัดจำ จำแนกข้อมูลเชิงลึกของเสียงเพื่อนำเอาไปประยุกต์ใช้ในชีวิตประจำวัน

1. ชุดข้อมูล (Dataset) การเตรียมชุดข้อมูลนั้นเป็นกระบวนการแรกที่ใช้ในการสร้างการเรียนรู้เชิงลึกของเสียง เพิ่มศึกษาข้อมูลเชิงลึกของเสียง โดยเริ่มต้นจากการค้นหาไฟล์เสียงที่ต้องการจะศึกษาแล้วจากนั้นดาวน์โหลดมัน ซึ่งในการค้นหาไฟล์เสียงนั้นถ้าหากดาวน์โหลดมาแล้วมันเป็นไฟล์ .mp3 จะต้องทำการแปลงไฟล์ให้อยู่ในรูปแบบของไฟล์ .wav ก่อนเพิ่มที่จะทำให้สามารถเห็นเสียงในรูปแบบของคลื่นสัญญาณและมันจะสามารถทำให้ได้เห็นสัญญาณรบกวนในเสียงที่ต้องการสร้างเป็นชุดข้อมูล เมื่อเห็นคลื่นสัญญาณเสียงรบกวนในไฟล์เสียงตัดมันออกไปเพื่อที่จะทำให้เสียงที่ต้องการนั้นมีความสะอาด

หรือว่าชัดเจนและไร้เสียงรบกวน เมื่อเสียงไร้ซึ่งสัญญาณรบกวน นำเสียงนั้นมาตัดตามความเหมาะสมกับชิ้นงานที่กำลังทำ

2. เตรียมชุดข้อมูล (Prepare Dataset) เมื่อค้นหาไฟล์เสียงที่ผู้จัดทำพอใจได้แล้วและนำมาแปลงเป็นไฟล์ .wav เรียบร้อยแล้ว จากนั้นสร้างไฟล์เตอร์มา 1 ไฟล์ เพื่อเก็บไฟล์เสียงที่เป็นนามสกุล .wav เอาไว้ ซึ่งถ้ามีไฟล์เสียงเยอะจะยิ่งดี ซึ่งมันทำให้สามารถสร้างการเรียนรู้เชิงลึกของเสียง ที่มีความแม่นยำได้มากขึ้น ในการเตรียมไฟล์นั้นจะต้องเตรียมไฟล์ไว้ 2 ชุด คือ ไฟล์เสียงสำหรับชุดข้อมูลตัวอย่าง 1 ชุด และ ไฟล์เสียงสำหรับทดสอบชุดข้อมูลตัวอย่างอีก 1 ชุด

3. เตรียมชุดข้อมูลเสียงเพื่อประมวลผล โดยเริ่มจากนำเอาไฟล์เสียงที่เป็นนามสกุล .wav ที่ตัดเสียงไว้แล้วไปใส่ไว้ในไฟล์เตอร์ อาจจะตั้งชื่อว่า ข้อมูลเมต้า (metadata) ได้ หลังจากนั้นเขียนโปรแกรมไพทอน ขึ้นมาแล้วดึง แพทช์ (path) ไฟล์ ข้อมูลเมต้า มาใส่ในชุดโค้ดไพทอนที่เขียนขึ้นมา เพื่อบันทึกเป็นไฟล์ .csv หรือ .เจสัน หรือไฟล์อื่นๆ จากนั้นเขียนโค้ดไพทอนสำหรับอ่านไฟล์ .csv หรือ .เจสัน เพื่อดึงพีตเจอร์ที่อยู่ในรูปแบบของค่าอาร์เรย์ออกมา ซึ่งก่อนที่จะได้ค่าอาร์เรย์ออกมานั้นจะต้องดึงชุดโค้ดบางส่วนมาแปลงไฟล์เสียงให้อยู่ในรูปแบบของสัญญาณสเปกโตรแกรม แล้วดึงภาพสเปกโตรแกรมมาแปลงเป็น mfcc เพื่อสังเกตคลื่นสัญญาณเสียง จากนั้นนำพีตเจอร์ที่ได้มาเตรียมไว้สำหรับการสร้างเป็นโมเดลเสียง

4. สร้างแบบจำลอง หรือ โมเดลเสียง โดยขั้นตอนสร้างแบบจำลองนั้นเป็นขั้นตอนที่นำพีตเจอร์ที่ได้มาประมวลผลต่อ โดยทำการย่อยรูปแบบของพีตเจอร์ที่ถอดออกมาเพื่อให้พีตเจอร์มีความชัดเจนมากยิ่งขึ้น ซึ่งเรียกกระบวนการนี้ว่า กระบวนการ (Convolutional Neural Network : CNN) ซึ่งกระบวนการ CNN นั้นมีกระบวนการภายในที่ซับซ้อน เพื่อดึง พีตเจอร์ของไฟล์ที่ใช้ในการ เทรนและการทดสอบออกมาเพื่อให้มันเห็นความแตกต่าง ยกตัวอย่างเช่น เสียงไอโควิดกับเสียงไอปกติ จะทำอย่างไรให้เรารู้ว่าเสียงไหนเป็นเสียงไหน กระบวนการ CNN จึงเป็นกระบวนการสำคัญที่ทำให้การเรียนรู้เชิงลึก สามารถแยกแยะข้อมูลได้ และถ้าจะทำให้การเรียนรู้เชิงลึก มีประสิทธิภาพจะต้องทำการย่อย CNN หลายๆครั้งให้มันดึงพีตเจอร์ที่ชัดเจนออกมาแล้วจากนั้นเซฟเป็นโมเดลเสียง

5. กระบวนการเทรนโมเดล กระบวนการเทรนโมเดลเสียงเป็นกระบวนการทดสอบพีตเจอร์ที่เราได้ถอดออกมาจากกระบวนการ CNN เพื่อทดสอบค่าความผิดพลาดของโมเดล ถ้าหากโมเดลมีค่าความผิดพลาดที่สูงจะต้องย้อนไปทำกระบวนการ CNN ใหม่เพื่อทำให้โมเดลมันฉลาด จากนั้นเซฟไฟล์แล้วทำการทดสอบใหม่ และเมื่อเราได้โมเดลที่เราพอใจแล้วจากนั้นเซฟโมเดล แล้วนำโมเดลไปใช้

6. การนำโมเดลไปใช้ ซึ่งการนำโมเดลไปใช้ คือการนำโมเดลไปใช้ตามรูปแบบที่ต้องการ ยกตัวอย่างเช่น การนำโมเดลไปใส่ไว้ในแอปพลิเคชัน หรือ โปรแกรมเพื่อประโยชน์บางอย่าง ยกตัวอย่าง

เช่น โมเดลของ การจับคอร์คิ์ดาร์ เราใช้เพื่อทดสอบเสียงกีตาร์ ซึ่งไม่ว่าเราจะนำโมเดลไปใช้ในรูปแบบไหน จะต้องทำการเขียนโค้ดเพื่อเป็นพื้นฐานรองรับความสามารถของโมเดลนั้นๆด้วย หากว่าชุดโค้ดที่สร้างขึ้นมาสอดคล้องหรือไม่ตรงกับเงื่อนไขการทำงานงานของโมเดล แอปพลิเคชันที่สร้างหรือโปรแกรมที่สร้างจะใช้งานไม่ได้หรือเกิด ผิดพลาด (error) ดังนั้นจะต้องสร้างแอปพลิเคชันหรือโปรแกรม ขึ้นมาให้รองรับกับโมเดลด้วยเพื่อประโยชน์สูงสุดของชิ้นงาน

2.4 การเรียนรู้ของเครื่อง

การเรียนรู้ของเครื่อง [4] การเรียนรู้ของเครื่องจักร คำแปลตรงตัวของคำว่า การเรียนรู้ของเครื่อง เป็นสาขาย่อยของเอเรียความรู้อันกว้างใหญ่ไพศาลของสิ่งที่เรียกว่า ปัญญาประดิษฐ์

คำๆ นี้หมายถึงความสามารถของระบบไอทีในการหาหนทางแก้ปัญหาด้วยตนเอง โดยการจดจำรูปแบบในฐานข้อมูล กล่าวอีกนัยหนึ่งการเรียนรู้ของเครื่องช่วยให้ระบบไอทีรู้จักรูปแบบบนพื้นฐานของ อัลกอริทึมและชุดข้อมูลที่มีอยู่ เพื่อพัฒนากระบวนการแก้ปัญหาที่เหมาะสม ดังนั้นใน การเรียนรู้ของเครื่อง ความรู้ประดิษฐ์ จึงถูกสร้างขึ้นบนพื้นฐานของประสบการณ์

เพื่อให้ซอฟต์แวร์สามารถสร้างโซลูชันได้อย่างอิสระจำเป็นต้องมีการกระทำก่อนหน้านี้นี้ของมนุษย์ นั่นคือการป้อนชุดอัลกอริทึมและข้อมูลที่จำเป็นลงในระบบล่วงหน้า รวมถึงกฎการวิเคราะห์ตามลำดับเพื่อการจดจำรูปแบบต่างๆ ในสต็อกข้อมูล เมื่อทั้งสองขั้นตอนนี้เสร็จสิ้นระบบจึงจะสามารถแสดงผลการทำงานของการเรียนรู้ของเครื่องได้ ดังรูป 2.4



รูปที่ 2.4 การเรียนรู้ของเครื่อง

(ที่มา : <https://www.toolmakers.co/machine-learning>)

สรุปอย่างสั้นการเรียนรู้ของเครื่องคือ

1. การค้นหา แยกแยะและสรุปข้อมูลที่เกี่ยวข้อง
2. การคาดคะเนตามฐานข้อมูลการวิเคราะห์
3. การคำนวณความน่าจะเป็นสำหรับผลลัพธ์ที่เฉพาะเจาะจง
5. การปรับตัวให้เข้ากับการพัฒนาบางอย่างโดยอัตโนมัติ
6. การเพิ่มประสิทธิภาพกระบวนการตามรูปแบบที่รู้จักและจดจำ

2.4.1 หลักการทำงานของการเรียนรู้ของเครื่อง

ในทางหนึ่งการเรียนรู้ของเครื่องจะทำงานคล้ายกับการเรียนรู้ของมนุษย์ ตัวอย่างเช่น ถ้าแสดงภาพวัตถุประเภทต่างๆ ให้เด็กคนหนึ่งดู พวกเขาจะสามารถเรียนรู้ที่จะระบุและแยกความแตกต่างในภาพเหล่านั้นได้ การเรียนรู้ของเครื่องทำงานในลักษณะเดียวกัน: ด้วยการป้อนชุดข้อมูลและชุดคำสั่งต่างๆ คอมพิวเตอร์จะถูกใช้งานเพื่อเรียนรู้เพื่อจำแนกแยกแยะวัตถุต่างๆ รวมถึงบุคคล สิ่งของ ฯลฯ และเพื่อแยกความแตกต่างระหว่างกัน ด้วยเหตุนี้ ซอฟต์แวร์จึงถูกซัพพลายด้วยข้อมูลและการฝึกฝน ตัวอย่างเช่น โปรแกรมเมอร์สามารถบอกระบบได้ว่าวัตถุใดเป็นมนุษย์และอีกวัตถุหนึ่งไม่ใช่มนุษย์ ซอฟต์แวร์จะได้รับพื้นหลังอย่างต่อเนื่องจากโปรแกรมเมอร์ สัญญาณการตอบรับเหล่านี้จะถูกใช้โดยอัลกอริทึมเพื่อปรับและเพิ่มประสิทธิภาพโมเดล ด้วยชุดข้อมูลใหม่ที่ป้อนเข้าไปในระบบ รูปแบบจะได้รับการปรับให้เหมาะสมเพื่อให้สามารถแยกแยะระหว่างมนุษย์กับไม่ใช่มนุษย์ ได้อย่างชัดเจน

แต่การเรียนรู้ของเครื่องมีความหมายมากกว่าแค่การแยกความแตกต่างระหว่างสองสิ่งหรือสองระดับที่เป็นมนุษย์และที่ไม่ใช่มนุษย์การใช้หุ่นยนต์ตีปิงปอง เป็นตัวอย่างหนึ่งจะเห็นได้ว่าเครื่องจักรจะสแกนหาท่วงท่าที่ซับซ้อนและสไตล์การเล่นของฝ่ายตรงข้ามได้ จากนั้นปรับตัวให้เข้ากับการเล่นของพวกเขาหรือแม้แต่ทำให้แชมป์โลกหือตกได้ด้วยวิธีนี้

2.4.2 ข้อดีของการเรียนรู้ของเครื่อง

ไม่ต้องสงสัยเลยว่าการเรียนรู้ของเครื่องช่วยให้มนุษย์ทำงานได้อย่างสร้างสรรค์และมีประสิทธิภาพมากขึ้น คุณเองสามารถมอบหมายงานที่ค่อนข้างซับซ้อนหรือซ้ำซากให้กับคอมพิวเตอร์ผ่านการเรียนรู้ของเครื่องได้เริ่มจากการสแกน บันทึก และจัดเก็บเอกสารกระดาษ เช่นใบแจ้งหนี้เพื่อจัดระเบียบและแก้ไขภาพ

นอกเหนือจากงานง่ายๆ เหล่านี้ เครื่องจักรเรียนรู้ด้วยตนเอง ยังสามารถทำงานที่ซับซ้อนได้ ตัวอย่างเช่น การจดจำรูปแบบที่ผิดพลาดต่างๆ ได้ด้วย นี่เป็นข้อได้เปรียบที่สำคัญโดยเฉพาะ

อย่างยิ่งในแอเรียต่างๆ เช่นอุตสาหกรรมการผลิต ซึ่งเป็นอุตสาหกรรมที่ต้องพึ่งพาการผลิตอย่างต่อเนื่อง และปราศจากข้อผิดพลาด แม้แต่ในเรื่องที่ผู้เชี่ยวชาญไม่อาจมั่นใจได้ว่าจะเกิดข้อผิดพลาดในสายการผลิตได้อย่างไร แต่การเรียนรู้ของเครื่องเสนอความเป็นไปได้ในการระบุข้อผิดพลาดได้อย่างรวดเร็ว ซึ่งช่วยประหยัดเวลาและค่าใช้จ่าย

ตอนนี้โปรแกรมการเรียนรู้ด้วยตนเองยังถูกใช้ทางการแพทย์ ในอนาคตหลังจากที่ข้อมูลจำนวนมหาศาลจะสามารถเตือนคนไข้ในกรณีที่แพทย์ต้องการสั่งยาที่คนไข้ไม่อาจรับได้หรือตัวยาที่คนไข้เกิดอาการแพ้ในข้อนี้ยังหมายถึงการที่แอปสามารถเสนอทางเลือกอื่นๆ ได้โดยคำนึงถึงข้อจำกัดทางพันธุกรรมของผู้ป่วยนั้นๆ ด้วย

2.4.3 วิธีการที่ถูกใช้ในการเรียนรู้ของเครื่อง

ในการเรียนรู้ของเครื่องวิธีการทางคณิตศาสตร์และสถิติถูกนำมาใช้เพื่อเรียนรู้จากชุดข้อมูล มีอีกหลายสิบวิธีที่แตกต่างกัน ด้วยเหตุนี้ การแยกแยะความแตกต่างโดยทั่วไปสามารถทำได้ระหว่างสองระบบ คือการใช้ แนวทางเชิงสัญลักษณ์ในทางหนึ่งและและ แนวทางสัญลักษณ์ย่อยในอีกทางหนึ่ง ในขณะที่ระบบสัญลักษณ์อีกนัยยะหนึ่งคือระบบเชิงประจักษ์ที่เนื้อหาถูกนำมาแสดงอย่างชัดเจน ส่วน ระบบสัญลักษณ์ย่อยคือ โครงข่ายประสาทเทียม (artificial neuronal networks) ทั้งหมดนี้ทำงานบนหลักการของสมองมนุษย์โดยที่เนื้อหาความรู้จะถูกแสดงอย่างชัดเจน

โครงข่ายประสาทเทียม คือการสร้างคอมพิวเตอร์ที่จำลองเอาวิธีการทำงานของสมองมนุษย์หรือทำให้คอมพิวเตอร์รู้จักคิดและจดจำในแนวเดียวกับโครงข่ายประสาทของมนุษย์ เพื่อช่วยให้คอมพิวเตอร์ฟังภาษามนุษย์ได้เข้าใจ อ่านออก และรู้จำได้ ซึ่งอาจเรียกได้ว่าเป็นสมองกล

2.4.4 ประเภทต่างๆของการเรียนรู้ของเครื่อง

โดยพื้นฐานแล้วอัลกอริธึมมีบทบาทสำคัญในการเรียนรู้ของเครื่องในทางหนึ่งมันมีส่วนในการจัดจํารูปแบบต่างๆ และในทางหนึ่งมันสามารถสร้างโซลูชันต่างๆ ได้ ทั้งนี้อัลกอริทึมสามารถถูกแบ่งออกเป็นหมวดหมู่ต่างๆ ได้ดังนี้

1. การเรียนรู้แบบมีผู้สอน (Supervised learning) จะมีการกำหนดโมเดลตัวอย่างไว้ล่วงหน้า เพื่อให้แน่ใจว่ามีการจัดสรรข้อมูลอย่างเพียงพอกับกลุ่มโมเดลที่เกี่ยวข้องของอัลกอริทึมเหล่านั้น ซึ่งจากนั้นต้องมีการจำเพาะเจาะจง กล่าวอีกนัยหนึ่งระบบจะเรียนรู้บนพื้นฐานของ อินพุต และ เอาท์พุต ที่กำหนดไว้ ทั้งนี้ ตามหลักของเรียนรู้ที่ได้รับการตรวจสอบ โปรแกรมเมอร์ซึ่งทำหน้าที่เป็นเหมือนครูผู้สอนเป็นผู้จัดสรรค่าที่เหมาะสมสำหรับโดยเฉพาะอินพุต เป้าหมายคือการฝึกอบรม ระบบในบริบทของการคำนวณต่อเนื่องกับอินพุตและเอาท์พุต ที่แตกต่างกันและเพื่อสร้างการเชื่อมต่อ

2. การเรียนรู้แบบไม่มีผู้สอน (Unsupervised learning) ปัญญาประดิษฐ์จะเรียนรู้โดยไม่มีค่าเป้าหมายที่กำหนดไว้ล่วงหน้าและไม่มีผลตอบแทน ส่วนใหญ่ใช้เพื่อการแบ่งกลุ่มการเรียนรู้ (clustering) เครื่องจักรพยายามที่จะจัดโครงสร้างและจัดเรียงข้อมูลที่ถูกป้อนเข้ามาตามลักษณะบางอย่าง ตัวอย่างเช่นเครื่องสามารถ เรียนรู้ได้ว่าเหรียญที่มีสีแตกต่างกันสามารถจัดเรียงตามลักษณะสีเพื่อจัดโครงสร้างให้มันได้

3. การเรียนรู้แบบมีผู้สอนบางส่วน (Partially supervised learning) ผู้สอนบางส่วนเป็นความรู้อีกแบบของการรวมกันของการเรียนรู้แบบมีผู้สอนและไม่มีผู้สอน โดยที่ผู้สอนจะไม่สอนอย่างสมบูรณ์นั่นคือ บางข้อมูลในเซตการสอนนั้นขาดข้อมูลขาออก

4. การเรียนรู้แบบส่งเสริม (Encouraging learning) เหมือนกับเงื่อนไขแบบคลาสสิกของสกินเนอร์ คือขึ้นอยู่กับผลตอบแทนและการลงโทษอัลกอริทึมนี้ได้รับการสอนโดยปฏิสัมพันธ์ในเชิงบวกหรือเชิงลบซึ่งจะเกิดปฏิกิริยากับสถานการณ์บางอย่าง

5. การสืบค้นและแสดงผลของข้อมูล (Active learning) ภายใต้ขอบข่ายงานของอัลกอริทึมมีหน้าที่ในการสืบค้นและแสดงผลของข้อมูลขาเข้าที่จำเพาะเจาะจง บนพื้นฐานของคำถามที่กำหนดไว้ซึ่งถือว่าสำคัญ โดยปกติแล้วอัลกอริทึมจะเลือกคำถามที่มีความเกี่ยวข้องสัมพันธ์กันในระดับสูงโดยทั่วไปแล้วฐานข้อมูลเป็นได้ทั้งแบบออฟไลน์หรือแบบออนไลน์ ทั้งนี้ขึ้นอยู่กับระบบที่สอดคล้องกัน

2.4.5 การเรียนรู้ของเครื่องกับแอปพลิเคชันที่ได้รับความนิยมสูงสุด

การเรียนรู้ของเครื่องถูกนำไปใช้ที่เน็ตฟลิกซ์และแอมะซอนรวมถึงการจดจำใบหน้าของเฟซบุ๊ก สำหรับคุณในฐานะผู้ใช้การเรียนรู้ของเครื่องสะท้อนให้เห็นถึงความเป็นไปได้ของการติดแท็กคนในภาพที่อัปโหลดโดยอัตโนมัติความเป็นจริงคือ เฟซบุ๊กมีฐานข้อมูลใบหน้าที่ใหญ่ที่สุดในโลกข้อมูลที่ป้อนโดยผู้ใช้ในเครือข่ายสังคมออนไลน์จะถูกใช้โดย เฟซบุ๊กเพื่อเพิ่มประสิทธิภาพและเทรนระบบการเรียนรู้ของเครื่องในส่วนของการจดจำภาพ

อีกหนึ่งแอปพลิเคชันของการเรียนรู้ของเครื่องที่รวมอยู่ในชีวิตประจำวันอย่างเหนียวแน่นคือการตรวจจับสแปมโดยอัตโนมัติซึ่งถูกรวมเข้ากับโปรแกรมอีเมลเกือบทั้งหมดในขอบเขตของการตรวจจับสแปม ข้อมูลที่มีอยู่ในอีเมลส์จะได้รับการวิเคราะห์และจัดหมวดหมู่รูปแบบสแปมและไม่ใช่สแปมถูกใช้ในส่วนนี้ หากอีเมลล์ถูกจดจำว่าเป็นอีเมลล์ขยะโปรแกรมจะเรียนรู้ว่าจะระบุเมลล์สแปมได้อย่างมีประสิทธิภาพมากยิ่งขึ้นส่วนอื่นๆ ของแอปพลิเคชันอื่นๆ สำหรับการเรียนรู้ของเครื่องได้แก่ การจัดอันดับของเครื่องมือ การต่อสู้กับอาชญากรรมทางไซเบอร์และการป้องกันการโจมตีทางคอมพิวเตอร์

2.4.6 แอปพลิเคชันเพื่อการค้าของการเรียนรู้ของเครื่อง

ด้วยความช่วยเหลือของการเรียนรู้ของเครื่องข้อมูลทางเศรษฐกิจสามารถเปลี่ยนเป็นเงินได้ บริษัทที่พึ่งพาการเรียนรู้ของเครื่องหรือการเรียนรู้ของเครื่องแมตตอดไม่เพียงเพิ่มความพึงพอใจให้กับลูกค้าเท่านั้นแต่ยังสามารถลดต้นทุนได้ด้วย การเรียนรู้ของเครื่องช่วยให้สามารถประเมินความคาดหวังและความต้องการของลูกค้าและสามารถกำหนดมาตรการด้านการตลาดต่อได้ นำไปสู่ประสบการณ์เชิงบวกของลูกค้าและเพิ่มความภักดีของลูกค้าในที่สุด

นอกจากนี้ยังมีการใช้แชทบอท (chat bots) มากขึ้นในพื้นที่ของการบริการลูกค้าทางโทรศัพท์ เหล่านี้เป็นโปรแกรมอัตโนมัติที่สื่อสารกับลูกค้าด้วยวิธีนี้แชทบอทสามารถเพิ่มประสิทธิภาพความสามารถด้านความรู้ความเข้าใจเกี่ยวกับการตีความโทนเสียงในสถานการณ์ต่างๆ นอกจากนี้ แชทบอทยังสามารถส่งต่อสายได้เช่น กรณีที่เป็นคำร้องขอที่ซับซ้อนมากขึ้นไปยังพนักงานของศูนย์บริการ

นอกจากนี้การเรียนรู้ของเครื่องยังเป็นเทคโนโลยีสำคัญในการพัฒนาระบบอิสระ นอกเหนือจากรไคนซ์แล้วการเรียนรู้ของเครื่องยังถูกใช้ในหุ่นยนต์ ส่วนอื่นๆ ของแอปพลิเคชันสำหรับการเรียนรู้ของเครื่องจะเป็น:

1. การวิเคราะห์ตลาดหุ้น
2. การตรวจสอบการฉ้อโกงบัตรเครดิตขึ้นตอน
3. การวินิจฉัยโดยอัตโนมัติ
4. การได้มาซึ่งทุนระเบิดในเซ็นเซอร์เสียงและข้อมูลเรดาร์

2.4.7 การเรียนรู้ของเครื่องกับผู้นำเทคโนโลยี

นอกเหนือจาก ไมโครซอฟท์ (Microsoft), กูเกิล (Google), เฟสบุ๊ก (Facebook), ไอบีเอ็ม (IBM), แอมะซอน (Amazon) และ แอปเปิ้ล (Apple) ยังใช้ทรัพยากรทางการเงินมหาศาลในการใช้และพัฒนาศาสตร์การเรียนรู้ของเครื่องต่อไปซูเปอร์คอมพิวเตอร์ เวสตัน (Watson) ของไอบีเอ็มยังคงเป็นเครื่องมือที่รู้จักกันดีสำหรับการเรียนรู้ของเครื่องส่วนใหญ่วัตถุดิบจะถูกใช้ในภาคการแพทย์และการเงินตามที่กล่าวมาแล้ว เฟสบุ๊กใช้การเรียนรู้ของเครื่องเพื่อการรับรู้ภาพ ไมโครซอฟท์ ใช้สำหรับระบบจดจำเสียงพูดคอร์ตานา (Cortana), แอปเปิ้ล สำหรับสิริ (Siri) แน่นอนว่าการเรียนรู้ของเครื่อง ถูกใช้งานที่ กูเกิล ด้วยทั้งในด้านการบริการภาพและเครื่องมือค้นหาขั้นสูง ผู้ให้บริการระบบคลาวด์เช่น กูเกิล, ไมโครซอฟท์, แอมะซอน บริการเว็บ (Webservice) และ ไอบีเอ็ม ได้สร้างบริการสำหรับการเรียนรู้ของเครื่องขึ้นแล้ว ด้วยความช่วยเหลือของพวกเขามีความเป็นไปได้ว่านักพัฒนาซอฟต์แวร์ที่ไม่มีความรู้

เกี่ยวกับการเรียนรู้ของเครื่องโดยเฉพาะจะพัฒนาแอปพลิเคชันต่อไปได้ แอปพลิเคชันเหล่านี้สามารถเรียนรู้จากชุดข้อมูลที่กำหนดได้อย่างอิสระ แพลตฟอร์มเหล่านี้มีชื่อแตกต่างกันขึ้นอยู่กับผู้ให้บริการ เช่น

1. ไอบีเอ็ม: เวสตัน
2. แอมะซอน: แอมะซอน การเรียนรู้ของเครื่อง
3. ไมโครซอฟท์: เอชอน การเรียนรู้ของเครื่อง สตูดิโอ (Azure ML Studio)
4. กูเกิล: เท็นเซอร์โฟล (TensorFlow)

นอกจากแพลตฟอร์มดังกล่าวข้างต้นแล้ว ยังมีโปรแกรมโอเพ่นซอร์สฟรี (free open source) ที่มีคุณภาพสูงซึ่งทำให้การเรียนรู้ของเครื่องสามารถเข้าถึงผู้ชมได้กว้างขวางขึ้น ช่วยให้นักพัฒนาซอฟต์แวร์หรือผู้เชี่ยวชาญด้านข้อมูลสามารถทำงานร่วมกันได้

2.5 วิชชนสตูดิโอโค้ด (Visual Studio Code)

วิชชนสตูดิโอโค้ด [5] จากบริษัทไมโครซอฟท์ เป็นโปรแกรมประเภท บรรณาธิการ (Editor) ใช้ในการแก้ไขโค้ดที่มีขนาดเล็ก แต่มีประสิทธิภาพสูง เป็น โอเพ่นซอร์สโปรแกรมจึงสามารถนำมาใช้งานได้โดยไม่ค่าใช้จ่าย เหมาะสำหรับนักพัฒนาโปรแกรมที่ต้องการใช้งานหลายแพลตฟอร์ม รองรับการใช้งานทั้งบน ระบบปฏิบัติการวินโดวส์ , ระบบปฏิบัติการแมค (mac Os) และ ระบบปฏิบัติการลินุกซ์ (Linux Os) รองรับหลายภาษาทั้ง จาวาสคริปต์ (Java Script), ไพทอน และ โนดเจส (Node.js) ในตัว และสามารถเชื่อมต่อกับกิต (Git) ได้ง่าย สามารถนำมาใช้งานได้ง่ายไม่ซับซ้อน มีเครื่องมือและส่วนขยายต่างๆ ให้เลือกใช้มากมาย รองรับการใช้งานภาษาอื่นๆ ทั้ง ภาษาซี++, ซี#, จาวา, ไพทอน , พีเอชพี (PHP) หรือ โก (Go) สามารถปรับเปลี่ยนธีมได้มีส่วนดีบั๊กเกอร์ (Debugger) และคอมมานด์ (Commands) เป็นต้น มีสัญลักษณ์ดังรูปที่ 2.5



รูปที่ 2.5 สัญลักษณ์วิชชนสตูดิโอโค้ด

(ที่มา : <https://www.mindphp.com>)

2.6 ภาษาไพทอน

ภาษาไพทอน [6] คือภาษาโปรแกรมคอมพิวเตอร์ระดับสูง โดยถูกออกแบบมาให้เป็นภาษาสคริปต์ที่อ่านง่าย โดยตัดความซับซ้อนของโครงสร้างและไวยากรณ์ของภาษาออกไป ในส่วนของการแปลงชุดคำสั่งที่เขียนให้เป็นภาษาเครื่อง ไพทอนมีการทำงานแบบ ล่าม (Interpreter) คือเป็นการแปลชุดคำสั่งทีละบรรทัด เพื่อป้อนเข้าสู่หน่วยประมวลผลให้คอมพิวเตอร์ทำงานตามที่ต้องการ นอกจากนั้นภาษาโปรแกรมไพทอน ยังสามารถนำไปใช้ในการเขียนโปรแกรมได้หลากหลายประเภท โดยไม่ได้จำกัดอยู่ที่งานเฉพาะทางใดทางหนึ่ง (General-purpose language) จึงทำให้มีการนำไปใช้กันแพร่หลายในหลายองค์กรใหญ่ระดับโลก เช่น กูเกิล, ยูทูบ (YouTube), (Instagram), ดริอปบ็อก (Dropbox) และ นาซา (NASA) เป็นต้น มีสัญลักษณ์ดังรูปที่ 2.6



รูปที่ 2.6 สัญลักษณ์ภาษาไพทอน
(ที่มา : <https://www.python.org>)

2.7 เจสัน (Json)

เจสัน [7] คือ รูปแบบของข้อมูลที่ใช้สำหรับแลกเปลี่ยนข้อมูลที่มีขนาดเล็ก ซึ่งคนสามารถทำความเข้าใจได้ง่ายและสามารถถูกสร้างและอ่านโดยเครื่องได้ง่าย มันถูกกำหนดภายใต้ภาษา จาวาสคริปต์ เจสัน เป็นรูปแบบข้อมูลตัวอักษรที่มีความเป็นอิสระอย่างสมบูรณ์ แต่จะมีหลักการการเขียนที่คุ้นเคยกับนักเขียนโปรแกรมภาษาต่างๆ ได้ ไม่ว่าจะเป็น ภาษา ซี, ซี++, ซี#, จาวา, จาวาสคริปต์, เพอร์ (Perl), ไพทอน และอื่นๆ คุณสมบัติเหล่านี้ทำให้ เจสัน เป็นภาษาแลกเปลี่ยนข้อมูลที่มีสมบูรณ์แบบ ดังรูปที่ 2.7



รูปที่ 2.7 Json

(ที่มา : <https://www.boxsingle.com>)

2.8 ลิโบรซ่า (librosa)

ลิโบรซ่า [8] เป็นแพ็คเกจไพทอนใช้สำหรับการวิเคราะห์เพลงและเสียง สามารถสร้างระบบดึงข้อมูลเพลงได้ โดยลิโบรซ่าจะวิเคราะห์เพลงและเสียงที่อยู่ในรูปแบบของไฟล์ .wav แล้วแปลงสัญญาณให้อยู่ในรูปของสัญญาณคลื่นเสียง (spectrogram) ซึ่ง สเปกโตรแกรมเป็นกลุ่มของ FFT ที่ซ้อนกัน เป็นการแสดงความดังของสัญญาณหรือแอมพลิจูดของสัญญาณด้วยภาพ จะแตกต่างกันไปตามความถี่ที่ต่างกันตามเวลา เมื่อคำนวณสเปกโตรแกรม แกน y จะถูกแปลงเป็นขนาดลอการิทึม (log scale) และมีมิติสี (color dimension) จะถูกแปลงเป็นเดซิเบล เนื่องจากมนุษย์สามารถรับรู้ช่วงความถี่และแอมพลิจูดที่มีขนาดเล็กและเข้มข้นเท่านั้น (fast Fourier transform: FFT) คือ สูตรทางคณิตศาสตร์ที่ช่วยให้เราสามารถแยกสัญญาณออกเป็นความถี่เดียวและแอมพลิจูดของความถี่ได้ กล่าวคือจะแปลงสัญญาณจากโดเมนเวลาเป็นโดเมนความถี่ ผลลัพธ์เรียกว่าสเปกตรัม ดังรูปที่ 2.8



รูปที่ 2.8 ลิโบรซ่า

(ที่มา : <https://www.borntodev.com>)

2.9 นัมพาย (NumPy)

นัมพาย [9] เป็นไลบรารี (library) ที่นิยมใช้ในการคำนวณเช่น ใช้คำนวณเมทริกซ์ (Matrix) หรือ คำนวณกับอาร์เรย์ ในงานข้อมูลศาสตร์, ข้อมูลการวิเคราะห์และในการทำการเรียนรู้ของเครื่อง หรือ การเรียนรู้เชิงลึก ยังคงต้องใช้ นัมพายซึ่งนัมพายเป็นไลบรารี พื้นฐานที่ใช้คำนวณทางคณิตศาสตร์ด้วยภาษาไพทอนสามารถคำนวณ หรือ ดำเนินการทางตรรกะในอาร์เรย์ หลายมิติหรือเมทริกซ์ได้อย่างรวดเร็วเพราะไลบรารี เขียนด้วยภาษาซีที่มีการบันทึกไว้แล้วและไม่มีการวนลูปซ้ำในชุดโค้ดทำให้รหัสแวกเตอร์มีความกระชับและอ่านง่ายขึ้น มีสัญลักษณ์ดังรูปที่ 2.9



รูปที่ 2.9 นัมพาย

(ที่มา : <https://www.borntodev.com>)

2.10 แมทซ์ (Math)

แมทซ์ [10] เป็นฟังก์ชันที่ใช้คำนวณทางคณิตศาสตร์ เพื่อหาค่าทางคณิตศาสตร์ โดยฟังก์ชันแมทซ์ ในไพทอนนี้จะใช้ฟังก์ชันทางคณิตศาสตร์ที่กำหนดโดยมาตรฐานภาษาซีและฟังก์ชัน แมทซ์ นั้นยังสามารถคำนวณได้ตั้งแต่สูตรพื้นฐานทางคณิตศาสตร์จนถึงการคำนวณระดับสูงที่ใช้คณิตศาสตร์เช่น การคำนวณทางด้านการเรียนรู้ของเครื่องหรือการเรียนรู้เชิงลึก ดังรูปที่ 2.10

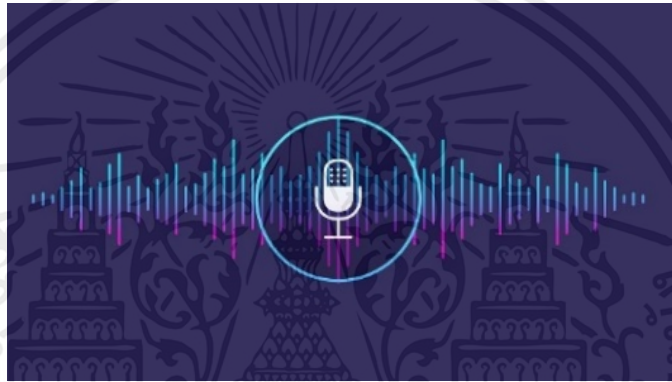


รูปที่ 2.10 แมทซ์

(ที่มา : <https://python3.wannaphong.com>)

2.11 อุปกรณ์รับเสียง (sounddevice)

อุปกรณ์รับเสียง [11] เป็นไลบรารีของภาษาไพทอนที่สามารถทำงานร่วมกับนัมพาย ซึ่งเป็นไลบรารีที่ใช้ในการอัดเสียงบันทึกเสียงและเล่นเสียงทั้งเบื้องหน้าและเบื้องหลัง ซึ่งในการทำงานร่วมกับนัมพายนั้น อุปกรณ์รับเสียงสามารถที่จะรับค่าเสียงแล้วบันทึกอยู่ในรูปแบบของค่าอาร์เรย์แล้วนำไปใช้งานต่อได้ คือสามารถที่จะนำไปเปรียบเทียบค่าของพีคเจอร์ที่อยู่ในระบบการเรียนรู้เชิงลึกได้ ดังรูปที่ 2.11



รูปที่ 2.11 อุปกรณ์รับเสียง

(ที่มา : <https://python-sounddevice.readthedocs.io>)

2.12 ซาวน์ไฟล์ (soundfile)

ไฟล์เสียง [12] สามารถอ่านและเขียนไฟล์เสียงได้ การอ่านเขียนไฟล์ได้รับการสนับสนุนผ่าน libsndfile ซึ่งเป็นไลบรารีโอเพ่นซอร์ส (LGPL) ฟรีข้ามแพลตฟอร์มสำหรับการอ่านและเขียนรูปแบบไฟล์เสียงตัวอย่างต่างๆ มากมายที่ทำงานบนหลายแพลตฟอร์ม รวมถึง ระบบปฏิบัติการวินโดวส์, โอเอส เอ็กซ์ (OS X) และยูนิก (Unix) สามารถเข้าถึงได้ผ่าน CFFI ซึ่งเป็นอินเทอร์เฟซฟังก์ชันต่างประเทศสำหรับไพทอน เรียกรหัส ซี CFFI รองรับ ซี,ไพทอน 2.6+, 3.x และ PyPy 2.0+ ไฟล์เสียง แสดงข้อมูลเสียงเป็นอาร์เรย์ นัมพาย มีสัญลักษณ์ดังรูปที่ 2.12



รูปที่ 2.12 สัญลักษณ์ซาวน์ไฟล์

(ที่มา : <https://pysoundfile.readthedocs.io>)

2.13 กี้วี่ (Kivy)

กี้วี่ [13] คือไลบรารีชนิดหนึ่งในภาษาไพทอนที่ใช้สำหรับการพัฒนาแอปพลิเคชันที่สามารถติดต่อสื่อสารกับผู้ใช้งานได้ เนื่องจากการออกแบบที่สะดวกเรียบง่ายจึงทำให้กี้วี่ได้รับความนิยมสำหรับการใช้พัฒนาแอปพลิเคชันด้วยภาษาไพทอนเป็นอย่างมาก มีลักษณะเป็นแบบข้ามแพลตฟอร์มใช้งานได้ฟรีสามารถใช้พัฒนาโปรแกรมและใช้ในธุรกิจได้โดยไม่ต้องเสียค่าลิขสิทธิ์ ออกแบบกราฟฟิกด้วย OpenGL ES 2 ทำให้มีความรวดเร็วและดูทันสมัย มีอุปกรณ์ให้เลือกมากกว่า 20 วิดเจ็ต จึงทำให้ออกแบบงานที่ต้องการได้หลากหลายและสามารถสร้างแอปพลิเคชันสำหรับใช้งานบนระบบปฏิบัติการวินโดวส์, แมคโอเอส, ลินุกซ์, แอนดรอยด์และไอโอเอสโดยการใช้วิดเจ็ตจากกี้วี่และสิ่งที่พิเศษที่สุดสำหรับกี้วี่คือการรองรับระบบมัลติทัช มีสัญลักษณ์ดังรูปที่ 2.13



รูปที่ 2.13 สัญลักษณ์ของกี้วี่

(ที่มา : <https://stackpython.medium.com>)

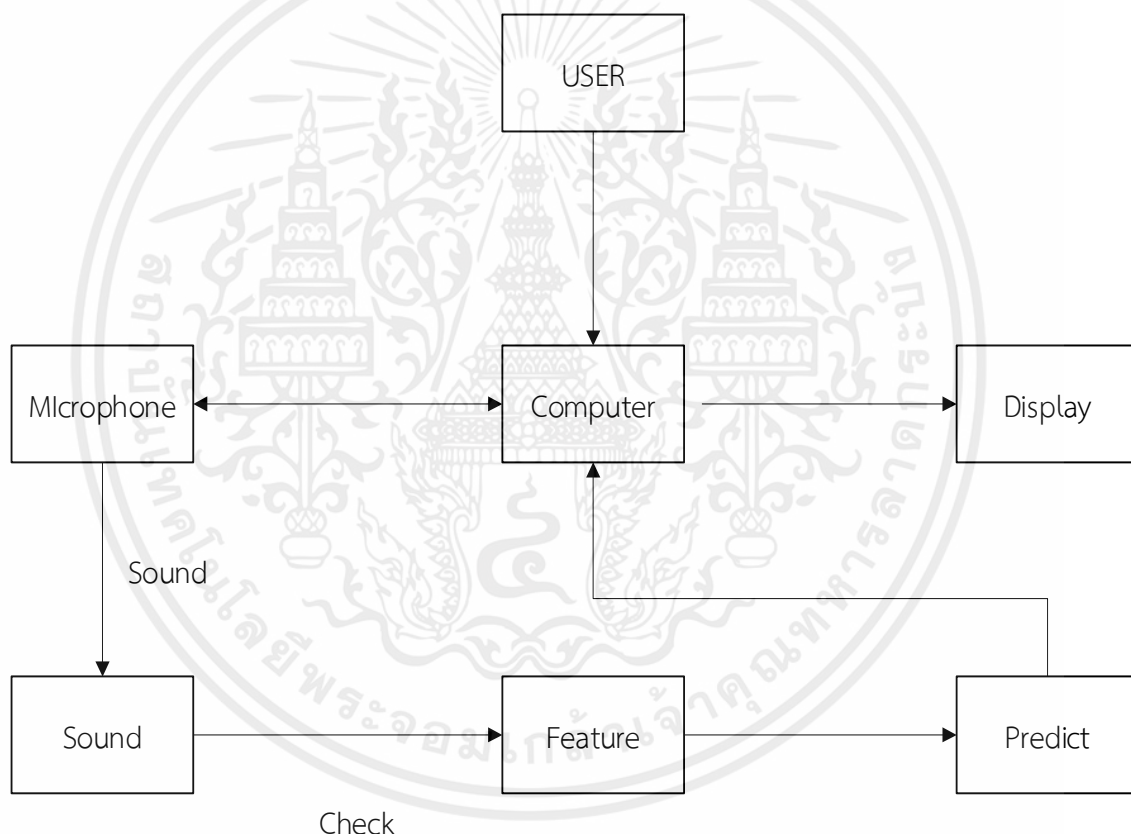
บทที่ 3

การออกแบบ

ในส่วนบทนี้กล่าวถึง การออกแบบ การทำงานของแอปพลิเคชันโดยได้ดำเนินการตามขั้นตอนต่อไปนี บล็อกไดอะแกรมของส่วนต่าง ๆ ในโครงการ การออกแบบโครงสร้าง การออกแบบแอปพลิเคชัน

3.1 บล็อกไดอะแกรม

3.1.1 บล็อกไดอะแกรมการออกแบบแอปพลิเคชันช่วยประเมินโรคจากอาการไอ



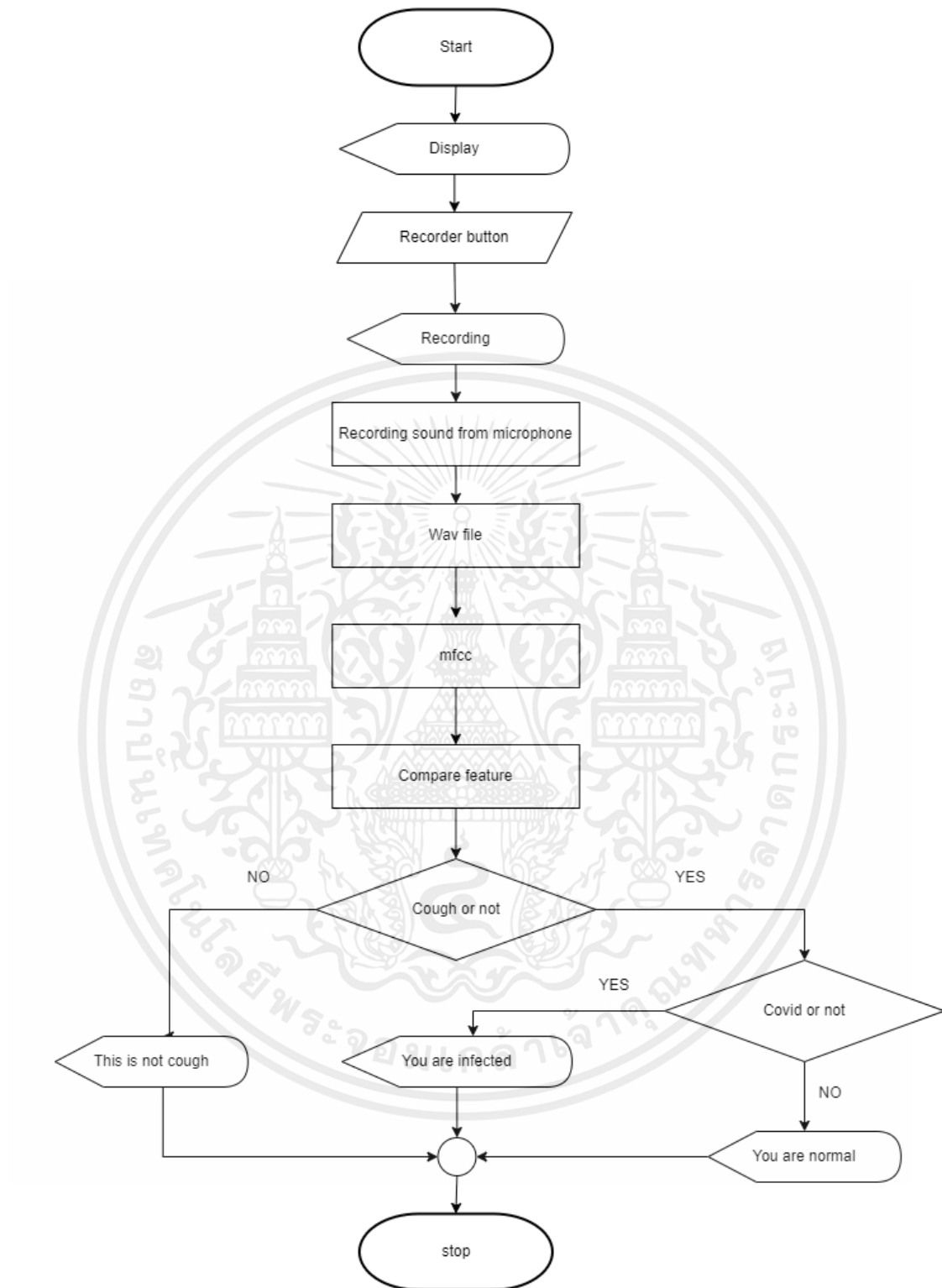
รูปที่ 3.1 บล็อกไดอะแกรมการออกแบบแอปพลิเคชันช่วยประเมินโรคจากอาการไอ

จากรูปที่ 3.1 ผู้จัดทำสามารถอธิบายรายละเอียดบล็อกไดอะแกรมของแอปพลิเคชันช่วยประเมินโรคจากอาการไอได้ดังนี้

1. ผู้ใช้ทั่วไป (User) กดใช้งานแอปพลิเคชันบนสมาร์ตโฟน โดยกดเปิดแอปขึ้นมาผ่านระบบปฏิบัติการวินโดวบนคอมพิวเตอร์
2. เมื่อผู้ใช้ทั่วไปกดเปิดแอปพลิเคชันขึ้นมาแล้วกดที่ปุ่ม Record เพื่ออัดเสียงโอ โดยกดที่ปุ่มอัดเสียงและอัดเสียงผ่านไมโครโฟน (Microphone) บนคอมพิวเตอร์
3. เมื่อผู้ใช้ทั่วไปอัดเสียงโอ (Record) เสร็จ ตัวระบบการเรียนรู้เชิงลึกของเสียง จะทำการประมวลผลเพื่อวิเคราะห์และประเมินโรคเบื้องต้น โดยอัลกอริทึมภายในจะทำการเปรียบเทียบเสียง (Sound) กับฟีเจอร์ (Feature) แล้วประมวลผล (Predict) ออกมา
4. ตัวระบบการเรียนรู้เชิงลึกของเสียง จะส่งข้อมูลกลับมาบนตัวแอปพลิเคชันแสดงผลผ่านหน้าจอ (Display) เพื่อให้ผู้ใช้ทั่วไป นั้นทราบข้อมูลว่าเป็นโรค โรคโควิด-19 หรือไม่

3.2 แผนผังการทำงานแอปพลิเคชัน

จากรูปที่ 3.2 เป็นรูปโฟรชาตที่เป็นหลักการทำงานของแอปพลิเคชัน โดยเริ่มจากเปิดแอปพลิเคชันขึ้นมา จะมีหน้าต่างแอปพลิเคชันแสดงขึ้นมา ผู้ใช้งานทำการอัดเสียงด้วยการกดปุ่ม Record เพื่ออัดเสียงผ่านไมโครโฟนบนคอมพิวเตอร์ หลังจากนั้นระบบจะทำการบันทึกเสียงของผู้ใช้งานอยู่ในรูปแบบของ .wav file แล้วระบบจะทำการแปลงเสียงจาก .wav file เป็น ค่าสัมประสิทธิ์ใช้ปรัลความถี่เมลเพื่อถอดฟีเจอร์ของเสียงโอของผู้ใช้งาน หลังจากนั้นระบบจะทำการเปรียบเทียบฟีเจอร์ของเสียงโอผู้ใช้งานกับเสียงโอโควิดว่าตรงกันหรือไม่ ระบบจะทำการตรวจสอบและประมวล จากนั้นระบบจะแสดงผลการทดสอบมายังหน้าจอแอปพลิเคชัน จบการทำงาน



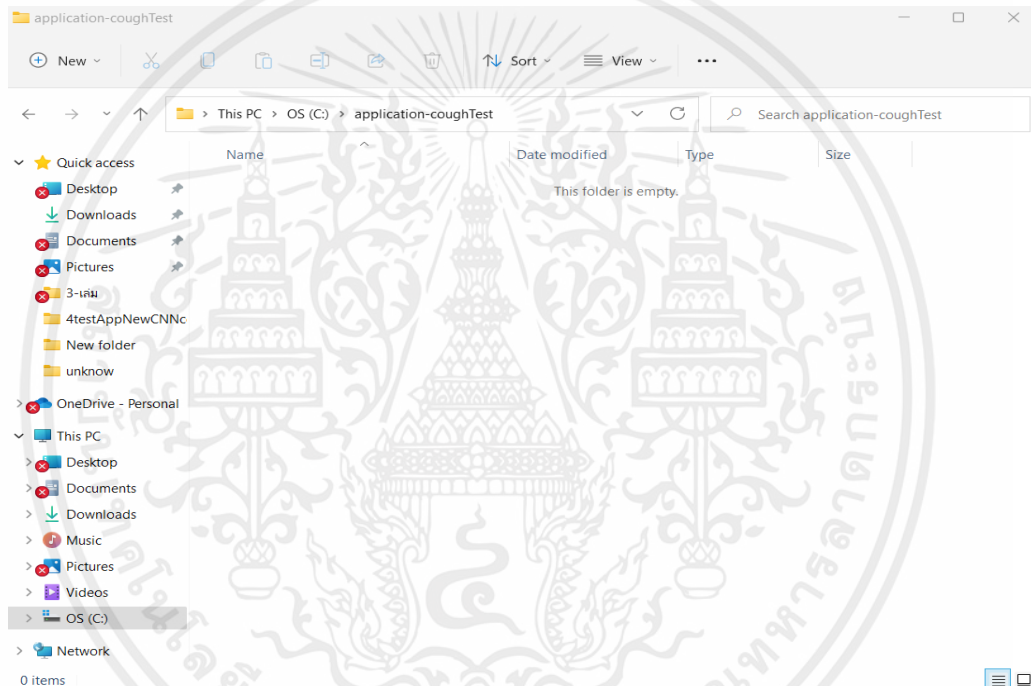
รูปที่ 3.2 เป็นรูปแผนผังการทำงานของแอปพลิเคชัน

3.3 การออกแบบสร้างชุดข้อมูลสำหรับเทรน

การสร้างแอปพลิเคชันช่วยประเมินโรคจากอาการไอประกอบไปด้วยชุดการทำงานของโปรแกรมหลาย ๆ ส่วนเข้าด้วยกัน เพื่อให้แอปพลิเคชันสามารถใช้งานได้สะดวกและมีประสิทธิภาพ โดยคำนึงถึงการใช้งานบนคอมพิวเตอร์ที่มีการติดตั้งโปรแกรมที่ใช้เขียนและอ่านภาษาไพทอนแล้วจะต้องมีการติดตั้งไลบรารีที่ใช้งาน มีขั้นตอนดังนี้

3.3.1 สร้างโฟลเดอร์ของแอปพลิเคชัน

สร้างโฟลเดอร์ของแอปพลิเคชันที่จะสร้างในตำแหน่งที่ต้องการบนคอมพิวเตอร์

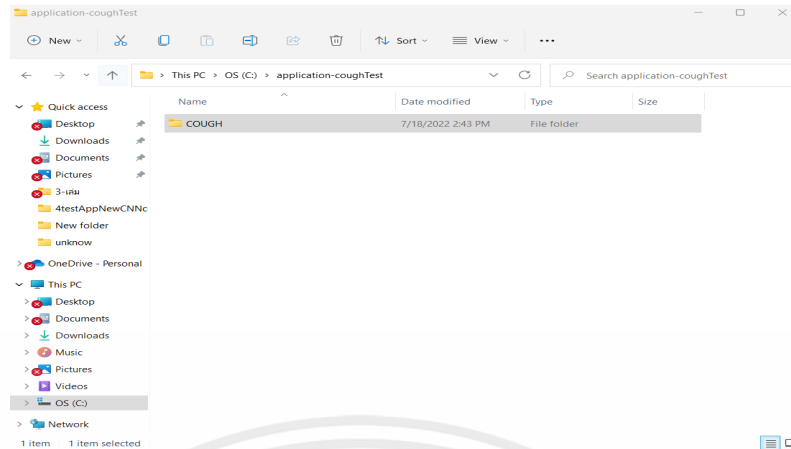


รูปที่ 3.3 โฟลเดอร์ที่สร้างเพื่อเป็นที่อยู่ของแอปพลิเคชัน

จากรูปที่ 3.3 เริ่มต้นสร้างไฟล์เดอร์เพื่อสร้างเป็นไฟล์ที่อยู่ของแอปพลิเคชัน โดยเริ่มจากไปที่หน้าเดสทอปแล้วคลิกขวา เลือก New Folder

3.3.2 สร้างโฟลเดอร์สำหรับเสียงไอ

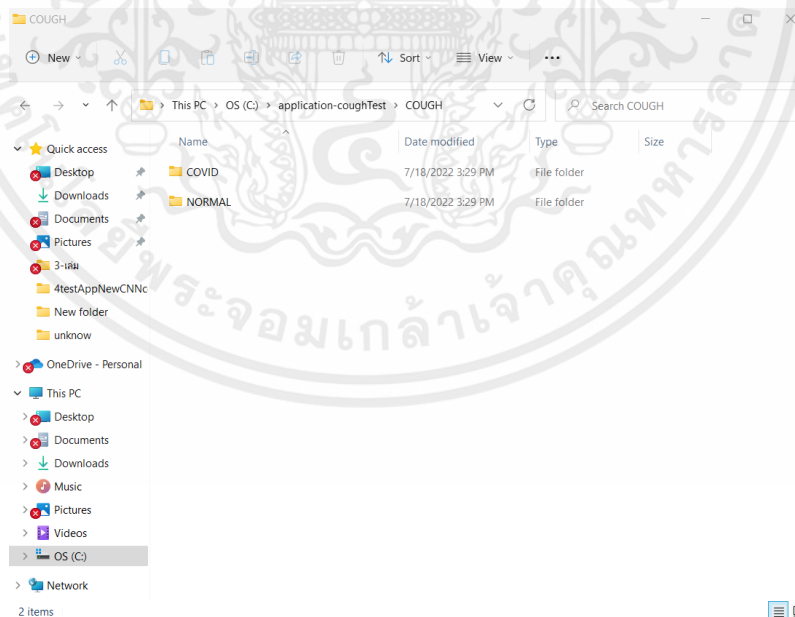
1. สร้างโฟลเดอร์ขึ้นมาโดยใช้ชื่อว่า “COUGH” ภายในโฟลเดอร์ของแอปพลิเคชัน



รูปที่ 3.4 โฟลเดอร์ที่สร้างเพื่อเป็นที่อยู่ของเสียงไอ

จากรูที่ 3.4 เข้าไปยังโฟลเดอร์แอปพลิเคชัน จากนั้นกดสร้างโฟลเดอร์ใหม่สำหรับเก็บเสียงไอพร้อมกับตั้งชื่อ “COUGH”

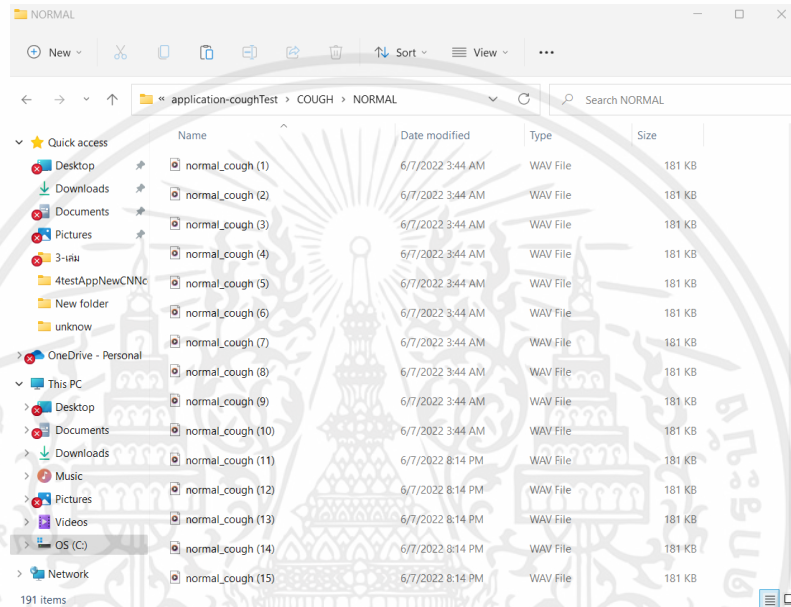
2. เข้าไปภายในโฟลเดอร์ “COUGH” แล้วสร้างโฟลเดอร์อีก 2 โฟลเดอร์ภายในโดยใช้ชื่อว่า “COVID” สำหรับเก็บเสียงไอโควิดและ “NORMAL” สำหรับเก็บเสียงไอปกติ



รูปที่ 3.5 โฟลเดอร์ที่สร้างเพื่อเป็นที่อยู่ของเสียงไอแต่ละประเภท

จากรูปที่ 3.5 ในไฟล์เดอร์แอปพลิเคชัน สร้างไฟล์เดอร์เก็บเสียงไอที่มีชื่อว่า “COUGH” และ ภายในไฟล์เดอร์นี้ สร้างไฟล์เสียงไอโควิดชื่อว่า “COVID” ไฟล์เดอร์เก็บเสียงไอปกติชื่อว่า “NORMAL”

3. นำเสียงที่ต้องการเข้าสู่กระบวนการเทรนฝึกอบรม ใส่ลงในไฟล์เดอร์ให้ถูกต้องตามประเภทของเสียงไอ โดยจะต้องเป็นเสียงไอขนาด 1 วินาทีและมีสกุลไฟล์ wav



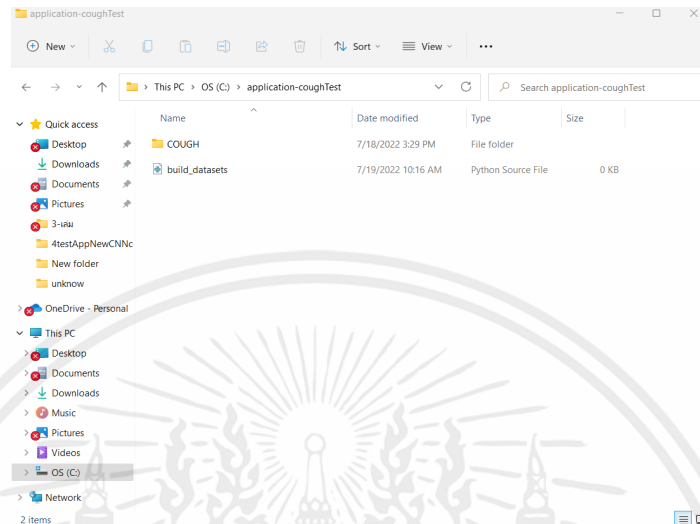
รูปที่ 3.6 ไฟล์เดอร์ที่สร้างเพื่อเป็นที่ย่อยของชุดข้อมูลที่ต้องการเทรน

จากรูปที่ 3.6 นำไฟล์เสียงที่ค้นหาได้ไปใส่ไว้ในไฟล์เดอร์แต่ละประเภท เช่น เสียงไอโควิด นำไฟล์เสียง .wav ไปใส่ไว้ในไฟล์เดอร์ “COVID” เพื่อเตรียมไว้สร้างชุดข้อมูล

3.3.3 เขียนโปรแกรมสร้างชุดข้อมูลที่ใช้ในการเทรนด้วยภาษาไพทอน

สร้างชุดโปรแกรมสำหรับใช้ในกระบวนการฝึกอบรมข้อมูลไฟล์เสียงไอโควิดและเสียงไอปกติ ชุดข้อมูลที่น่าเข้าสู่กระบวนการฝึกอบรมเป็นส่วนสำคัญในการประมวลผลและทำนายผล ไฟล์เสียงไอที่นำมาจะต้องผ่านการคัดกรองไม่มีเสียงอื่นใดแทรกเพื่อให้แอปพลิเคชันประเมินผลได้อย่างมีประสิทธิภาพ

1. สร้างไฟล์นามสกุลไพทอนแล้วให้ทำการบันทึกไฟล์ลงในโฟลเดอร์แอปพลิเคชัน (application folder) โดยใช้ชื่อว่า “build_ชุดข้อมูล”



รูปที่ 3.7 ไฟล์สกุลไพทอนที่ใช้เขียนโปรแกรมสร้างชุดข้อมูลสำหรับการเทรน

จากรูปที่ 3.7 สร้างไฟล์นามสกุลไพทอน เพื่อสร้างไฟล์ไพทอนสำหรับเขียนโค้ด ในโฟลเดอร์แอปพลิเคชัน

2. เรียกใช้ไลบรารี(ไลบรารี)ที่จำเป็นในการสร้างชุดข้อมูล

```
1 import json
2 import os
3 import math
4 import librosa
```

รูปที่ 3.8 ไลบรารี ที่จำเป็นในการใช้งานสร้างชุดข้อมูล

จากรูปที่ 3.8 เรียกใช้ไลบรารีที่จำเป็นในการใช้งานสร้างชุดข้อมูลไฟล์ไพทอนที่สร้างขึ้น เพื่อสร้างชุดข้อมูลสำหรับการเทรน

3. เขียนโปรแกรมการทำงานในส่วนต่าง ๆ เพื่อสร้างชุดข้อมูลเป็นสกุลไฟล์เจสัน

```

5
6 DATASET_PATH = "COUGH"
7 JSON_PATH = "datasets.json"
8
9 SAMPLE_RATE = 22050
10 TRACK_DURATION = 1
11 num_segments=1
12 SAMPLES_PER_TRACK = SAMPLE_RATE * TRACK_DURATION
13
14
15 def save_mfcc(dataset_path, json_path, num_mfcc=13, n_fft=2048, hop_length=512):
16
17     data = {
18         "mapping": [],
19         "labels": [],
20         "mfcc": []
21     }
22
23     samples_per_segment = int(SAMPLES_PER_TRACK / num_segments)
24     num_mfcc_vectors_per_segment = math.ceil(samples_per_segment / hop_length)
25
26     for i, (dirpath, dirnames, filenames) in enumerate(os.walk(dataset_path)):
27
28         if dirpath is not dataset_path:
29
30             semantic_label = dirpath.split("/")[-1]
31             data["mapping"].append(semantic_label)
32             print("\nProcessing: {}".format(semantic_label))
33
34             for f in filenames:
35
36                 file_path = os.path.join(dirpath, f)
37                 signal, sample_rate = librosa.load(file_path, sr=SAMPLE_RATE)
38
39                 for d in range(num_segments):
40
41                     start = samples_per_segment * d
42                     finish = start + samples_per_segment
43
44                     if finish == start + samples_per_segment:
45
46                         mfcc = librosa.feature.mfcc(signal[start:finish], sample_rate, n_mfcc=num_mfcc, n_fft=n_fft, hop_length=hop_length)
47                         mfcc = mfcc.T
48
49                         if len(mfcc) == num_mfcc_vectors_per_segment:
50                             data["mfcc"].append(mfcc.tolist())
51                             data["labels"].append(i-1)
52                             print("{} segment: {}".format(file_path, d+1))
53
54             with open(json_path, "w") as fp:
55                 json.dump(data, fp, indent=4)
56
57 if __name__ == "__main__":
58     save_mfcc(DATASET_PATH, JSON_PATH)

```

รูปที่ 3.9 ชุดคำสั่งต่าง ๆ ที่ใช้สร้างชุดข้อมูลสำหรับการเทรน

จากรูปที่ 3.9 สร้างชุดคำสั่งต่าง ๆ ที่ใช้สร้างชุดข้อมูลสำหรับการเทรน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4. บันทึกและรันโปรแกรม

```

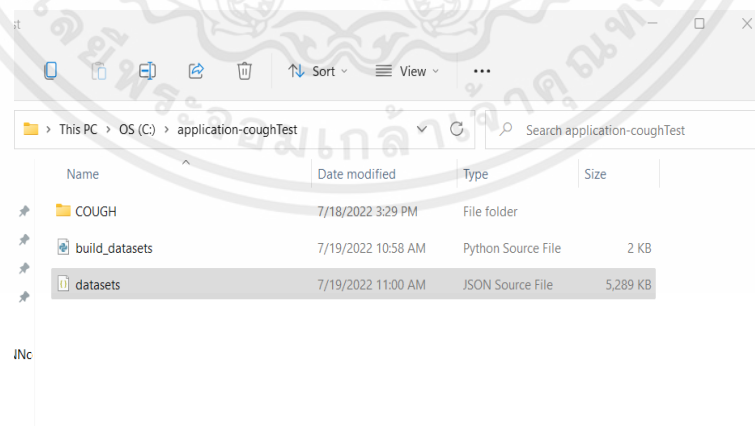
EXPLORER
...
build_datasets.py X
v APPLICATION-COUGHTEST
> COUGH
  build_datasets.py
  datasets.json
1 import json
2 import os
3 import math
4 import librosa
5
6 DATASET_PATH = "COUGH"
7 JSON_PATH = "datasets.json"
8
9 SAMPLE_RATE = 22050
10 TRACK_DURATION = 1
11 num_segments=1
12 SAMPLES_PER_TRACK = SAMPLE_RATE * TRACK_DURATION
13
14
15 def save_mfcc(dataset_path, json_path, num_mfcc=13, n_fft=2048, hop_length=512):
16
17     ion 0.10 passing these as positional arguments will result in an error
18     mfcc = librosa.feature.mfcc(signal[start:finish], sample_rate, n_mfcc=num_mfcc, n_fft=n_fft, hop_length=
19     COUGH\NORMAL\normal_cough (94).wav, segment:1
20     c:\application-coughTest\build_datasets.py:46: FutureWarning: Pass y=[ 0.          0.          0.
21     0.00052592], sr=22050 as keyword args. From version 0.10 passing these as positional arguments will resu
22     mfcc = librosa.feature.mfcc(signal[start:finish], sample_rate, n_mfcc=num_mfcc, n_fft=n_fft, hop_length=
23     COUGH\NORMAL\normal_cough (95).wav, segment:1
24     c:\application-coughTest\build_datasets.py:46: FutureWarning: Pass y=[0.0000000e+00 0.0000000e+00 0.0000000
25     7.5244556e-05], sr=22050 as keyword args. From version 0.10 passing these as positional arguments will re
26     mfcc = librosa.feature.mfcc(signal[start:finish], sample_rate, n_mfcc=num_mfcc, n_fft=n_fft, hop_length=
27     COUGH\NORMAL\normal_cough (96).wav, segment:1

```

รูปที่ 3.10 การรันโปรแกรม

จากรูปที่ 3.10 ทดสอบรันโปรแกรมที่สร้างขึ้น เพื่อดูบัคในชุดโค้ด หากมีจุดผิดพลาดต้องทำการแก้ไข

5. ชุดข้อมูลสำหรับการเทรนที่ได้เป็นไฟล์สกุล. เจสัน

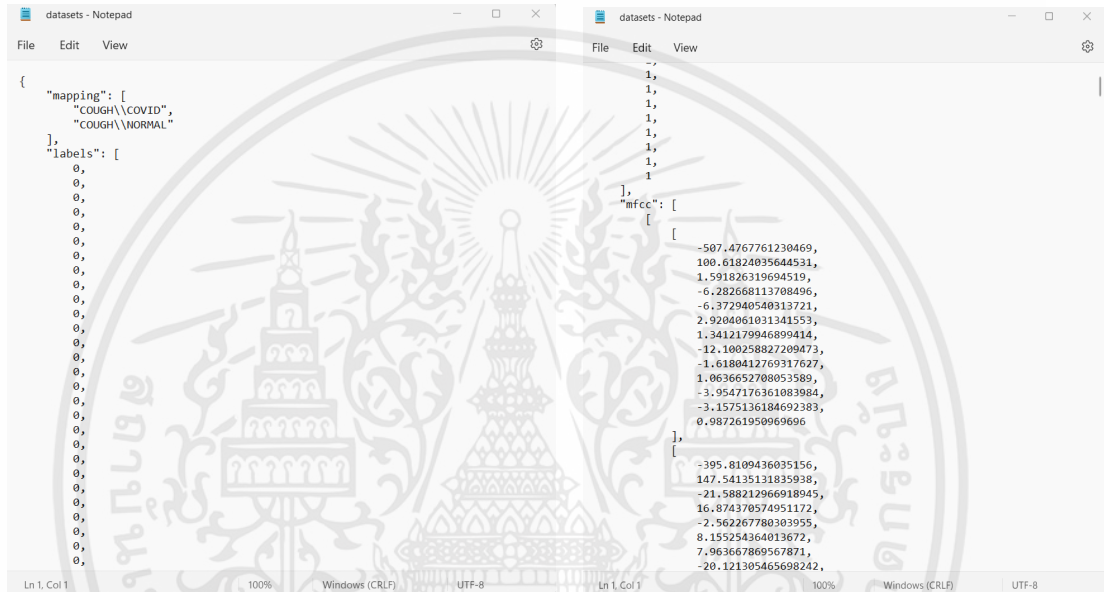


รูปที่ 3.11 ชุดข้อมูลสำหรับการเทรน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากรูปที่ 3.11 จากการสร้างชุดข้อมูลนำชุดข้อมูลมาบันทึกเป็นไฟล์ .เจสัน เพื่อนำมาถอดฟิจเจอร์ในไฟล์เสียง ซึ่งอยู่ในรูปแบบของอาร์เรย์ ออกมาซึ่งเราก็ต้องนำ ฟิจเจอร์ไปทำการหาค่าสัมประสิทธิ์ที่ใกล้เคียงกับฟิจเจอร์ของเสียงไอโควิด ซึ่งค่าอาร์เรย์ จะนับตั้งแต่เลเบลที่ 1 ไปจนถึง ค่าสัมประสิทธิ์เซปทรัลความถี่เมล ที่เราได้เขียนในบรรทัดโค้ดก่อนการสร้างไฟล์เตรียมชุดข้อมูลเมต้า

6. ตัวอย่างชุดข้อมูลสำหรับการเทรนที่ได้บันทึก



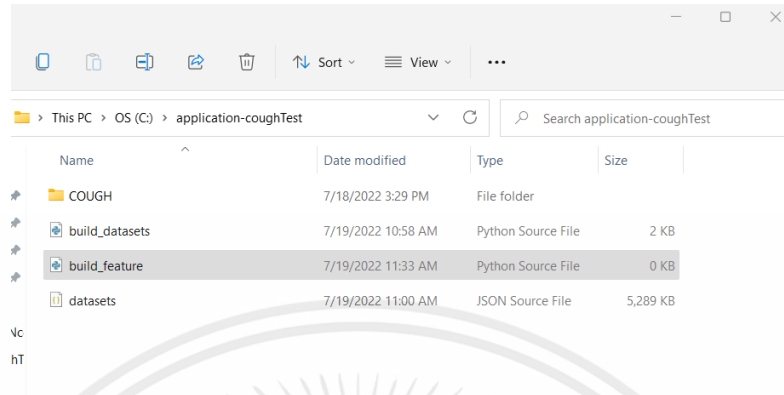
รูปที่ 3.12 ตัวอย่างชุดข้อมูลสำหรับการเทรน

จากรูปที่ 3.12 ตัวอย่างชุดข้อมูลอยู่ในรูปแบบของค่าอาร์เรย์ที่อยู่ในไฟล์เจสัน เพื่อเตรียมหาฟิจเจอร์ของเสียง

3.4 การออกแบบชุดโปรแกรมสำหรับสร้างฟิจเจอร์

การออกแบบชุดโปรแกรมสำหรับสร้างฟิจเจอร์(feature)มีการทำงานของโปรแกรมหลายขั้นตอน ซึ่งแต่ละขั้นตอนมีความสัมพันธ์กัน ในการสร้างฟิจเจอร์(feature)นี้จะนำชุดข้อมูลที่ได้เตรียมไว้มาสร้างฟิจเจอร์(feature)เพื่อหาขอบเขตหรือความเหมือนกันของเสียงไอในแต่ละประเภท โดยมีขั้นตอนดังนี้

3.4.1 สร้างไฟล์สกุลไพทอนของโปรแกรมสร้างฟิจเจอร์



รูปที่ 3.13 ไฟล์โปรแกรมสร้างฟิจเจอร์

จากรูปที่ 3.13 สร้างไฟล์ไพทอนเพื่อสร้างเป็นไฟล์ไพทอนสำหรับการสร้างฟิจเจอร์ที่อยู่ในเสียงไอ

3.4.2 ไลบรารีที่จำเป็นในการสร้างฟิจเจอร์และชื่อไฟล์ที่ต้องการอ่านแล้วบันทึก

```

1 import json
2 import numpy as np
3 import matplotlib.pyplot as plt
4
5 DATA_PATH = "datasets.json"
6 JSON_PATH = "feature.json"

```

รูปที่ 3.14 ไลบรารีและชื่อไฟล์ที่ต้องการอ่านและบันทึก

จากรูปที่ 3.14 แสดงถึงไลบรารีที่จำเป็นต้องเรียกใช้และในบรรทัดที่ 5 คือชื่อไฟล์นามสกุลเจสันที่ต้องการอ่านซึ่งเป็นชุดข้อมูลที่เตรียมไว้ใช้ในกระบวนการเทรนในบรรทัดที่ 6 คือชื่อไฟล์นามสกุลเจสันที่ผ่านกระบวนการเทรนเป็นฟิจเจอร์แล้วต้องการบันทึกเพื่อนำไปใช้ทำนายผล

3.4.3 ชุดคำสั่งหลักในการสร้างฟิจเจอร์

```

259 if __name__ == "__main__":
260     X, y, cough_type_data = load_data(DATA_PATH)
261
262     cnn_data1 = compare_data_high(X)
263     cnn_data2 = compare_data_low(X)
264
265     feature_top_max = cnn_data_max_top(cnn_data1, y)
266     feature_top_min = cnn_data_max_bottom(cnn_data1, y)
267     feature_bottom_max = cnn_data_min_top(cnn_data2, y)
268     feature_bottom_min = cnn_data_min_bottom(cnn_data2, y)
269
270     accuracy = find_accuracy(feature_top_max, feature_top_min, feature_bottom_max, feature_bottom_min, cnn_data1, cnn_data2, y)
271     print("accuracy is {}".format(accuracy))
272
273     save_feature_data(JSON_PATH, feature_top_max, feature_top_min, feature_bottom_max, feature_bottom_min, cnn_data1, cnn_data2,
274     plot_feature(feature_top_max, feature_top_min, feature_bottom_max, feature_bottom_min)

```

รูปที่ 3.15 ชุดคำสั่งหลักในการสร้างฟิจเจอร์

จากรูปที่ 3.15 แสดงถึงชุดคำสั่งหลักที่จะทำงานเชื่อมโยงกันโดยรับค่าจากการประมวลผลในชุดคำสั่งก่อนหน้านี้นำมาประมวลผลต่อ โดยชุดคำสั่งนั้นเมื่อมีการทำงานจะอินเตอร์รัปต์ (interrupt) ไปยังส่วนการทำงานนั้น ๆ เมื่อทำเสร็จจะคืนค่ากลับมาให้คำสั่งในบรรทัดต่อ ๆ ไปได้นำค่าที่ได้ไปใช้ บรรทัดที่ 258 อ่านค่าชุดข้อมูลในไฟล์ ชุดข้อมูล บรรทัดที่ 260-261 ทำการสร้างขอบเขตบนและขอบเขตล่างจากข้อมูลสัมประสิทธิ์เซปสตรัมบนสเกลเมล โดยทราบประเภทสัมประสิทธิ์เซปสตรัมบนสเกลเมลของเสียงแต่ละเสียงได้จากชุดข้อมูล label ที่จะบอกว่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลชุดนั้น ๆ เป็นของเสียงไอโควิดหรือเสียงไอปกติ บรรทัดที่ 263-266 นำขอบเขตบนและขอบเขตล่างที่ได้มาหาขีดจำกัดมากที่สุดและน้อยสุดที่เป็นไปได้ บรรทัดที่ 268-269 หาค่าความแม่นยำของฟิจเจอร์ด้วยชุดข้อมูลเสียงที่นำมาสร้างฟิจเจอร์จากไฟล์ ชุดข้อมูล แล้วแสดง บรรทัดที่ 271 บันทึกฟิจเจอร์ลงในไฟล์สกุลเจสันบรรทัดที่ 273 พล็อตกราฟ (plot graph) ฟิจเจอร์ที่ได้

3.4.4 ชุดคำสั่งอ่านค่าข้อมูลในชุดข้อมูล

```

8  def load_data(data_path):
9      if data_path == DATA_PATH :
10         with open(data_path, "r") as fp:
11             data = json.load(fp)
12             X = np.array(data["mfcc"])
13             y = np.array(data["labels"])
14             cough_type_data = data["labels"]
15             return X, y, cough_type_data

```

รูปที่ 3.16 ชุดคำสั่งอ่านค่าข้อมูลใน ชุดข้อมูล

จากรูปที่ 3.16 เป็นชุดคำสั่งอ่านค่าข้อมูลใน ชุดข้อมูล โดยจะแยกค่าอาร์เรย์ ในเลเบล ต่างๆเพื่อนำข้อมูลฟิจเจอร์มาเปรียบเทียบกับในแกน X และแกน y เพื่อเตรียมหาขอบเขตของฟิจเจอร์

3.4.5 ชุดคำสั่งสร้างขอบเขตของชุดข้อมูล

```

17 def compare_data_high(X):
18     feature = []
19     segments = X.shape[0]
20     for k in range(segments):
21         feature.append(k)
22         mfcc_num = []
23         mfcc_va = 13
24         for i in range(mfcc_va):
25             c = -20000
26             mfcc_compare_in_segment = X.shape[1]
27             for j in range(mfcc_compare_in_segment):
28                 a = X[k][j][i]
29                 if a >= c :
30                     c = a
31             mfcc_num.append(c)
32         feature[k] = mfcc_num
33     feature1 = feature
34     return feature1

```

รูปที่ 3.17 ชุดคำสั่งสร้างขอบเขตบนของชุดข้อมูล

จากรูปที่ 3.17 สร้างชุดคำสั่งสำหรับสร้างขอบเขตบนของชุดข้อมูลที่อยู่ในขอบเขตของ แกน X

```

36 def compare_data_low(X):
37     feature = []
38     segments = X.shape[0]
39     for k in range(segments):
40         feature.append(k)
41         mfcc_num = []
42         mfcc_va = 13
43         for i in range(mfcc_va):
44             c = 20000
45             mfcc_compare_in_segment = 44
46             for j in range(mfcc_compare_in_segment):
47                 a = X[k][j][i]
48                 if a <= c :
49                     c = a
50             mfcc_num.append(c)
51         feature[k] = mfcc_num
52     feature2 = feature
53     return feature2

```

รูปที่ 3.18 ชุดคำสั่งสร้างขอบเขตล่างของชุดข้อมูล

จากรูปที่ 3.18 สร้างชุดคำสั่งสำหรับสร้างขอบเขตล่างของชุดข้อมูลที่อยู่ในขอบเขตของ

แกน X

3.4.6 ชุดคำสั่งสร้างขีดจำกัดจากขอบเขตของชุดข้อมูล

```

55 def cnn_data_max_top(cnn_data1, y):
56     feature_top_max = []
57     y_num_file = y.shape[0]
58     n = -1
59     for d in range(y_num_file):
60         if y[d] != n:
61             n = y[d]
62             feature_top_max.append(n)
63             mfcc_num = []
64             cnn_mfcc_num = 13
65             for j in range(cnn_mfcc_num):
66                 mfcc_num.append(j)
67                 a = -20000
68                 file_num = y.shape[0]
69                 for k in range(file_num):
70                     x = y[k]
71                     b = cnn_data1[k][j]
72                     if x == n:
73                         if a <= b:
74                             a = b
75                 mfcc_num[j] = a
76             feature_top_max[n] = mfcc_num
77     return feature_top_max

```

รูปที่ 3.19 ชุดคำสั่งสร้างขีดจำกัดมากที่สุดขอบเขตบนของชุดข้อมูล

จากรูปที่ 3.19 ชุดคำสั่งสร้างขีดจำกัดมากที่สุดขอบเขตบนของชุดข้อมูลเพื่อกำหนด
ขีดจำกัดมากสุดในชุดข้อมูลขอบเขตบน

```

79 def cnn_data_max_bottom(cnn_data1, y):
80     feature_top_min = []
81     y_num_file = y.shape[0]
82     n = -1
83     for d in range(y_num_file):
84         if y[d] != n:
85             n = y[d]
86             feature_top_min.append(n)
87             mfcc_num = []
88             cnn_mfcc_num = 13
89             for j in range(cnn_mfcc_num):
90                 mfcc_num.append(j)
91                 a = 20000
92                 file_num = y.shape[0]
93                 for k in range(file_num):
94                     x = y[k]
95                     b = cnn_data1[k][j]
96                     if x == n:
97                         if a >= b:
98                             a = b
99                 mfcc_num[j] = a
100             feature_top_min[n] = mfcc_num
101     return feature_top_min

```

รูปที่ 3.20 ชุดคำสั่งสร้างขีดจำกัดน้อยที่สุดขอบเขตบนของชุดข้อมูล

จากรูปที่ 3.20 ชุดคำสั่งสร้างขีดจำกัดน้อยที่สุดขอบเขตบนของชุดข้อมูลเพื่อกำหนด
ขีดจำกัดน้อยสุดในชุดข้อมูลขอบเขตบน

```

103 def cnn_data_min_top(cnn_data2, y):
104     feature_bottom_max = []
105     y_num_file = y.shape[0]
106     n = -1
107     for d in range(y_num_file):
108         if y[d] != n:
109             n = y[d]
110             feature_bottom_max.append(n)
111             mfcc_num = []
112             cnn_mfcc_num = 13
113             for j in range(cnn_mfcc_num):
114                 mfcc_num.append(j)
115                 a = -20000
116                 file_num = y.shape[0]
117                 for k in range(file_num):
118                     x = y[k]
119                     b = cnn_data2[k][j]
120                     if x == n:
121                         if a <= b:
122                             a = b
123                 mfcc_num[j] = a
124             feature_bottom_max[n] = mfcc_num
125     return feature_bottom_max

```

รูปที่ 3.21 ชุดคำสั่งสร้างขีดจำกัดมากที่สุดขอบเขตล่างของชุดข้อมูล

จากรูปที่ 3.21 ชุดคำสั่งสร้างขีดจำกัดมากที่สุดขอบเขตล่างของชุดข้อมูลเพื่อกำหนดขีดจำกัดมากสุดในชุดข้อมูลขอบเขตล่าง

```

127 def cnn_data_min_bottom(cnn_data2, y):
128     feature_bottom_min = []
129     y_num_file = y.shape[0]
130     n = -1
131     for d in range(y_num_file):
132         if y[d] != n:
133             n = y[d]
134             feature_bottom_min.append(n)
135             mfcc_num = []
136             cnn_mfcc_num = 13
137             for j in range(cnn_mfcc_num):
138                 mfcc_num.append(j)
139                 a = 20000
140                 file_num = y.shape[0]
141                 for k in range(file_num):
142                     x = y[k]
143                     b = cnn_data2[k][j]
144                     if x == n:
145                         if a >= b:
146                             a = b
147                     mfcc_num[j] = a
148                 feature_bottom_min[n] = mfcc_num
149     return feature_bottom_min

```

รูปที่ 3.22 ชุดคำสั่งสร้างขีดจำกัดน้อยที่สุดขอบเขตล่างของชุดข้อมูล

จากรูปที่ 3.22 ชุดคำสั่งสร้างขีดจำกัดน้อยที่สุดขอบเขตล่างของชุดข้อมูลเพื่อกำหนดขีดจำกัดน้อยสุดในชุดข้อมูลขอบเขตล่าง

3.4.7 ชุดคำสั่งหาความแม่นยำของฟิจเจอร์

```

151 def find_accury(feature_top_max, feature_top_min, feature_bottom_max, feature_bottom_min, cnn_data1
152     point_for_covid = 0
153     point_for_normal = 0
154     point_for_error = 0
155     point_for_all_covid = 0
156     point_for_all_normal = 0
157
158     for k in range(y.shape[0]):
159         covid = 0
160         normal = 0
161         all_covid = 0
162         all_normal = 0
163
164         if y[k] == 0 :
165             all_covid += 26
166             point_for_all_covid += 26
167         elif y[k] == 1 :
168             all_normal += 26
169             point_for_all_normal += 26
170
171         for j in range(13):
172
173             if cnn_data1[k][j] <= feature_top_max[0][j]:
174                 if cnn_data1[k][j] >= feature_top_min[0][j]:
175                     if cnn_data1[k][j] >= feature_bottom_max[0][j]:
176                         covid += 1
177             if cnn_data2[k][j] <= feature_bottom_max[0][j]:
178                 if cnn_data2[k][j] >= feature_bottom_min[0][j]:
179                     if cnn_data2[k][j] <= feature_top_min[0][j]:
180                         covid += 1
181
182             if cnn_data1[k][j] <= feature_top_max[1][j]:
183                 if cnn_data1[k][j] >= feature_top_min[1][j]:
184                     if cnn_data2[k][j] <= feature_bottom_max[1][j]:
185                         normal += 1
186             if cnn_data2[k][j] <= feature_bottom_max[1][j]:
187                 if cnn_data2[k][j] >= feature_bottom_min[1][j]:
188                     if cnn_data2[k][j] <= feature_top_min[1][j]:
189                         normal += 1
190
191         if covid == 26 or normal == 26:
192             if covid == normal:
193                 point_for_error += covid
194             elif covid > normal:
195                 point_for_covid += covid
196             elif covid < normal:
197                 point_for_normal += normal
198
199         accury_point_for_covid = (point_for_covid / point_for_all_covid)
200         print("accury of covid cough is {}".format(accurry_point_for_covid))
201         accury_point_for_normal = (point_for_normal / point_for_all_normal)
202         print("accury of normal cough is {}".format(accurry_point_for_normal))
203         total_point_for_error = (point_for_error / (point_for_all_covid + point_for_all_normal))
204         print("accury for unknown cough is {}".format(total_point_for_error))
205         accury_train_cough = ((accurry_point_for_covid + accury_point_for_normal) / 2)
206
207     return accury_train_cough

```

รูปที่ 3.23 ชุดคำสั่งหาความแม่นยำของฟิจเจอร์

จากรูปที่ 3.23 แสดงชุดคำสั่งหาความแม่นยำของฟิจเจอร์ จากการนำข้อมูลเสียงไอแต่ละเสียงไอนำมาเป็นฟิจเจอร์มาเปรียบเทียบกับฟิจเจอร์ที่ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.4.8 ชุดคำสั่งบันทึกฟิจเจอร์

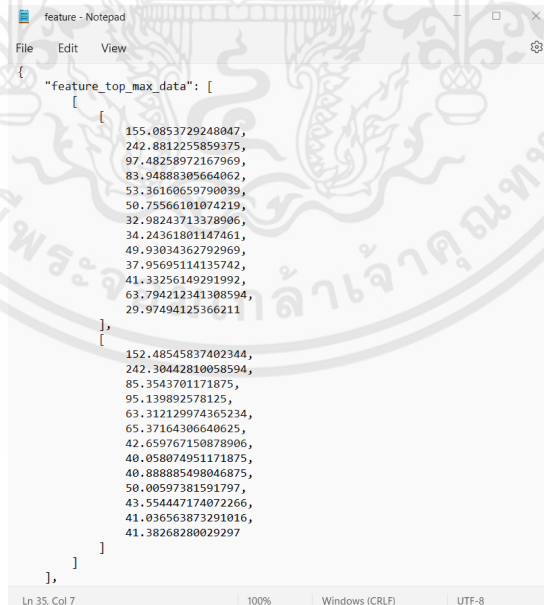
```

214 def save_feature_data(json_path, feature_top_max, feature_top_min, feature_bottom_max,
215 feature_bottom_min, cnn_data1, cnn_data2, cough_type_data):
216     data = {
217         "feature_top_max_data": []
218         , "feature_top_min_data": []
219         , "feature_bottom_max_data": []
220         , "feature_bottom_min_data": []
221         , "cnn_top_data": []
222         , "cnn_bottom_data": []
223         , "cough_type_data": []
224     }
225     data["feature_top_max_data"].append(feature_top_max)
226     data["feature_top_min_data"].append(feature_top_min)
227     data["feature_bottom_max_data"].append(feature_bottom_max)
228     data["feature_bottom_min_data"].append(feature_bottom_min)
229
230     data["cnn_top_data"].append(cnn_data1)
231     data["cnn_bottom_data"].append(cnn_data2)
232     data["cough_type_data"].append(cough_type_data)
233
234     # save cnn data to json file
235     with open(json_path, "w") as fp:
236         json.dump(data, fp, indent=4)

```

รูปที่ 3.24 ชุดคำสั่งบันทึกฟิจเจอร์เป็นสกุลไฟล์เจสัน

จากรูปที่ 3.24 ชุดคำสั่งบันทึกฟิจเจอร์ เพื่อบันทึกฟิจเจอร์ที่ได้จากการแยกฟิจเจอร์อยู่ในไฟล์เจสัน



```

{
  "feature_top_max_data": [
    155.0853729248047,
    242.8812255859375,
    97.48258972167969,
    83.94888305664062,
    53.36160659790039,
    50.75566101074219,
    32.98243713378906,
    34.24361801147461,
    49.93034362792969,
    37.95695114135742,
    41.33256149291992,
    63.794212341308594,
    29.97494125366211,
    ],
  "feature_top_min_data": [
    152.48545837402344,
    242.30442810058594,
    85.3543701171875,
    95.139892578125,
    63.312129974365234,
    65.37164306640625,
    42.659767150878906,
    40.058074951171875,
    40.888885498046875,
    50.00597381591797,
    43.554447174072266,
    41.036563873291016,
    41.38268280029297,
    ]
}

```

รูปที่ 3.25 ตัวอย่างฟิจเจอร์สกุลไฟล์เจสัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากรูปที่ 3.25 ตัวอย่างฟังก์ชันสกริปต์ไฟล์เจสัน ที่บันทึกฟังก์ชัน

3.4.9 ชุดคำสั่งพล็อตกราฟฟังก์ชัน

```

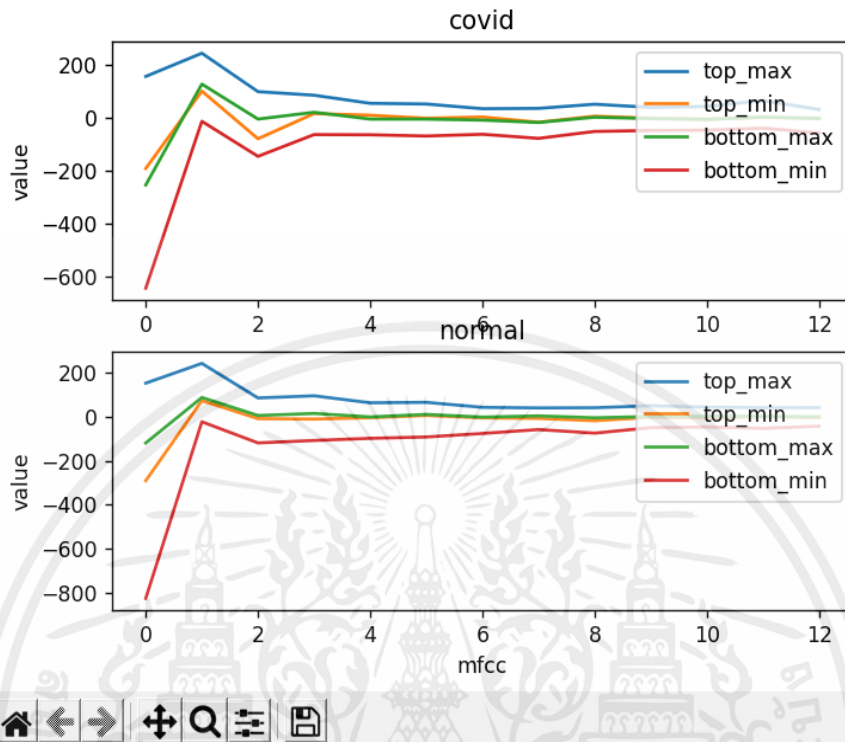
232 v def plot_feature(feature_top_max, feature_top_min, feature_bottom_max, feature_bottom_min):
233
234     fig, axs = plt.subplots(2)
235
236     # create left subplot
237     axs[0].plot(feature_top_max[0], label="top_max")
238     axs[0].plot(feature_top_min[0], label="top_min")
239     axs[0].plot(feature_bottom_max[0], label="bottom_max")
240     axs[0].plot(feature_bottom_min[0], label="bottom_min")
241
242     axs[0].set_ylabel("value")
243     axs[0].legend(loc="upper right")
244     axs[0].set_title("covid")
245
246     # create right subplot
247     axs[1].plot(feature_top_max[1], label="top_max")
248     axs[1].plot(feature_top_min[1], label="top_min")
249     axs[1].plot(feature_bottom_max[1], label="bottom_max")
250     axs[1].plot(feature_bottom_min[1], label="bottom_min")
251
252     axs[1].set_ylabel("value")
253     axs[1].set_xlabel("mfcc num")
254     axs[1].legend(loc="upper right")
255     axs[1].set_title("normal")
256
257     plt.show()

```

รูปที่ 3.26 ชุดคำสั่งพล็อตกราฟ ฟังก์ชัน

จากรูปที่ 3.26 ชุดคำสั่งพล็อตกราฟ ฟังก์ชัน เพื่อพล็อตกราฟของขอบเขตบนและขอบเขตล่าง ซึ่งจะได้ค่ากราฟออกมาในช่องของตัวแปร `axs[1]` แล้วนำไปเก็บค่าในแต่ละเลเบลจากนั้นก็นำมาแสดงค่าเพื่อให้นักพัฒนาเปรียบเทียบและเก็บผลการทดลอง

Figure 1



รูปที่ 3.27 กราฟฟิจเจอร์ที่ได้จากการเทรน

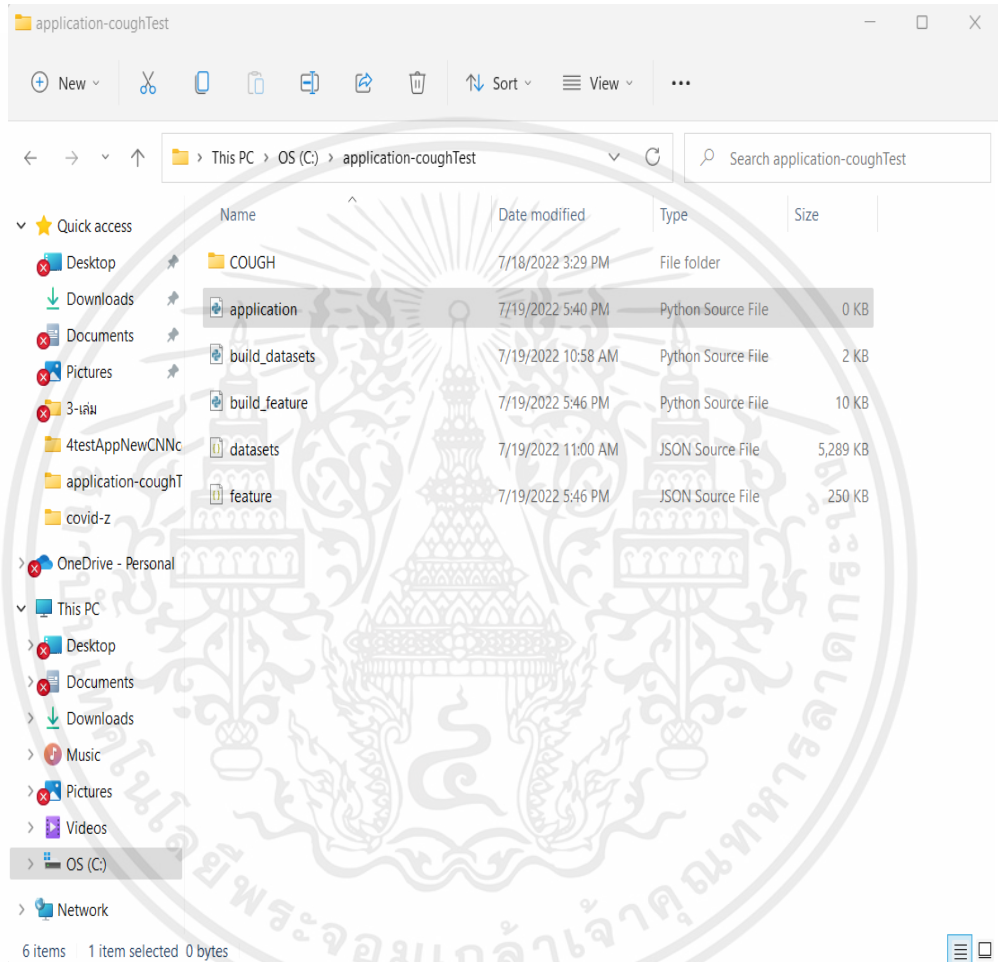
จากรูปที่ 3.27 กราฟฟิจเจอร์ที่ได้จากการเทรน สร้างด้วยการใช้ไลบรารีพายพลอต โดยค่าตัวแปร value บนตารางคือ ค่าสัมประสิทธิ์เซปสตรัมบนสนสเกลเมล ค่าตัวแปร mfcc บนตารางหมายถึง ลำดับทั้ง 13 ของค่าสัมประสิทธิ์เซปสตรัมบนสนสเกลเมล กราฟบนเป็นฟิจเจอร์ของเสียงโควิดและกราฟล่างเป็นฟิจเจอร์ของเสียงปกติ ซึ่งมีขอบเขตบนและขอบเขตล่าง แต่ละขอบเขตมีขีดจำกัดสูงสุดและต่ำสุด

3.5 การออกแบบแอปพลิเคชันและการทำงาน

การออกแบบแอปพลิเคชันมีผลอย่างมากต่อความสะดวกในการใช้งานรวมถึงการประมวลผลที่ใช้เวลาน้อยและให้มีประสิทธิภาพมากที่สุด ทำงานโดยการบันทึกเสียงแล้วนำเสียงที่ได้ไปประมวลผลกับฟิจเจอร์ที่เตรียมไว้ โดยมีขั้นตอนการออกแบบดังนี้

3.5.1 สร้างไฟล์สกุลไพทอนของแอปพลิเคชัน

สร้างไฟล์แอปพลิเคชันเป็นไฟล์ไพทอนเพื่อนำมาสร้างแอปพลิเคชันที่รันบนระบบปฏิบัติการวินโดวส์ เพื่อนำไปสร้างแอปพลิเคชันโดยใช้ GUI ของกวีแล้วจากนั้นก็ทำการออกแบบแอปพลิเคชัน ดังรูปที่ 3.28



รูปที่ 3.28 ไฟล์แอปพลิเคชัน

3.5.2 โลบาร์รี่ที่จำเป็นในการสร้างแอปพลิเคชันและค่าที่ต้องกำหนด

แสดงโลบาร์รี่ที่ต้องเรียกใช้ ตำแหน่งไฟล์ที่ต้องเรียกอ่าน เวลาการบันทึก 3 วินาที และการแบ่งไฟล์เสียงที่บันทึก ซึ่งรวมไปถึงโลบาร์รี่ตั้งแต่การสร้างชุดข้อมูล เช่น ลิบโซ่า นัมพาย เจสันตลอดจนถึงโลบาร์รี่ในการสร้างแอปพลิเคชัน ดังรูปที่ 3.29

```

1  import json
2  import kivy
3  from kivy.app import App
4  from kivy.uix.boxlayout import BoxLayout
5  import librosa
6  import numpy as np
7  import math
8  import matplotlib.pyplot as plt
9  import sounddevice
10 import soundfile as sf
11
12 kivy.require('2.1.0')
13
14 DATA_FEATURE = "feature.json"
15 UNKNOWN_COUGH_PATH = "unknow_cough.wav"
16 duration = 3
17 num_segments = (duration * 2) - 1

```

รูปที่ 3.29 ไบรารีที่จำเป็นและค่าที่ต้องกำหนด

3.5.3 คำสั่งหลักในการทำงานของแอปพลิเคชัน

```

276 if __name__ == "__main__":
277
278     W, X, Y, Z, top_data, bottom_data, type_data = load_data(DATA_FEATURE)
279
280     AHADC().run()

```

รูปที่ 3.30 คำสั่งหลักในการทำงานของแอปพลิเคชันส่วนที่ 1

จากรูปที่ 3.30 แสดงคำสั่งหลักในการทำงานของแอปพลิเคชันส่วนที่ 1 มีการเรียกอ่านค่าจากพีจเจอร์ที่ได้เตรียมไว้และสั่งการทำงานหน้าต่างแอปพลิเคชันให้มีการใช้งานอยู่ตลอด

```

19 class MyRoot(BoxLayout):
20
21     def __init__(self):
22         super(MyRoot, self).__init__()
23
24     def start_record(self):
25         self.state_label.text = str("recording")
26
27     def start_predict_on_app(self):
28
29         recording_cough()
30
31         feature_unknown_cough_high, feature_unknown_cough_low = load_unknown_cough(UNKNOWN_COUGH_PATH)
32
33         accurry_covid, accurry_normal, error = classifier_unknown_cough(W, X, Y ,Z ,top_data ,botton_data
34         ,type_data, feature_unknown_cough_high, feature_unknown_cough_low)
35
36         predict_cough = predict(accurry_covid, accurry_normal, error)
37         if predict_cough == 0:
38             self.state_label.text = str("You are infected covid-19")
39         elif predict_cough == 1:
40             self.state_label.text = str("You are normal")
41         elif predict_cough == 5:
42             self.state_label.text = str("this is not cough")
43
44         #plot_predict(W, X, Y, Z, feature_unknown_cough_high, feature_unknown_cough_low)
45
46 class AHADC(App):
47
48     def build(self):
49         return MyRoot()

```

รูปที่ 3.31 คำสั่งหลักในการทำงานของแอปพลิเคชันส่วนที่ 2

จากรูปที่ 3.31 คำสั่งหลักในการทำงานของแอปพลิเคชันส่วนที่ 2 บรรทัดที่ 46-49 แสดงหน้าต่างที่ได้อ่านจาก class MyRoot บรรทัดที่ 21-22 อ่านหน้าต่างเดิมเพื่อนำไปวนลูปมาแสดงซ้ำ บรรทัดที่ 24-25 หากมีการกดปุ่มบันทึกเสียงให้แสดงคำว่า “recording” บรรทัดที่ 27-44 เป็นส่วนการทำงานหลังจากมีการกดปุ่มบันทึกเสียง โดยบรรทัดที่ 29 เริ่มการบันทึกเสียงผ่านไมโครโฟน (micro phone)บนคอมพิวเตอร์ แล้วบันทึกเป็นนามสกุล wav ไฟล์ บรรทัดที่ 31 อ่านค่าเสียงที่บันทึกได้แล้ว สร้างขอบเขตบนและขอบเขตล่าง บรรทัดที่ 33-34 ประมวลผลความเข้ากันของพีจเจอร์กับขอบเขตบน และขอบเขตล่างเสียงที่บันทึก บรรทัดที่ 36 ทำนายผลเสียงที่รับเข้า บรรทัดที่ 37-42 แสดงข้อความการทำนายผลบนหน้าต่างแอปพลิเคชัน บรรทัดที่ 44 แสดงกราฟการประมวลผล

3.5.4 ออกแบบสร้างหน้าต่างแอปพลิเคชัน

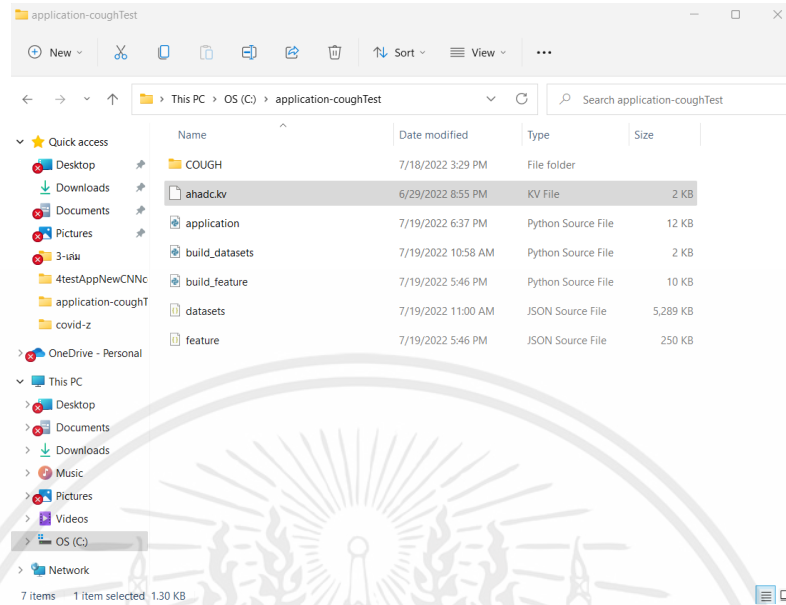
```

1  <MyRoot>:
2  canvas.before:
3      Color:
4          rgba: 1, 1, 1, 1
5      Rectangle:
6          pos: self.pos
7          size: self.size
8  state_label: state_label
9  BoxLayout:
10     orientation: 'vertical'
11     Button:
12         background_color: 97/255, 208/255, 1, 1
13         color: 1, 1, 1, 1
14         font_size: 48
15         text: "Application Helps Assess Disease From Coughing"
16     Label:
17     Label:
18     Label:
19         color: 0.92, 0.45, 0, 1
20         font_size: 48
21         outline_color: 0,0,0
22         outline_width: 2
23         text: "press Start button then cough in 3 seconds"
24     Label:
25     Label:
26     Label:
27         color: 0.25, 1, 0.98, 1
28         font_size: 72
29         id: state_label
30         text: "ready to record"
31     Label:
32     Label:
33     BoxLayout:
34         orientation: 'horizontal'
35         Label:
36         Button:
37             background_color: 0, 126/255, 181/255, 1
38             color: 64/255, 1, 35/255, 1
39             font_size: 32
40             on_release: root.start_predict_on_app()
41             on_press: root.start_record()
42             size: 100, 50
43             text: "Start"
44         Label:
45     Label :
46     Label :

```

รูปที่ 3.32 คำสั่งออกแบบหน้าต่างแอปพลิเคชัน

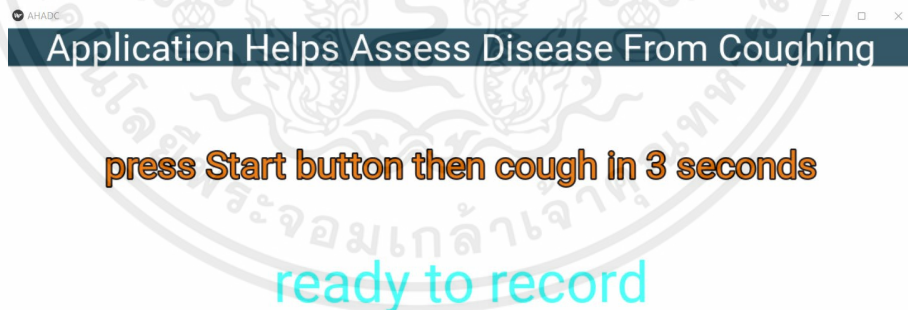
จากรูปที่ 3.32 แสดงคำสั่งออกแบบหน้าต่างแอปพลิเคชัน เมื่อเขียนโปรแกรมเสร็จให้บันทึกลงในโฟลเดอร์เดียวกับแอปพลิเคชันโดยตั้งชื่อว่า “ahadc” เป็นไฟล์สกุล KV



รูปที่ 3.33 ไฟล์ “ahadc.kv” ที่ได้หลังจากการบันทึกเสียงโพลเดอร์เดียวกับแอปพลิเคชัน

จากรูปที่ 3.33 ไฟล์ “ahadc.kv” ที่ได้หลังจากการบันทึกเสียงโพลเดอร์เดียวกับ

แอปพลิเคชัน



รูปที่ 3.34 หน้าต่างแอปพลิเคชันหลังการออกแบบ

จากรูปที่ 3.34 หน้าต่างแอปพลิเคชันหลังการออกแบบ โดยจะมีปุ่ม Start สำหรับ
อัดเสียง

3.5.5 คำสั่งอ่านข้อมูลฟิจเจอร์ที่ได้เตรียมไว้

```

59 def load_data(data_path):
60
61     if data_path == DATA_FEATURE :
62         with open(data_path, "r") as fp:
63             data = json.load(fp)
64
65             W = np.array(data["feature_top_max_data"])
66             X = np.array(data["feature_top_min_data"])
67             Y = np.array(data["feature_bottom_max_data"])
68             Z = np.array(data["feature_bottom_min_data"])
69
70             top_data = np.array(data["cnn_top_data"])
71             bottom_data = np.array(data["cnn_bottom_data"])
72             type_data = np.array(data["cough_type_data"])
73
74             return W, X, Y ,Z ,top_data ,bottom_data ,type_data

```

รูปที่ 3.35 คำสั่งอ่านข้อมูลฟิจเจอร์ที่ได้เตรียมไว้

จากรูปที่ 3.35 คำสั่งอ่านข้อมูลฟิจเจอร์ที่ได้เตรียมไว้ เพื่อนำมาอ่านฟิจเจอร์ที่ได้จาก
เสียงไอของผู้ใช้งาน

3.5.6 คำสั่งบันทึกเสียง

```

51 def recording_cough():
52     fps = 22050
53     print("Recording...")
54     recording_unknow_cough = sounddevice.rec(int(duration*fps), samplerate=fps, channels=2)
55     sounddevice.wait()
56     print("Done!")
57     sf.write("unknow_cough.wav",recording_unknow_cough,fps)

```

รูปที่ 3.36 คำสั่งบันทึกเสียง

จากรูปที่ 3.36 แสดงคำสั่งการบันทึกเสียงผ่านไมโครโฟนบนคอมพิวเตอร์ แล้ว
บันทึกเสียงที่ได้เป็นไฟล์นามสกุล wav โดยมีชื่อไฟล์ว่า “unknow_cough.wav”

3.5.7 คำสั่งอ่านไฟล์เสียงที่ได้บันทึกเป็นไฟล์สกุล wav

```

76 def load_unknown_cough(data_path, num_mfcc=13, n_fft=2048, hop_length=512):
77     data = {
78         "mfcc": []
79     }
80     SAMPLE_RATE = 22050
81     num_mfcc_vectors_per_segment = math.ceil(SAMPLE_RATE / hop_length)
82     signal, sample_rate = librosa.load(data_path, sr=SAMPLE_RATE)
83     for d in range(num_segments):
84         samples_per_segment = SAMPLE_RATE / 2
85         start = int(samples_per_segment * d)
86         finish = start + SAMPLE_RATE
87         if finish == start + SAMPLE_RATE:
88             mfcc = librosa.feature.mfcc(signal[start:finish], sample_rate, n_mfcc=num_mfcc, n_fft=n_fft, hop_length=hop_length)
89             mfcc = mfcc.T
90             if len(mfcc) == num_mfcc_vectors_per_segment:
91                 data["mfcc"].append(mfcc.tolist())
92     data = np.array(data["mfcc"])
93     feature_unknown_cough_high = compare_data_high(data)
94     feature_unknown_cough_high = np.array(feature_unknown_cough_high)
95     feature_unknown_cough_low = compare_data_low(data)
96     feature_unknown_cough_low = np.array(feature_unknown_cough_low)
97     return feature_unknown_cough_high, feature_unknown_cough_low

```

รูปที่ 3.37 คำสั่งอ่านไฟล์เสียงที่ได้บันทึกเป็นไฟล์สกุล .wav

จากรูปที่ 3.37 แสดงคำสั่งอ่านไฟล์เสียงที่ได้บันทึกเป็นไฟล์สกุล wav แล้วแยกเป็นส่วน ๆ และยังได้ทำการหาขอบเขตของไฟล์เสียงที่รับมาในแต่ละส่วนเพื่อจะนำเข้าสู่กระบวนการประมวลผล ก่อนทำนายผล

```

99 def compare_data_high(X):
100     feature = []
101     segments = X.shape[0]
102     for k in range(segments):
103         feature.append(k)
104         mfcc_num = []
105         mfcc_va = X.shape[2]
106         for i in range(mfcc_va):
107             c = -20000
108             mfcc_compare_in_segment = X.shape[1]
109             for j in range(mfcc_compare_in_segment):
110                 a = X[k][j][i]
111                 if a >= c :
112                     c = a
113             mfcc_num.append(c)
114         feature[k] = (mfcc_num)
115     feature_unknown_cough_high = feature
116     return feature_unknown_cough_high

```

รูปที่ 3.38 คำสั่งหาขอบเขตบนของไฟล์เสียงที่ได้บันทึกเป็นไฟล์สกุล wav ในแต่ละส่วน

จากรูปที่ 3.38 คำสั่งหาขอบเขตบนของไฟล์เสียงที่ได้บันทึกเป็นไฟล์สกุล .wav ในแต่ละส่วน

```

118 v def compare_data_low(X):
119     feature = []
120     segments = X.shape[0]
121 v   for k in range(segments):
122       feature.append(k)
123       mfcc_num = []
124       mfcc_va = X.shape[2]
125 v     for i in range(mfcc_va):
126       c = 20000
127       mfcc_compare_in_segment = X.shape[1]
128 v     for j in range(mfcc_compare_in_segment):
129       a = X[k][j][i]
130 v     if a <= c :
131       c = a
132       mfcc_num.append(c)
133     feature[k] = (mfcc_num)
134     feature_unknow_cough_low = feature
135     return feature_unknow_cough_low

```

รูปที่ 3.39 คำสั่งหาขอบเขตล่างของไฟล์เสียงที่ได้บันทึกเป็นไฟล์สกุล wav ในแต่ละส่วน

จากรูปที่ 3.39 คำสั่งหาขอบเขตล่างของไฟล์เสียงที่ได้บันทึกเป็นไฟล์สกุล wav ในแต่ละส่วน

3.5.8 คำสั่งประมวลผลเปรียบเทียบเสียงที่รับเข้ากับฟิจเจอร์

```

137 def classifier_unknown_cough(W, X, Y, Z, top_data, bottom_data, type_data, feature_unknown_cough_high, feature_un
138     accurcy_covid = 0
139     accurcy_normal = 0
140     cough_point = 0
141     error = 100
142     data = []
143     for d in range(num_segments):
144         data.append(d)
145         point = []
146         for k in range(W.shape[1]):
147             if k == 0:
148                 point.append(k)
149                 point.append(k+1)
150                 point[k+1] = 0
151                 for j in range(13):
152                     if k == 0:
153                         if feature_unknown_cough_high[d][j] <= W[0][k][j]:
154                             if feature_unknown_cough_high[d][j] >= X[0][k][j]:
155                                 if feature_unknown_cough_high[d][j] >= Y[0][k][j]:
156                                     point[k+1] += 1
157                         if feature_unknown_cough_low[d][j] <= X[0][k][j]:
158                             if feature_unknown_cough_low[d][j] <= Y[0][k][j]:
159                                 if feature_unknown_cough_low[d][j] >= Z[0][k][j]:
160                                     point[k+1] += 1
161                         if feature_unknown_cough_high[d][j] > W[0][k][j]:
162                             if feature_unknown_cough_high[d][j] > W[0][k+1][j]:
163                                 point[k] += 1
164                         if feature_unknown_cough_high[d][j] < X[0][k][j]:
165                             if feature_unknown_cough_high[d][j] < X[0][k+1][j]:
166                                 point[k] += 1
167                         if feature_unknown_cough_low[d][j] > Y[0][k][j]:
168                             if feature_unknown_cough_low[d][j] > Y[0][k+1][j]:
169                                 point[k] += 1
170                         if feature_unknown_cough_low[d][j] < Z[0][k][j]:
171                             if feature_unknown_cough_low[d][j] < Z[0][k+1][j]:
172                                 point[k] += 1
173                     if k == 1:
174                         if feature_unknown_cough_high[d][j] <= W[0][k][j]:
175                             if feature_unknown_cough_high[d][j] >= X[0][k][j]:
176                                 if feature_unknown_cough_high[d][j] >= Y[0][k][j]:
177                                     point[k+1] += 1
178                         if feature_unknown_cough_low[d][j] <= X[0][k][j]:
179                             if feature_unknown_cough_low[d][j] <= Y[0][k][j]:
180                                 if feature_unknown_cough_low[d][j] >= Z[0][k][j]:
181                                     point[k+1] += 1
182                 data[d] = (point)
183                 print(data)
184             for a in range(num_segments):
185                 if data[a][0] == 0:
186                     cough_point += 1
187             else :
188                 if data[a][0] < error:
189                     error = data[a][0]

```

รูปที่ 3.40 คำสั่งประมวลผลเปรียบเทียบเสียงที่รับเข้ากับฟิจเจอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากรูปที่ 3.40 คำสั่งประมวลผลเปรียบเทียบเสียงที่รับเข้ากับพีจเจอร์ โดยชุดคำสั่งนี้จะเปรียบเทียบพีเจอร์ของเสียงไอผู้ใช้งานกับเสียงไอที่อยู่ในชุดข้อมูลแล้วประมวลผลออกมา

3.5.9 คำสั่งทำนายผล

```

224 def predict(accurry_covid, accurry_normal, error):
225     x = error
226     a = accurry_covid * 100
227     b = accurry_normal * 100
228     print(error)
229     if x == 0 :
230         if b >= 100:
231             print("you are normal")
232             print("accurry to normal cough {} percent".format(b))
233             print("accurry to covid cough {} percent".format(a))
234             return 1
235         elif a >= 100 :
236             print("you are infected")
237             print("accurry to covid cough {} percent".format(a))
238             print("accurry to normal cough {} percent".format(b))
239             return 0
240         else :
241             print("this is not cough")
242             print("accurry to covid cough {} percent".format(a))
243             print("accurry to normal cough {} percent".format(b))
244             print("error point = {} ".format(x))
245             return 5
246     else :
247         print("this is not cough")
248         print("accurry to covid cough {} percent".format(a))
249         print("accurry to normal cough {} percent".format(b))
250         print("error point = {} ".format(x))
251         return 5

```

รูปที่ 3.41 คำสั่งทำนายผล

จากรูปที่ 3.41 คำสั่งทำนายผล โดยชุดคำสั่งชุดนี้จะเป็นจะทำนายผลหลังจากการเปรียบเทียบเสียงไอของผู้ใช้งานกับเสียงไอที่อยู่ในชุดข้อมูล โดยเปรียบเทียบจากค่าพีจเจอร์ที่ถอดมาได้กับเสียงไปที่อินพุตเข้าไปใหม่ผ่านไมโครโฟน

3.5.10 คำสั่งการพล็อตกราฟเปรียบเทียบเสียงกับฟิจเจอร์

```

253 def plot_predict(W, X, Y, Z, feature_unknown_cough_high, feature_unknown_cough_low):
254     fig, axs = plt.subplots(2)
255     axs[0].plot(W[0][0], label="Covid-top_max")
256     axs[0].plot(X[0][0], label="Covid-top_min")
257     axs[0].plot(Y[0][0], label="Covid-bottom_max")
258     axs[0].plot(Z[0][0], label="Covid-bottom_min")
259     axs[0].plot(feature_unknown_cough_high[0], label="unknown-cough-HIGH")
260     axs[0].plot(feature_unknown_cough_low[0], label="unknown-cough-LOW")
261     axs[0].set_ylabel("MFCC value")
262     axs[0].legend(loc="lower right")
263     axs[0].set_title("covid test")
264     axs[1].plot(W[0][1], label="Normal-top_max")
265     axs[1].plot(X[0][1], label="Normal-top_min")
266     axs[1].plot(Y[0][1], label="Normal-bottom_max")
267     axs[1].plot(Z[0][1], label="Normal-bottom_min")
268     axs[1].plot(feature_unknown_cough_high[0], label="unknown-cough-HIGH")
269     axs[1].plot(feature_unknown_cough_low[0], label="unknown-cough-LOW")
270     axs[1].set_ylabel("MFCC value")
271     axs[1].set_xlabel("Mel-frequency cepstral coefficients : MFCCs")
272     axs[1].legend(loc="lower right")
273     axs[1].set_title("normal test")
274     plt.show()

```

รูปที่ 3.42 คำสั่งการพล็อตกราฟเปรียบเทียบเสียงกับฟิจเจอร์

จากรูปที่ 3.42 คำสั่งการพล็อตกราฟเปรียบเทียบเสียงกับฟิจเจอร์ เพื่อแสดงให้เห็นขอบเขตของการตรวจสอบข้อมูลที้อนพุตเข้าไป ว่าอยู่ในช่วงไหนสำหรับการทำนายผล

3.5.11 เริ่มการทำงานแอปพลิเคชัน

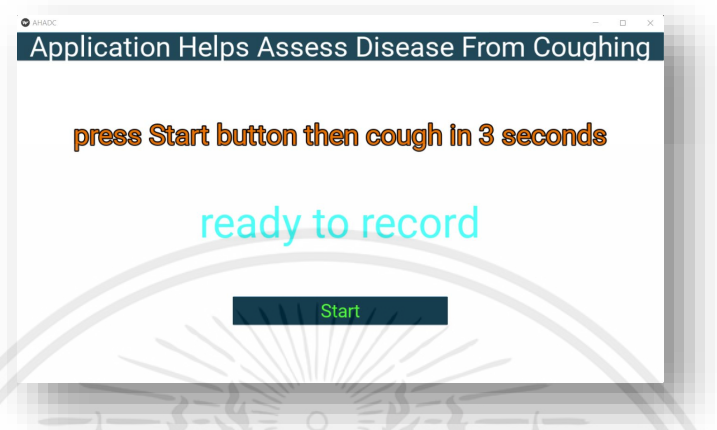
```

PROBLEMS OUTPUT TERMINAL JUPYTER DEBUG CONSOLE
PS C:\application-coughTest> python -u "c:\application-coughTest\application.py"
[INFO] [Logger] Record log in c:\Users\ASUS\Kivy\logs\kivy-22-07-19_3.txt
[INFO] [deps] Successfully imported "kivy_deps.angle" 0.3.2
[INFO] [deps] Successfully imported "kivy_deps.glew" 0.3.1
[INFO] [deps] Successfully imported "kivy_deps.sdl2" 0.4.5
[INFO] [Kivy] v2.1.0
[INFO] [Kivy] Installed at "C:\Users\ASUS\AppData\Roaming\Python\Python39\site-packages\kivy\_init_.py"
[INFO] [Python] v3.9.6 (tags/v3.9.6:db3ff76, Jun 28 2021, 15:26:21) [MSC v.1929 64 bit (AMD64)]
[INFO] [Python] Interpreter at "C:\Program Files\Python39\python.exe"
[INFO] [Logger] Purge log fired. Processing...
[INFO] [Logger] Purge finished!
[INFO] [Factory] 189 symbols loaded
[INFO] [Image] Providers: img_tex, img_dds, img_sdl2, img_pil (img_ffpyplayer ignored)
[INFO] [Image] Provider: sdl2
[INFO] [GL] Using the "OpenGL" graphics system
[INFO] [GL] GLSL initialization succeeded
[INFO] [GL] Backend used <glew>
[INFO] [GL] OpenGL version <b'4.6.0 NVIDIA 512.78'>
[INFO] [GL] OpenGL vendor <b'NVIDIA Corporation'>
[INFO] [GL] OpenGL renderer <b'NVIDIA GeForce GTX 1650/PCIe/SSE2'>
[INFO] [GL] OpenGL parsed version: 4.6
[INFO] [GL] Shading version <b'4.60 NVIDIA'>
[INFO] [GL] Texture max size <32768>
[INFO] [GL] Texture max units <32>
[INFO] [GL] auto add sdl2 input provider
[INFO] [GL] virtual keyboard not allowed, single mode, not docked
[INFO] [GL] Provider: sdl2
[INFO] [Base] Start application main loop
[INFO] [GL] NPOT texture support is available

```

รูปที่ 3.43 หน้าต่างเทอร์มินอลที่ทำงานเมื่อรันแอปพลิเคชัน

จากรูปที่ 3.43 หน้าต่างเทอร์มินอลที่ทำงานเมื่อรันแอปพลิเคชัน เพื่อตรวจสอบบัคในการสร้างและรันแอปพลิเคชัน



รูปที่ 3.44 หน้าต่างแอปพลิเคชันเมื่อรันแอปพลิเคชัน

จากรูปที่ 3.44 หน้าต่างแอปพลิเคชันเมื่อรันแอปพลิเคชัน แสดงปุ่ม Start สำหรับการอัดเสียงของผู้ใช้งาน ซึ่งรันอยู่ในระบบปฏิบัติการวินโดวส์

```

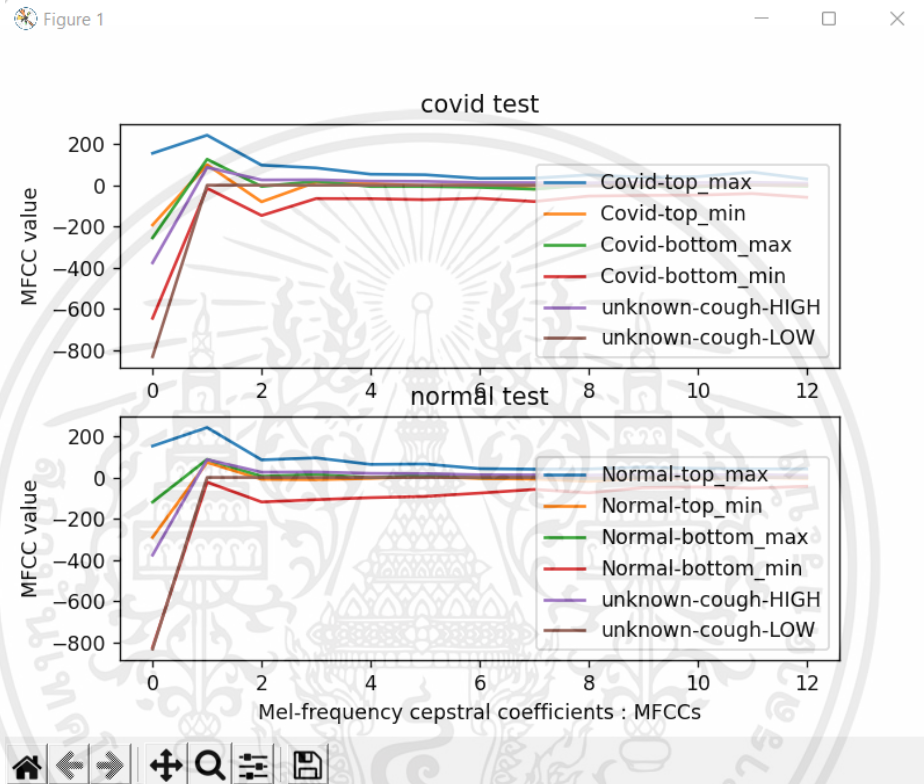
PROBLEMS OUTPUT TERMINAL JUPYTER DEBUG CONSOLE
[INFO] [GL] Backend used (glsl)
[INFO] [GL] OpenGL version 4.6.0 NVIDIA 512.78
[INFO] [GL] OpenGL vendor <NVIDIA Corporation>
[INFO] [GL] OpenGL renderer <NVIDIA GeForce GTX 1650/PCIe/SSE2>
[INFO] [GL] OpenGL parsed version: 4.6
[INFO] [GL] Shading version 4.60 NVIDIA
[INFO] [GL] Texture max size 32768
[INFO] [GL] Texture max units 32
[INFO] [GL] auto add sdl2 input provider
[INFO] [window] virtual keyboard not allowed, single mode, not docked
[INFO] [Text] Provider: sdl2
[INFO] [Base] Start application main loop
[INFO] [GL] NPOT texture support is available
Recording...
Done!
c:\application-coughTest\application.py:88: FutureWarning: Pass y=[ 0.0000000e+00 -1.5258789e-05 0.0000000e+00
-3.3920288e-02 -3.1845099e-02], sr=22050 as keyword args. From version 0.10 passing these as positional argument
mfcc = librosa.feature.mfcc(signal[start:finish], sample_rate, n_mfcc=num_mfcc, n_fft=n_fft, hop_length=hop_le
c:\application-coughTest\application.py:88: FutureWarning: Pass y=[-0.00095354 -0.00195286 0.00075294 ... 0.041
0.03649902], sr=22050 as keyword args. From version 0.10 passing these as positional arguments will result in a
mfcc = librosa.feature.mfcc(signal[start:finish], sample_rate, n_mfcc=num_mfcc, n_fft=n_fft, hop_length=hop_le
c:\application-coughTest\application.py:88: FutureWarning: Pass y=[-0.03258122 -0.0337677 -0.03323364 ... 0.001
0.00646973], sr=22050 as keyword args. From version 0.10 passing these as positional arguments will result in a
mfcc = librosa.feature.mfcc(signal[start:finish], sample_rate, n_mfcc=num_mfcc, n_fft=n_fft, hop_length=hop_le
c:\application-coughTest\application.py:88: FutureWarning: Pass y=[0.03778876 0.04165649 0.03843689 ... 0.0407867
n 0.10 passing these as positional arguments will result in an error
mfcc = librosa.feature.mfcc(signal[start:finish], sample_rate, n_mfcc=num_mfcc, n_fft=n_fft, hop_length=hop_le
c:\application-coughTest\application.py:88: FutureWarning: Pass y=[0.01945227 0.00579834 0.00588989 ... 0.0220947
n 0.10 passing these as positional arguments will result in an error
mfcc = librosa.feature.mfcc(signal[start:finish], sample_rate, n_mfcc=num_mfcc, n_fft=n_fft, hop_length=hop_le
[[4, 15, 15], [8, 13, 14], [8, 14, 14], [8, 14, 14], [8, 13, 13]]
4
this is not cough
accuracy to covid cough 0 percent
accuracy to normal cough 0 percent

```

รูปที่ 3.45 หน้าต่างเทอร์มินอลเมื่อบันทึกเสียงแล้วผ่านการประมวลผลและทำนายผล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

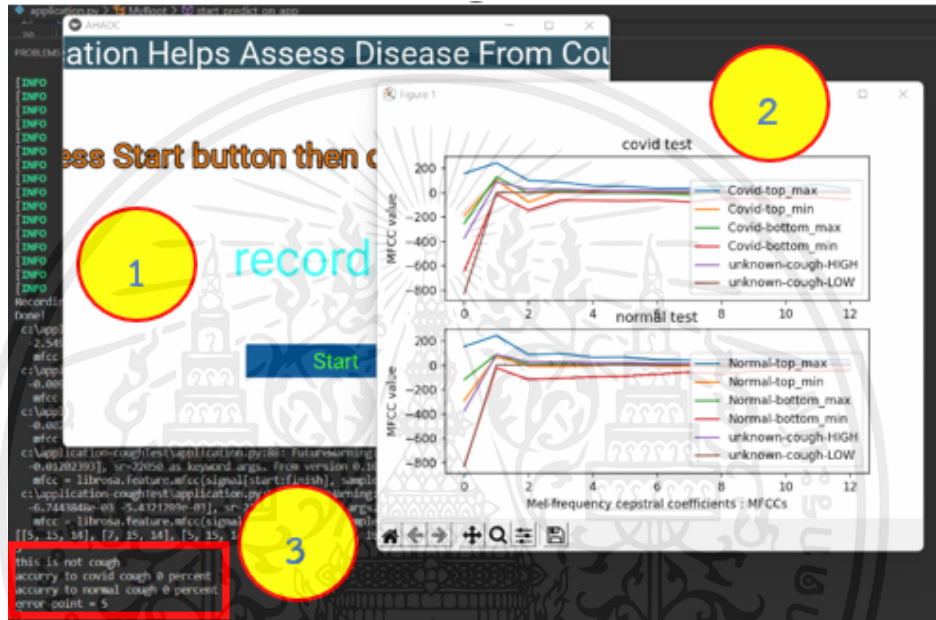
จากรูปที่ 3.45 แสดงหน้าต่างเทอร์มินอลเมื่อบันทึกเสียงแล้วผ่านการประมวลผลและทำนายผล โดยจะบอกถึงข้อมูลที่ผ่านการเปรียบเทียบของเสียงโอแต่ละส่วนกับฟิจเจอร์ในบรรทัดอาเรย์ แต้มผิดพลาด ที่น้อยที่สุดที่นำเข้าประมวลผล ความตรงกันของเสียงโอส่วนนั้นกับฟิจเจอร์เป็นเปอร์เซ็นต์ และการทำนายผลเสียงที่รับเข้านั้น



รูปที่ 3.46 กราฟการเปรียบเทียบเสียงไอและฟิจเจอร์

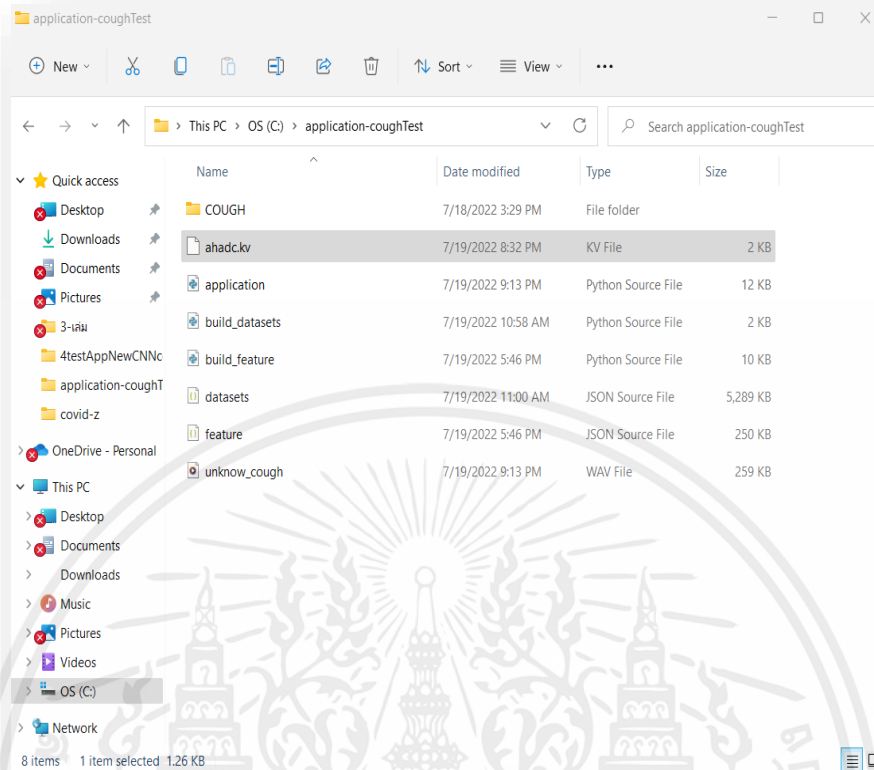
จากรูปที่ 3.46 กราฟการเปรียบเทียบเสียงที่รับเข้ากับฟิจเจอร์ที่ได้เทรนไว้ สร้างด้วยการเรียกใช้ไลบรารีพายพลอต จากกราฟตัวแปร MFCC value คือ ค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมล ตัวแปร MFCCs คือ ลำดับทั้ง 13 ของค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมล สีของเส้นกราฟที่แตกต่างกันตามข้อมูลค่าของสัมประสิทธิ์เซปสตรัมบนสเกลเมล โดยกราฟบนเป็นการเปรียบเทียบเสียงที่รับเข้ากับฟิจเจอร์ของเสียงไอโควิดเพื่อดูความตรงกันของลักษณะเสียงไอโควิดก่อนทำนายผล กราฟเส้นสีฟ้าคือลักษณะขอบเขตบนขีดจำกัดสูงสุดของค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลของฟิจเจอร์เสียงไอโควิด กราฟเส้นสีส้มคือลักษณะขอบเขตบนขีดจำกัดต่ำสุดของค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลของฟิจเจอร์

เสียงไอโควิดแล้วมีกราฟ ในกราฟบนของเสียงไอที่ต้องการประมวลผลมีชื่อว่า “unknown cough” ซึ่งมีขอบเขตบนและขอบเขตล่างที่นำไปเทียบกับฟิจเจอร์เสียงไอ หากค่าขอบเขตบนของเสียงไอที่รับเข้าอยู่ภายในขอบเขตบนขีดจำกัดสูงสุดและต่ำสุดของฟิจเจอร์ส่งผลให้ค่าความตรงกันกับฟิจเจอร์โควิดมีค่ามากขึ้น ในกราฟล่างเป็นการเปรียบเทียบเสียงที่รับเข้ากับฟิจเจอร์ของเสียงไอปกติเพื่อดูความตรงกันของลักษณะเสียงปกติก่อนทำนายผล



รูปที่ 3.47 ภาพรวมการทำงานของแอปพลิเคชัน

จากรูปที่ 3.47 ภาพรวมการทำงานของแอปพลิเคชัน หมายเลข 1 แสดงถึงหน้าต่างแอปพลิเคชันที่ต้องกดใช้งาน หมายเลข 2 แสดงถึงหน้าต่างแสดงกราฟค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลหลังกดใช้งานปุ่มบันทึกเสียงในหน้าต่างแอปพลิเคชัน หมายเลข 3 แสดงค่าความผิดพลาด ค่าความตรงกันของเสียงที่ทับเข้ากับฟิจเจอร์เสียงไอโควิดและเสียงไอปกติ โดยมีการประมวลผลก่อนการแสดงผลบนหน้าต่างแอปพลิเคชันบนหน้าจอ Display ของเครื่องคอมพิวเตอร์ระบบปฏิบัติการวินโดวส์ 11



รูปที่ 3.48 โฟลเดอร์หลังจากการทำงานของแอปพลิเคชัน

จากรูปที่ 3.48 โฟลเดอร์หลังจากการทำงานของแอปพลิเคชันที่แสดงอยู่ในโฟลเดอร์

แอปพลิเคชัน

บทที่ 4

การทดลองและผลการทดลอง

ในบทนี้จะกล่าวถึงการทดลองและผลการทดลองการทำงานของแอปพลิเคชันช่วยประเมินโรคจากอาการไอ ซึ่งมีหัวข้อการทดลองคือ การทดลองใช้งานแอปพลิเคชันบนคอมพิวเตอร์ระบบปฏิบัติการวินโดวส์ 11 การทดลองหาค่าความแม่นยำของพีจีเออร์จากการทราบจำนวนข้อมูลต่างกัน การทดลองหาค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลของเสียงไอคนเดียวแบบหนึ่งลูกคลื่นที่มีขนาดไฟล์หนึ่งวินาทีในลักษณะการไอต่างกัน การทดลองหาค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลของเสียงไอคนเดียวขนาดไฟล์หนึ่งวินาทีที่มีจำนวนลูกคลื่นต่างกัน การทดลองหาค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลของเสียงไอคนเดียวจำนวนหนึ่งลูกคลื่นที่มีขนาดไฟล์หนึ่งวินาทีแบบมีและไม่มีตัวกรองเสียงในอุปกรณ์รับเสียง และการทดลองทำนายผลลัพธ์ของแอปพลิเคชันด้วยไฟล์เสียง

4.1 การทดลองแอปพลิเคชันบนคอมพิวเตอร์ระบบปฏิบัติการวินโดวส์ 11

การทดลองใช้งานแอปพลิเคชันบนคอมพิวเตอร์ระบบปฏิบัติการวินโดวส์ 11 เป็นการตรวจสอบความถูกต้องการทำงานในตัวโปรแกรมของแอปพลิเคชันที่ใช้งานบนคอมพิวเตอร์ระบบปฏิบัติการวินโดวส์ 11 รวมถึงการกดปุ่มบันทึกเสียงและการแสดงผลบนหน้าต่างแอปพลิเคชันทั้งก่อนและหลังกดปุ่ม

4.1.1 วัตถุประสงค์

1. เพื่อเปิดใช้งานแอปพลิเคชันบนระบบปฏิบัติการวินโดวส์ 11
2. เพื่อทดสอบปุ่มบันทึกเสียงบนหน้าต่างแอปพลิเคชัน
3. เพื่อทดสอบการแสดงผลของหน้าต่างของแอปพลิเคชันก่อนและหลังบันทึกเสียง

4.1.2 อุปกรณ์ในการทดลอง

1. คอมพิวเตอร์ระบบปฏิบัติการวินโดวส์ 11
2. โปรแกรม Visual Studio Code
3. ไฟล์แอปพลิเคชันทั้งหมด

4.1.3 วิธีการทดลอง

1. นำแอปพลิเคชันไปบนคอมพิวเตอร์ระบบปฏิบัติการวินโดวส์ 11
2. เปิดแอปพลิเคชันด้วยโปรแกรม Visual Studio Code
3. กดปุ่มบันทึกเสียง

4. ทำเสียงไอ และเสียงต่าง ๆ เพื่อทดสอบการแสดงผลบนหน้าต่างแอปพลิเคชัน

4.1.4 ผลการทดลอง

1. เมื่อกดเปิดแอปพลิเคชันบนคอมพิวเตอร์ระบบปฏิบัติการวินโดวส์ 11 จะได้หน้าต่างแอปพลิเคชันช่วยประเมินโรคจากอาการไอ มีบรรทัดข้อความ “Press Start button then cough in 3 seconds” บรรทัดข้อความแสดงผลสีฟ้า “ready to record” เป็นข้อความที่แสดงผลบนหน้าต่างของแอปพลิเคชันที่แสดงการเตรียมพร้อมที่ให้ผู้ใช้งานอัดเสียง และมีปุ่มกด Start สำหรับเริ่มการบันทึกเสียงเมื่อกด ดังรูปที่ 4.1



รูปที่ 4.1 หน้าต่างแสดงข้อความพร้อมที่จะเริ่มบันทึกเสียงไอ

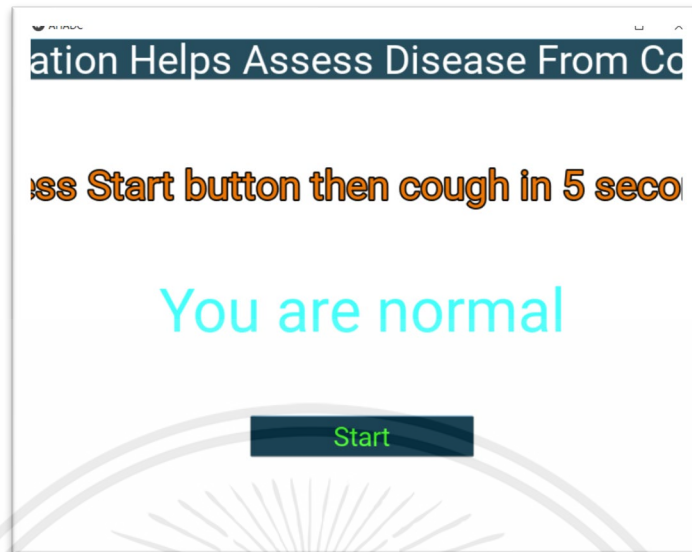
2. เมื่อกดปุ่ม Start ทำให้แอปพลิเคชันเริ่มการบันทึกเสียง โดยมีการแสดงผลข้อความคำว่า “recording” บนหน้าต่างแอปพลิเคชันหลังจากการกดปุ่มบันทึกเสียง เมื่อขึ้นข้อความดังกล่าวบนหน้าต่างแสดงผลของแอปพลิเคชัน ให้ผู้ทดลองสามารถเปล่งเสียงไอบนอุปกรณ์รับเสียงที่ได้เชื่อมต่อไว้เพื่อนำเสียงที่บันทึกไปวิเคราะห์ได้ ดังรูปที่ 4.1



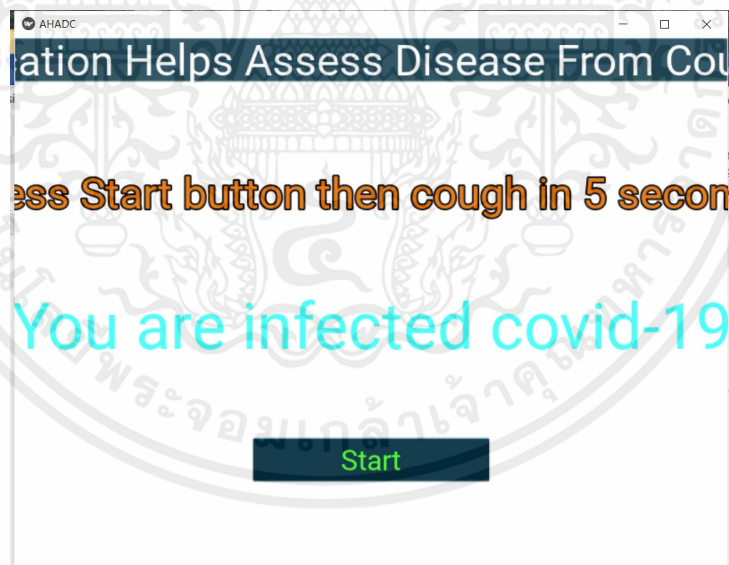
รูปที่ 4.2 หน้าต่างแสดงข้อความกำลังบันทึกเสียง

จากรูปที่ 4.2 แสดงหน้าต่างแสดงข้อความกำลังบันทึกเสียงขณะที่ผู้ใช้งานกำลังบันทึกเสียงไอ

3. เมื่อผู้ทดลองบันทึกเสียงแล้วมีการประมวลผลเสียงที่รับเข้าเสร็จ จะแสดงผลการวิเคราะห์เสียงบนหน้าต่างแอปพลิเคชันในบริเวณข้อความแสดงผลสีฟ้าในรูปแบบการทำนายผลต่าง ๆ ที่แอปพลิเคชันวิเคราะห์ได้ โดยผลการวิเคราะห์จะแสดงคำว่า “You are normal” เมื่อเสียงที่รับเข้ามีลักษณะตรงกับพีจเจอร์ของเสียงไอคนที่เป็นปกติ ดังรูปที่ 4.3



รูปที่ 4.3 หน้าต่างแสดงผลการประมวลผลเสียงไอปกติ



รูปที่ 4.4 หน้าต่างแสดงผลการประมวลผลเสียงติดเชื้อโควิด19

4. หน้าต่างแอปพลิเคชันบริเวณแสดงผลจากการวิเคราะห์มีการแสดงคำว่า “You are infected covid-19” เมื่อเสียงที่รับเข้ามีลักษณะตรงกับฟิเจอร์เสียงไอคนที่เป็นโควิด ดังรูปที่ 4.4

5. เมื่อเสียงที่บันทึกไม่มีความตรงกันกับฟิจเจอร์เสียงไอคนที่เป็นโควิดและคนที่เป็นปกติ หรือมีค่าความตรงกันกับเสียงไอน้อย หน้าต่างแอปพลิเคชันบริเวณแสดงผลพัธ์การวิเคราะห์แสดงคำว่า “this is not cough” ดังรูปที่ 4.5



รูปที่ 4.5 หน้าต่างแสดงผลพัธ์การประมวลผลเสียงที่ไม่ใช่เสียงไอ

4.2 การทดลองหาความแม่นยำของฟิจเจอร์จากการเทรน

การทดลองหาค่าความแม่นยำของฟิจเจอร์จากการเทรนจำนวนข้อมูลต่างกันเป็นการสร้าง ฟิจเจอร์จากการเทรนข้อมูลในรูปแบบที่ข้อมูลนั้นมีจำนวนและมีขนาดไฟล์เสียงที่ต่างกัน แล้วนำฟิจเจอร์ที่ ได้นั้นมาเปรียบเทียบกับไฟล์เสียงที่นำไปสร้างฟิจเจอร์ทั้งหมดเพื่อให้ได้ค่าความแม่นยำในการทำนายผล ว่าเสียงที่สร้างฟิจเจอร่นั้นมีความตรงกันกับฟิจเจอร์มากน้อยเพียงใด

4.2.1 วัตถุประสงค์

1. เพื่อหาลักษณะการสร้างฟิจเจอร์ค่าความแม่นยำสูง
2. เพื่อปรับปรุงโปรแกรมการสร้างฟิจเจอร์
3. เพื่อสร้างฟิจเจอร์ที่ทำให้ค่าความแม่นยำมากขึ้น

4.2.2 อุปกรณ์ในการทดลอง

1. คอมพิวเตอร์ระบบปฏิบัติการวินโดวส์ 11

2. โปรแกรม Visual Studio Code
3. ไฟล์แอปพลิเคชันทั้งหมด
4. ไฟล์เสียงไอโควิดและเสียงไอปกติจำนวนและขนาดต่าง ๆ

4.2.3 วิธีการทดลอง

1. นำไฟล์เสียงที่ต้องการสร้างฟิจเจอร์ไปไว้ในโฟลเดอร์ไฟล์เสียงตามแต่ละประเภท
2. เปิดแอปพลิเคชันด้วยโปรแกรม Visual Studio Code
3. เปิดและใช้งานโปรแกรมสร้างชุดข้อมูล (datasets)
4. เปิดและใช้งานโปรแกรมสร้างฟิจเจอร์จะได้ฟิจเจอร์และค่าความแม่นยำออกมา
5. ทดลองซ้ำโดยการปรับเปลี่ยนลักษณะชุดข้อมูลไฟล์เสียงทั้งจำนวนและขนาด

4.2.4 ผลการทดลอง

ค่าความแม่นยำของฟิจเจอร์นั้นเปลี่ยนไปเมื่อจำนวนหรือขนาดของไฟล์เสียงไอที่ใช้ในการสร้างฟิจเจอร์เปลี่ยน โดยค่าความแม่นยำมีค่าสูงเมื่อจำนวนไฟล์ที่เทรนมีน้อยและค่าความแม่นยำจะลดลงอย่างมากเมื่อการเทรนจำนวนไฟล์ที่มากขึ้น ค่าความตรงกันของเสียงไอโควิดมักมีค่าน้อยกว่าค่าความตรงกันของเสียงไอปกติ เมื่อเพิ่มจำนวนไฟล์ขนาดมากกว่า 1 วินาที ทำให้ค่าของไม่สามารถระบุเสียงได้เพิ่มขึ้นอย่างมาก และการทดลองให้ค่าความแม่นยำมีความผันผวนสูง ดังตารางที่ 4.1

ตารางที่ 4.1 การทดลองหาค่าความแม่นยำของฟิจเจอร์จากการเทรนจำนวนข้อมูลต่างกัน

ลำดับ การ ทดสอบ ไฟล์ เสียง	จำนวนเสียง covid (ไฟล์)		จำนวนเสียงปกติ (ไฟล์)		ความตรงกัน ของเสียง		ไม่ สามารถ ระบุ เสียงได้ (%)	ค่าความ แม่นยำ (%)
	ขนาด 1 วินาที (ไฟล์)	ขนาด มากกว่า 1วินาที (ไฟล์)	ขนาด 1 วินาที (ไฟล์)	ขนาด มากกว่า 1วินาที (ไฟล์)	เสียง covid (%)	เสียง ปกติ (%)		
1	5	0	5	0	100	100	0	100
2	10	0	10	0	100	100	0	100
3	15	0	15	0	100	100	0	100
4	20	0	20	0	90	100	5	95
5	25	0	25	0	60	100	19	80

ตารางที่ 4.1 การทดลองหาค่าความแม่นยำของพีเจอรจากการเทรนจำนวนข้อมูลต่างกัน (ต่อ)

ลำดับ การ ทดสอบ ไฟล์ เสียง	จำนวนเสียง covid (ไฟล์)		จำนวนเสียงปกติ (ไฟล์)		ความตรงกัน ของเสียง		ไม่ สามารถ ระบุ เสียงได้ (%)	ค่าความ แม่นยำ (%)
	ขนาด 1 วินาที (ไฟล์)	ขนาด มากกว่า 1วินาที (ไฟล์)	ขนาด 1 วินาที (ไฟล์)	ขนาด มากกว่า 1วินาที (ไฟล์)	เสียง covid (%)	เสียง ปกติ (%)		
6	30	0	30	17	58	85	21	72
7	35	0	35	47	45	81	26	63
8	40	0	40	76	38	83	24	60
9	45	0	45	76	17	89	19	53
10	50	0	50	76	59	79	26	69
11	53	0	55	0	28	91	18	59
12	30	0	30	0	50	100	22	75
13	35	0	35	0	34	100	29	67
14	45	0	45	0	24	100	32	62
15	53	0	55	0	19	15	21	67
16	53	0	191	0	19	65	30	42
17	53	0	191	26	19	5	72	12
18	53	0	191	76	19	4	68	11

หมายเหตุ ขนาด 1วินาที (ไฟล์) คือ ขนาดไฟล์เสียงที่มีช่วงเวลา 1 วินาที

ขนาดมากกว่า 1วินาที (ไฟล์) คือ ขนาดไฟล์เสียงที่มีช่วงเวลามากกว่า 1 วินาที

เสียง covid (%) คือ เสียงไอโควิดซึ่งคิดเปอร์เซ็นต์

เสียงปกติ (%) คือ เสียงไอปกติซึ่งคิดเปอร์เซ็นต์

ไม่สามารถระบุเสียงได้ (%) คือ ไม่สามารถระบุเสียงได้ซึ่งคิดเปอร์เซ็นต์

ค่าความแม่นยำ (%) คือ ค่าความแม่นยำในแต่ละครั้งของการเทรนซึ่งคิดเปอร์เซ็นต์

4.3 การทดลองเสียงที่มีลักษณะการไต่ต่างกัน

การทดลองหาค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลของเสียงไอคนเดียวแบบหนึ่งลูกคลื่นที่มีขนาดไฟล์หนึ่งวินาทีในลักษณะการไต่ต่างกัน โดยการนำค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลของขอบเขตบนและขอบเขตล่างของเสียงไอคนเดียวแบบหนึ่งลูกคลื่นที่มีขนาดไฟล์หนึ่งวินาทีที่มีลักษณะแตกต่างกันมาเปรียบเทียบ เพื่อสังเกตการเปลี่ยนแปลงของค่าสัมประสิทธิ์และปรับปรุงการสร้างฟิจเจอร์

4.3.1 วัตถุประสงค์

1. เพื่อหาความเชื่อมโยงของการไต่ที่มีลักษณะการไต่ต่างกัน
2. เพื่อปรับปรุงโปรแกรมการสร้างฟิจเจอร์
3. เพื่อสร้างฟิจเจอร์ที่ทำให้ค่าความแม่นยำมากขึ้น

4.3.2 อุปกรณ์ในการทดลอง

1. คอมพิวเตอร์ระบบปฏิบัติการวินโดวส์ 11
2. โปรแกรม Visual Studio Code
3. ไฟล์แอปพลิเคชันทั้งหมด
4. ไฟล์เสียงไอคนเดียวขนาดหนึ่งวินาทีที่มีลักษณะการไต่ต่างกัน

4.3.3 วิธีการทดลอง

1. เปิดแอปพลิเคชันด้วยโปรแกรม Visual Studio Code
2. นำไฟล์ที่ต้องการทดลองใส่ในโฟลเดอร์สำหรับเสียงไอที่ต้องการวิเคราะห์
3. แสดงค่าสัมประสิทธิ์บนสเกลเมลที่ผ่านการหาขอบเขตบนและขอบเขตล่าง
4. ทดลองซ้ำโดยการเปลี่ยนไฟล์เสียงไอในโฟลเดอร์สำหรับวิเคราะห์

4.3.4 ผลการทดลอง

จากการทดลองพบว่าค่าค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมล ในขอบเขตบนและขอบเขตล่างของเสียงไอมีการเปลี่ยนแปลงตามลักษณะการไต่ที่ต่างกัน ซึ่งค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลของแต่ละลักษณะการไต่จะมีค่าใกล้เคียงกันในแต่ละลำดับ ตั้งแต่ลำดับที่ 1 ถึงลำดับที่ 13 โดยขอบเขตบนของทุกลักษณะการไต่ ในลำดับที่ 1 ค่าที่ได้จะอยู่ระหว่าง 103.42 ถึง 142.18 และในลำดับที่ 2 ค่าที่ได้มีค่าระหว่าง 73.05 ถึง 144.63 ซึ่งสังเกตได้ว่าระยะห่างของค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลระหว่างลำดับที่ 2 มีความมากกว่าระยะห่างของค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลลำดับที่ 1 และค่าของลำดับที่ 3 มีค่าน้อยกว่าลำดับที่ 1 กับ 2 ดังตารางที่ 4.2 ในส่วนของขอบเขตล่างของทุกลักษณะการไต่ ในลำดับที่ 1 ค่าที่ได้จะอยู่ระหว่าง -446.35 ถึง -579.83 และในลำดับที่ 2 ค่าที่ได้มีค่าระหว่าง -

1.25 ถึง 33.42 ซึ่งชี้ให้เห็นว่า ค่าในลำดับที่ 2 มีค่ามากกว่า ค่าในลำดับที่ 1 แต่ระยะห่างระหว่างค่าสัมประสิทธิ์ของลำดับที่ 1 มีมากกว่าระยะห่างระหว่างค่าสัมประสิทธิ์ลำดับ 2 ดังตารางที่ 4.3

ตารางที่ 4.2 แสดงค่าความแตกต่างที่มีลักษณะการไต่ต่างกันขอบเขตบน

สัมประสิทธิ์ ลำดับที่	เสี่ยงไอคอนเดียวแบบ 1 ลูกคลื่น ขนาดไฟล์ 1 วินาที (high value of MFCC)					
	ลักษณะ แบบที่ 1	ลักษณะ แบบที่ 2	ลักษณะ แบบที่ 3	ลักษณะ แบบที่ 4	ลักษณะ แบบที่ 5	ลักษณะ แบบที่ 6
1	124.04	142.18	134.09	103.42	105.31	120.95
2	144.63	137.80	139.09	100.03	83.93	73.05
3	18.65	12.19	18.84	20.51	17.31	12.26
4	20.30	20.03	12.93	21.52	27.55	12.95
5	20.83	27.42	28.91	30.66	26.19	21.26
6	17.05	13.42	39.18	17.60	20.82	13.93
7	4.23	4.22	4.68	9.83	4.63	9.50
8	18.60	17.72	19.53	26.27	15.07	13.19
9	0.66	0.44	2.62	-0.99	2.13	0.31
10	8.46	24.89	27.19	18.69	10.96	10.70
11	11.72	11.27	36.58	16.09	13.35	10.50
12	13.35	9.88	17.33	26.35	7.03	12.57
13	23.46	15.62	23.34	20.00	16.25	14.06

ตารางที่ 4.3 แสดงค่าความแตกต่างที่มีลักษณะการไต่ต่างกันขอบเขตล่าง

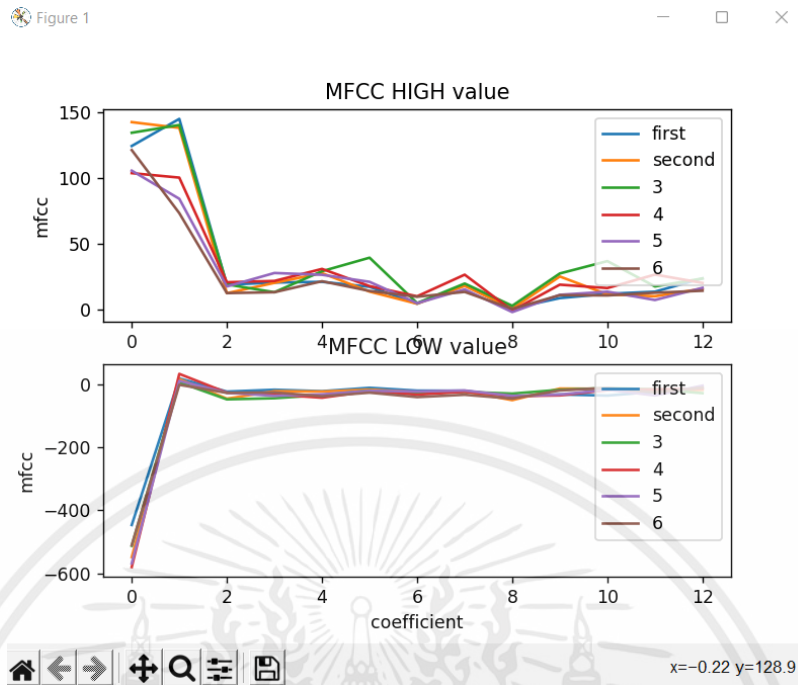
สัมประสิทธิ์ ลำดับที่	เสี่ยงไอคอนเดียวแบบ 1 ลูกคลื่น ขนาดไฟล์ 1 วินาที (low value of MFCC)					
	ลักษณะ แบบที่ 1	ลักษณะ แบบที่ 2	ลักษณะ แบบที่ 3	ลักษณะ แบบที่ 4	ลักษณะ แบบที่ 5	ลักษณะ แบบที่ 6
1	-446.35	-548.49	-512.47	-579.83	-568.09	-510.68
2	18.95	16.85	6.52	33.42	11.23	-1.25
3	-22.75	-4530	-47.50	-26.16	-26.34	-26.87
4	-16.80	-21.55	-43.92	-32.41	-36.43	-25.37
5	-21.47	-22.72	-33.86	-42.22	-31.58	-37.71

ตารางที่ 4.3 แสดงค่าความแตกต่างที่มีลักษณะการไต่ต่างกันขอบเขตล่าง (ต่อ)

สัมประสิทธิ์ ลำดับที่	เสียงไอคนเดียวแบบ 1 ลูกคลื่น ขนาดไฟล์ 1 วินาที (low value of MFCC)					
	ลักษณะ แบบที่ 1	ลักษณะ แบบที่ 2	ลักษณะ แบบที่ 3	ลักษณะ แบบที่ 4	ลักษณะ แบบที่ 5	ลักษณะ แบบที่ 6
6	-10.68	-16.33	-19.36	-21.71	-20.23	-26.17
7	-19.71	-24.59	-28.22	-32.44	-22.63	-40.27
8	-19.99	-21.93	-23.63	-23.41	-19.31	-32.85
9	-38.47	-50.07	-29.14	-37.26	-36.24	-44.99
10	-31.92	-13.15	-17.13	-34.86	-32.31	-19.29
11	-35.45	-13.99	-11.58	-18.27	-16.68	-10.89
12	-23.25	-22.25	-15.48	-15.43	-35.12	-25.06
13	-1050	-17.61	-27.78	-14.16	-3.89	-10.40

หมายเหตุ ตารางที่ 4.2 และตารางที่ 4.3 ในช่องลักษณะในแต่ละแบบ คือ ค่าของสัมประสิทธิ์เซปสตรัมบนสเกลเมลหรือค่าพีจเจอร์ที่ใช้ในการกำหนดขอบเขตของลายเสียง (ลายเสียงคือเสียงแต่ละประเภทที่แตกต่างกัน)

จากตารางที่ 4.2 และตารางที่ 4.3 แสดงการทดลองหาค่าของสัมประสิทธิ์เซปสตรัมบนสเกลเมลขอบเขตสูงสุดและขอบเขตต่ำสุดของเสียงไอคนเดียวแบบหนึ่งลูกคลื่นที่มีขนาดไฟล์หนึ่งวินาทีในลักษณะที่มีการไต่ต่างกัน โดยมีการแสดงค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมล (Mel frequency cepstrum coefficient, MFCC) ในรูปแบบของค่าอาเรย์ (Array)



รูปที่ 4.6 กราฟฟิจเจอร์เสียงโอคนเดียว 1 ลูกคลื่น ขนาดไฟล์ 1 วินาที ในลักษณะการโอต่างกัน

จากรูปที่ 4.6 แสดงกราฟฟิจเจอร์ทั้งขอบเขตบนและขอบเขตล่างของเสียงโอคนเดียว 1 ลูกคลื่น ขนาดไฟล์ 1 วินาที ในลักษณะต่าง ๆ โดยเส้นกราฟแต่ละลักษณะจะมีสีที่แตกต่างกันเพื่อให้สังเกตการเปลี่ยนแปลงของค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลได้อย่างชัดเจน ในลักษณะการโอที่ 1 จะเป็นสีฟ้า และในลักษณะการโอที่ 2 เป็นสีส้ม รวมถึงลักษณะการโออื่น ๆ ที่มีสีกำกับบนกราฟ จะสังเกตได้ว่าค่าสัมประสิทธิ์ที่ได้ในขอบเขตบนจะมีความเปลี่ยนแปลงมากกว่าค่าสัมประสิทธิ์ในขอบเขตล่าง

4.4 การทดลองเสียงโอที่มีจำนวนลูกคลื่นต่างกัน

การทดลองหาค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลของเสียงโอคนเดียวขนาดไฟล์หนึ่งวินาทีที่มีจำนวนลูกคลื่นต่างกัน โดยการนำค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลขอบเขตบนและขอบเขตล่างของเสียงโอคนเดียวขนาดไฟล์หนึ่งวินาทีที่มีจำนวนลูกคลื่นต่างกันมาเปรียบเทียบ เพื่อสังเกตการเปลี่ยนแปลงของค่าสัมประสิทธิ์และปรับปรุงการสร้างฟิจเจอร์

4.4.1 วัตถุประสงค์

1. เพื่อหาความเชื่อมโยงของการโอที่มีลักษณะการโอต่างกัน
2. เพื่อปรับปรุงโปรแกรมการสร้างฟิจเจอร์

3. เพื่อสร้างพีจเจอร์ที่ทำให้ค่าความแม่นยำมากขึ้น

4.4.2 อุปกรณ์ในการทดลอง

1. คอมพิวเตอร์ระบบปฏิบัติการวินโดวส์ 11
2. โปรแกรม Visual Studio Code
3. ไฟล์แอปพลิเคชันทั้งหมด
4. ไฟล์เสียงไอคอนเดียวขนาดไฟล์หนึ่งวินาทีที่มีจำนวนลูกคลื่นต่างกัน

4.4.3 วิธีการทดลอง

1. เปิดแอปพลิเคชันด้วยโปรแกรม Visual Studio Code
2. นำไฟล์ที่ต้องการทดลองใส่ในโฟลเดอร์สำหรับเสียงไอที่ต้องการวิเคราะห์
3. แสดงค่าสัมประสิทธิ์บนสเกลเมลที่ผ่านการหาขอบเขตบนและขอบเขตล่าง
4. ทดลองซ้ำโดยการเปลี่ยนไฟล์เสียงไอในโฟลเดอร์สำหรับวิเคราะห์

4.4.4 ผลการทดลอง

จากการทดลองพบว่าค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมล ในขอบเขตบนและขอบเขตล่างของเสียงไอมีการเปลี่ยนแปลงตามลักษณะการไอที่ต่างกัน ซึ่งค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลของแต่ละลักษณะการไอก็จะมีค่าใกล้เคียงกันในแต่ละลำดับ ตั้งแต่ลำดับที่ 1 ถึงลำดับที่ 13 โดยขอบเขตบนของทุกจำนวนคลื่น ในลำดับที่ 1 ของคลื่นทั้งสี่จำนวนมีความแตกต่างกันอย่างเห็นได้ชัด โดยสามารถสังเกตการเพิ่มขึ้นของค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลตามจำนวนคลื่นที่เพิ่มขึ้นได้ หรือค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมล ลำดับที่ 4 ในขอบเขตล่างมีแนวโน้มที่จะมีค่ามากขึ้นตามจำนวนคลื่นที่เพิ่ม ดังตารางที่ 4.4

ตารางที่ 4.4 การทดลองเสียงไอที่มีจำนวนลูกคลื่นต่างกัน

สัมประสิทธิ์บนสเกลเมล		เสียงไอคนเดียว ขนาดไฟล์1วินาที ลูกคลื่นต่างจำนวน			
ขอบเขต	ลำดับที่	แบบ 1 ลูกคลื่น	แบบ 2 ลูกคลื่น	แบบ 3 ลูกคลื่น	แบบ 4 ลูกคลื่น
บน	1	142.18	152.48	137.05	192.25
	2	144.63	146.35	139.62	152.25
	3	20.51	26.14	28.69	13.22
	4	27.55	28.44	30.85	35.30

ตารางที่ 4.4 การทดลองเสียงโอที่มีจำนวนลูกคลื่นต่างกัน (ต่อ)

สัมประสิทธิ์บนสเกลเมล		เสียงโคนเดี่ยว ขนาดไฟล์ 1 วินาที ลูกคลื่นต่างจำนวน			
ขอบเขต	ลำดับที่	แบบ 1 ลูกคลื่น	แบบ 2 ลูกคลื่น	แบบ 3 ลูกคลื่น	แบบ 4 ลูกคลื่น
บน	5	30.65	31.94	45.05	10.05
	6	39.18	32.72	25.35	11.19
	7	9.82	20.46	20.55	7.54
	8	26.27	23.28	18.83	-0.67
	9	2.62	8.87	28.97	-1.44
	10	27.19	32.69	39.34	26.56
	11	36.58	43.55	42.99	19.89
	12	26.35	25.62	21.28	12.46
	13	23.46	29.24	22.99	-0.88

หมายเหตุ สัมประสิทธิ์บนสเกลเมล คือค่าพีจเจอร์ที่ใช้ในการกำหนดขอบเขตของลายเสียง (ลายเสียงคือเสียงแต่ละประเภทที่แตกต่างกัน)

ตารางที่ 4.4 การทดลองเสียงโอที่มีจำนวนลูกคลื่นต่างกัน (ต่อ)

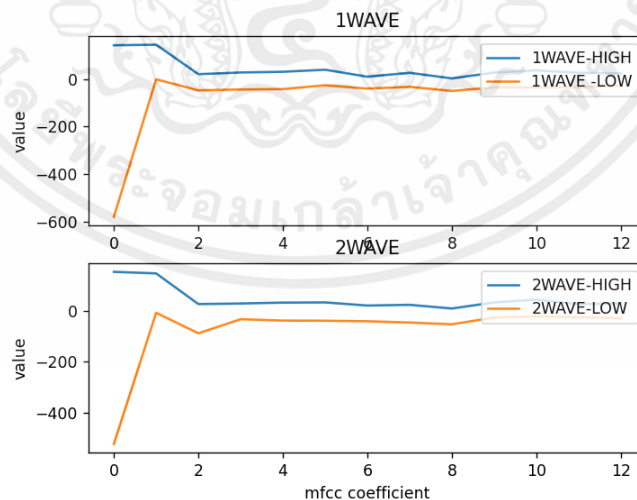
สัมประสิทธิ์บนสเกลเมล		เสียงโคนเดี่ยว ขนาดไฟล์ 1 วินาที ลูกคลื่นต่างจำนวน			
ขอบเขต	ลำดับที่	แบบ 1 ลูกคลื่น	แบบ 2 ลูกคลื่น	แบบ 3 ลูกคลื่น	แบบ 4 ลูกคลื่น
ล่าง	1	-579.83	-521.31	-539.79	-397.86
	2	-1.25	-8	-0.29	12.35
	3	-47.50	-88.91	-71.77	-59.43
	4	-43.92	-33.10	-26.49	-18.18
	5	-42.22	-38.40	-34.43	-30.36
	6	-26.17	-38.98	-37.91	-30.34
	7	-40.27	-41.15	-47.93	-42.29

ตารางที่ 4.4 การทดลองเสียงโอที่มีจำนวนลูกคลื่นต่างกัน (ต่อ)

สัมประสิทธิ์บนสเกลเมล		เสียงโอคนเดียว ขนาดไฟล์ 1 วินาที ลูกคลื่นต่างจำนวน			
ขอบเขต	ลำดับที่	แบบ 1 ลูกคลื่น	แบบ 2 ลูกคลื่น	แบบ 3 ลูกคลื่น	แบบ 4 ลูกคลื่น
ล่าง	8	-32.85	-46.14	-44.10	-50.02
	9	-50.07	-53.03	-55.07	-65.37
	10	-34.86	-26.37	-31.38	-35.12
	11	-35.45	-22.09	-22.88	-20.44
	12	-35.12	-25.13	-23.81	-24.20
	13	-27.78	-29.11	-37.09	-29.16

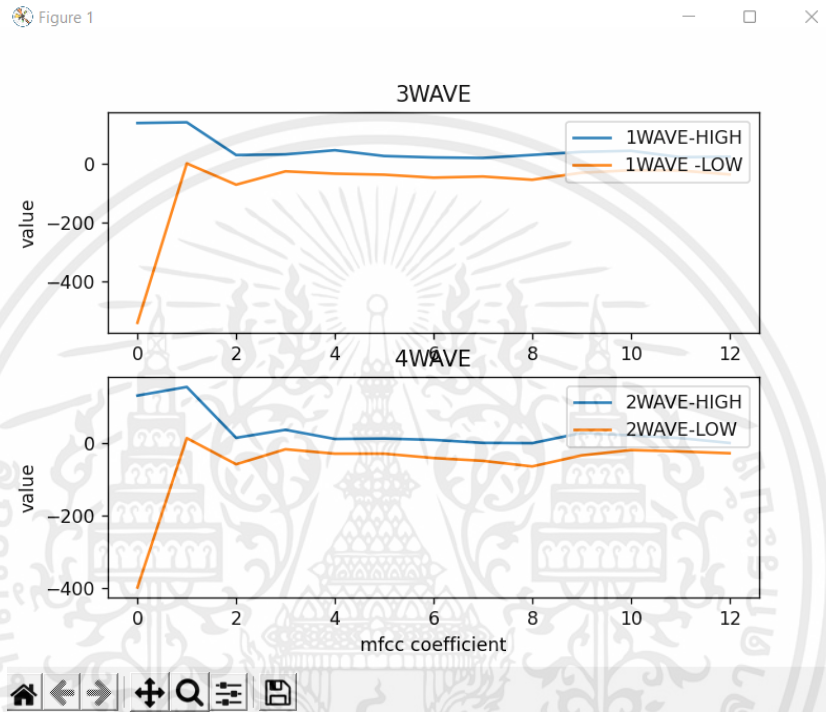
หมายเหตุ สัมประสิทธิ์บนสเกลเมล คือค่าพีจเจอร์ที่ใช้ในการกำหนดขอบเขตของลายเสียง (ลายเสียงคือเสียงแต่ละประเภทที่แตกต่างกัน)

จากตารางที่ 4.4 แสดงค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลของเสียงโอคนเดียวขนาดไฟล์หนึ่งวินาทีที่มีจำนวนลูกคลื่นต่างกัน มีการแสดงค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมล (Mel frequency cepstrum coefficient, MFCC) ในรูปแบบของค่าอาเรย์ (Array)



รูปที่ 4.7 เสียงโอคนเดียวขนาดไฟล์หนึ่งวินาทีแบบหนึ่งและสองลูกคลื่น

จากรูปที่ 4.7 แสดงเสียงโอคนเดี่ยวขนาดไฟล์หนึ่งวินาทีที่แสดงขอบเขตบนและขอบเขตล่างของเสียงโอแบบหนึ่งลูกคลื่นและสองลูกคลื่น โดยลักษณะกราฟมีความใกล้เคียงกัน ความแตกต่างของกราฟเสียงโอแบบหนึ่งลูกคลื่นและสองลูกคลื่นคือขอบเขตล่างของค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลลำดับที่ 2 ที่เสียงโอสองลูกคลื่นมีค่าต่ำกว่าเสียงโอแบบหนึ่งลูกคลื่นอย่างเห็นได้ชัด



รูปที่ 4.8 เสียงโอคนเดี่ยวขนาดไฟล์หนึ่งวินาทีแบบสามและสี่ลูกคลื่น

จากรูปที่ 4.8 แสดงกราฟพีจเจอร์ทั้งขอบเขตบนและขอบเขตล่างของเสียงโอคนเดี่ยว 1 ลูกคลื่นขนาดไฟล์ 1 วินาที ในลักษณะต่าง ๆ โดยค่า mfcc คือค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมล ค่า coefficient คือค่าลำดับทั้ง 13 ของค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมล เส้นกราฟแต่ละลักษณะจะมีสีที่แตกต่างกันเพื่อให้สังเกตการเปลี่ยนแปลงของค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลได้อย่างชัดเจน ในลักษณะการไอที่ 1 จะเป็นสีฟ้า และในลักษณะการไอที่ 2 เป็นสีส้ม รวมถึงลักษณะการไออื่น ๆ ที่มีสีกำกับบนกราฟ จะสังเกตได้ว่าค่าสัมประสิทธิ์ที่ได้ในขอบเขตบนจะมีความเปลี่ยนแปลงมากกว่าค่าสัมประสิทธิ์ในขอบเขตล่าง

4.5 การทดลองเสียงไอแบบมีตัวกรองเสียงและไม่มีตัวกรองเสียง

การทดลองหาค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลของเสียงไอคนเดียวจำนวนหนึ่งลูกคลื่นที่มีขนาดไฟล์หนึ่งวินาทีแบบมีและไม่มีตัวกรองเสียงในอุปกรณ์รับเสียง โดยการนำค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลขอบเขตบนและขอบเขตล่างของเสียงไอแบบมีและไม่มีตัวกรองเสียงในอุปกรณ์รับเสียงมาเปรียบเทียบ เพื่อสังเกตการเปลี่ยนแปลงของค่าสัมประสิทธิ์และปรับปรุงการสร้างฟิจเจอร์

4.5.1 วัตถุประสงค์

1. เพื่อหาความเชื่อมโยงของการไอที่มีลักษณะการไอต่างกัน
2. เพื่อปรับปรุงโปรแกรมการสร้างฟิจเจอร์
3. เพื่อสร้างฟิจเจอร์ที่ทำให้ค่าความแม่นยำมากขึ้น

4.5.2 อุปกรณ์ในการทดลอง

1. คอมพิวเตอร์ระบบปฏิบัติการวินโดวส์ 11
2. โปรแกรม Visual Studio Code
3. ไฟล์แอปพลิเคชันทั้งหมด
4. ไฟล์เสียงไอคนเดียวขนาดไฟล์หนึ่งวินาทีหนึ่งลูกคลื่นแบบมีและไม่มีตัวกรองเสียงในอุปกรณ์รับเสียง

4.5.3 วิธีการทดลอง

1. เปิดแอปพลิเคชันด้วยโปรแกรม Visual Studio Code
2. นำไฟล์ที่ต้องการทดลองใส่ในโฟลเดอร์สำหรับเสียงไอที่ต้องการวิเคราะห์
3. แสดงค่าสัมประสิทธิ์บนสเกลเมลที่ผ่านการหาขอบเขตบนและขอบเขตล่าง
4. ทดลองซ้ำโดยการเปลี่ยนไฟล์เสียงไอในโฟลเดอร์สำหรับวิเคราะห์

4.5.4 ผลการทดลอง

จากการทดลองพบว่าค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลในขอบเขตบนและขอบเขตล่างของเสียงไอมีการเปลี่ยนแปลงตามลักษณะการไอที่ต่างกัน ซึ่งค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลของแต่ละลักษณะการไอจะมีค่าใกล้เคียงกันในแต่ละลำดับ ตั้งแต่ลำดับที่ 1 ถึงลำดับที่ 13 โดยขอบเขตบนของเสียงในลำดับที่ 1 เสียงที่ไม่มีตัวกรองมีค่ามากกว่าเสียงที่มีตัวกรอง และสังเกตได้ว่าค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลของเสียงที่มีตัวกรองส่วนมากมีค่าน้อยกว่าเสียงที่ไม่มีตัวกรอง ดังตารางที่ 4.5

ตารางที่ 4.5 การทดลองเสียงโอบแบบมีตัวกรองเสียงและไม่มีตัวกรองเสียงในอุปกรณ์รับเสียง

สัมประสิทธิ์บนสเกลเมล		เสียงโอบคนเดียว ขนาดไฟล์ 1 วินาที 1 ลูกคลื่น	
ขอบเขต	ลำดับที่	แบบที่ไม่มีตัวกรองเสียง	แบบที่มีตัวกรองเสียง
บน	1	142.19	-127.38
	2	144.63	193.75
	3	20.51	30.91
	4	27.55	88.72
	5	30.66	27.52
	6	39.18	33.57
	7	9.83	10.62
	8	26.27	12.37
	9	2.62	13.74
	10	27.19	16.52
	11	36.58	29.46
	12	26.35	15.89
	13	23.46	21.54

ตารางที่ 4.5 การทดลองเสียงโอบแบบมีตัวกรองเสียงและไม่มีตัวกรองเสียงในอุปกรณ์รับเสียง

สัมประสิทธิ์บนสเกลเมล		เสียงโอบคนเดียว ขนาดไฟล์ 1 วินาที 1 ลูกคลื่น	
ขอบเขต	ลำดับที่	แบบที่ไม่มีตัวกรองเสียง	แบบที่มีตัวกรองเสียง
ล่าง	1	-579.84	-752.62
	2	-1.26	0.0
	3	-47.50	-108.83
	4	-43.92	-27.43
	5	-42.23	-73.19

ตารางที่ 4.5 การทดลองเสียงโอบแบบมีตัวกรองเสียงและไม่มีตัวกรองเสียงในอุปกรณ์รับเสียง (ต่อ)

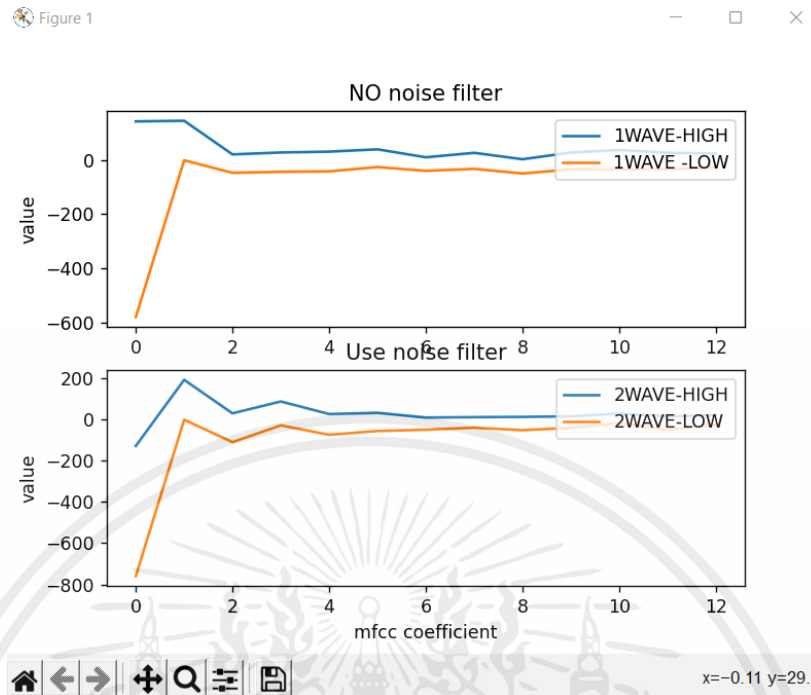
สัมประสิทธิ์บนสเกลเมล		เสียงโอบคนเดียว ขนาดไฟล์ 1 วินาที 1 ลูกคลื่น	
ขอบเขต	ลำดับที่	แบบที่ไม่มีตัวกรองเสียง	แบบที่มีตัวกรองเสียง
ล่าง	6	-26.17	-54.79
	7	-40.27	-48.50
	8	-32.85	-38.76
	9	-50.08	-50.60
	10	-34.87	-39.61
	11	-35.45	-16.89
	12	-35.12	-49.35
	13	-27.78	-23.77

หมายเหตุ สัมประสิทธิ์บนสเกลเมล คือค่าพีจเจอร์ที่ใช้ในการกำหนดขอบเขตของลายเสียง

(ลายเสียงคือเสียงแต่ละประเภทที่แตกต่างกัน)

เสียงโอบคนเดียว ขนาดไฟล์ 1 วินาที 1 ลูกคลื่น คือเสียงโอบคนเดียว ขนาดไฟล์ในช่วงเวลา 1 วินาที 1 ลูกคลื่น

จากตารางที่ 4.5 แสดงค่าสัมประสิทธิ์บนสเกลเมลของเสียงโอบคนเดียวขนาดไฟล์หนึ่งวินาที จำนวนหนึ่งลูกคลื่นแบบมีและไม่มีตัวกรองเสียงในอุปกรณ์รับเสียง (sound filter) มีการแสดงค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมล (Mel frequency cepstrum coefficient, MFCC) ในรูปแบบของค่าอาเรย์ (Array) จะสังเกตได้ว่าค่าบางตัวมีความใกล้เคียงกัน แต่บางค่ากลับแตกต่างกันมาก



รูปที่ 4.9 เสียงโอที่มีและไม่มีตัวกรองเสียงในอุปกรณ์รับเสียง

จากรูปที่ 4.9 แสดงกราฟลักษณะของค่าสัมประสิทธิ์บนสเกลเมลของเสียงโอที่มีและไม่มีตัวกรองในอุปกรณ์รับเสียง มีลักษณะกราฟที่ใกล้เคียงกันแต่ยังมีความแตกต่างในค่าที่ละเอียด กราฟที่มีตัวกรองเสียงจะมีความเรียบมากกว่ากราฟที่ไม่มีตัวกรองเสียง

4.6 การทดลองทำนายผลของแอปพลิเคชันด้วยไฟล์เสียง

การทดลองทำนายผลลัพธ์ของแอปพลิเคชันด้วยไฟล์เสียง โดยการนำค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลขอบเขตบนและขอบเขตล่างของเสียงโอที่ผ่านอุปกรณ์รับเสียงมาเปรียบเทียบกับฟิจเจอร์ เพื่อสังเกตการเปลี่ยนแปลงของค่าสัมประสิทธิ์และปรับปรุงการสร้างฟิจเจอร์

4.6.1 วัตถุประสงค์

1. เพื่อหาความเชื่อมโยงของการโอที่มีลักษณะการโอต่างกัน
2. เพื่อปรับปรุงโปรแกรมการสร้างฟิจเจอร์
3. เพื่อสร้างฟิจเจอร์ที่ทำให้ค่าความแม่นยำมากขึ้น

4.6.2 อุปกรณ์ในการทดลอง

1. คอมพิวเตอร์ระบบปฏิบัติการวินโดวส์ 11

2. โปรแกรม Visual Studio Code
3. ไฟล์แอปพลิเคชันทั้งหมด
4. ไฟล์เสียงที่ต้องการทำนายผล
5. ไฟล์เสียงสำหรับสร้างฟิจเจอร์ขนาด 1 วินาที ได้แก่ ไฟล์เสียงไอโควิดจำนวน 53 ไฟล์ และไฟล์เสียงไอปกติจำนวน 191 ไฟล์

4.6.3 วิธีการทดลอง

1. เปิดแอปพลิเคชันด้วยโปรแกรม Visual Studio Code
2. สร้างฟิจเจอร์
3. นำไฟล์ที่ต้องการทดลองใส่ในโพลเดอร์สำหรับเสียงไอที่ต้องการวิเคราะห์
4. ทำนายผลค่าความแม่นยำของฟิจเจอร์ที่เทียบกับเสียงที่ต้องการ
5. ทดลองซ้ำโดยการเปลี่ยนไฟล์เสียงไอในโพลเดอร์สำหรับวิเคราะห์
6. นำค่าการทำนายผลที่ได้คำนวณหาเปอร์เซ็นต์ความแม่นยำจากสมการที่

$$\text{ค่าความแม่นยำของเสียงที่ไม่ใช่เสียงไอ} = \frac{\text{เสียงที่ไม่ใช่เสียงไอที่ทำนายว่าเป็นเสียงไอ}}{\text{เสียงที่ไม่ใช่เสียงไอทั้งหมด}} \times 100\% \quad (4.1)$$

$$\text{ค่าความแม่นยำของเสียงไอโควิด} = \frac{\text{เสียงไอโควิดที่ทำนายว่าเป็นโควิด}}{\text{เสียงโควิดทั้งหมด}} \times 100\% \quad (4.2)$$

$$\text{ค่าความแม่นยำของเสียงไอปกติ} = \frac{\text{เสียงไอปกติที่ทำนายว่าเป็นปกติ}}{\text{เสียงปกติทั้งหมด}} \times 100\% \quad (4.3)$$

4.6.4 ผลการทดลอง

จากการทดลองพบว่าเมื่อใส่เสียงที่ไม่ใช่เสียงไอเข้าวิเคราะห์หาความตรงกันกับฟิจเจอร์ ความตรงกันกับเสียงไอโควิดมีค่าเป็น 0 ความตรงกันกับเสียงไอปกติมีค่าเป็น 0 มีค่าความผิดพลาด หรือ Error Point มีค่าเท่าหรือมากกว่า 1 ผลลัพธ์ในการทำนายผลคือ “ไม่ใช่เสียงไอ” จากการทดสอบเสียงที่ไม่ใช่เสียงไอนี้ได้ค่าความแม่นยำในการทำนายผลเสียงที่ไม่ใช่เสียงไอเป็น 100 เปอร์เซ็นต์ เมื่อนำเสียงไอปกติเข้าสู่การประมวลผล ค่าความตรงกันกับเสียงไอปกติอยู่ที่ 100 เปอร์เซ็นต์ ค่าความตรงกันกับเสียงไอโควิดอยู่ระหว่าง 76.92 ถึง 100 เปอร์เซ็นต์ มีค่าความผิดพลาดเท่ากับ 0 ผลลัพธ์ในการทำนายคือ “เสียงไอปกติ” ค่าความแม่นยำในการทำนายผลเสียงไอปกตินี้อยู่ที่ 77.78 เปอร์เซ็นต์ เมื่อนำเสียงไอโควิดเข้าสู่การประมวลผล ค่าความตรงกันกับเสียงไอปกติอยู่ระหว่าง 92.13 ถึง 100 เปอร์เซ็นต์ ค่าความตรงกันกับ

เสียงไอโควิดอยู่ที่ 100 เปอร์เซ็น มีค่าความผิดพลาดเท่ากับ 0 ผลลัพธ์ในการทำนายผลคือ “เสียงไอปกติ” และ “เสียงไอโควิด” ค่าความแม่นยำในการทำนายผลเสียงไอโควิดอยู่ที่ 25 เปอร์เซ็น

ตารางที่ 4.6 การทดลองทำนายผลลัพธ์ของแอปพลิเคชันด้วยไฟล์เสียง

ลำดับที่	ข้อมูลไฟล์เสียง	ความตรงกันกับเสียงไอ covid (100%)	ความตรงกันกับเสียงไอปกติ (100%)	Error Point	ทำนายผลลัพธ์
1	แมวร้อง1	0	0	2	ไม่ใช่เสียงไอ
2	แมวร้อง2	0	0	6	ไม่ใช่เสียงไอ
3	หมาเห่า1	0	0	3	ไม่ใช่เสียงไอ
4	หมาเห่า2	0	0	3	ไม่ใช่เสียงไอ
5	หมาเห่า3	0	0	2	ไม่ใช่เสียงไอ
6	เคาะประตู1	0	0	1	ไม่ใช่เสียงไอ
7	เคาะประตู2	0	0	1	ไม่ใช่เสียงไอ
8	ฝนตก1	0	0	1	ไม่ใช่เสียงไอ
9	ฝนตก2	0	0	1	ไม่ใช่เสียงไอ
10	ฝนตกฟ้าผ่า1	0	0	1	ไม่ใช่เสียงไอ
11	ฝนตกฟ้าผ่า2	0	0	2	ไม่ใช่เสียงไอ
12	เสียงไอลึกลับ	100	100	0	ปกติ
13	เสียงไอปกติ1	96.15	100	0	ปกติ
14	เสียงไอปกติ2	92.31	100	0	ปกติ
15	เสียงไอปกติ3	96.15	100	0	ปกติ
16	เสียงไอปกติ4	92.31	100	0	ปกติ
17	เสียงไอปกติ5	88.46	100	0	ปกติ
18	เสียงไอปกติ6	84.61	100	0	ปกติ
19	เสียงไอ covid1	100	100	0	ปกติ
20	เสียงไอ covid2	100	96.15	0	ติดเชื้อ
21	เสียงไอ covid3	100	100	0	ปกติ

ตารางที่ 4.6 การทดลองทำนายผลลัพธ์ของแอปพลิเคชันด้วยไฟล์เสียง (ต่อ)

ลำดับที่	ข้อมูลไฟล์เสียง	ความตรงกันกับเสียงไอ covid (100%)	ความตรงกันกับเสียงไอปกติ (100%)	Error Point	ทำนายผลลัพธ์
22	เสียงไอ covid4	100	100	0	ปกติ
23	เสียงไอ covid5	100	96.15	0	ติดเชื้อ
24	เสียงไอ covid6	100	96.15	0	ติดเชื้อ
25	เสียงไอ covid7	100	100	0	ปกติ
26	เสียงไอ covid8	100	92.13	0	ติดเชื้อ
27	เสียงไอ covid9	100	100	0	ปกติ
28	เสียงไอ covid10	100	100	0	ปกติ
29	เสียงไอ covid11	100	100	0	ปกติ
30	เสียงไอ covid12	100	100	0	ปกติ
31	เสียงไอ covid13	100	100	0	ปกติ
32	เสียงไอ covid14	100	100	0	ปกติ
33	เสียงไอ covid15	100	100	0	ปกติ
34	เสียงไอ covid16	100	100	0	ปกติ
35	เสียงไอ covid17	100	100	0	ปกติ
36	เสียงไอ covid18	100	100	0	ปกติ
37	เสียงไอ covid19	100	96.15	0	ติดเชื้อ
38	เสียงไอ covid20	100	100	0	ปกติ
39	เสียงไอปกติ	76.92	100	0	ปกติ
40	เสียงไอปกติ	88.46	100	0	ปกติ
41	เสียงไอปกติ	100	100	0	ปกติ
42	เสียงไอปกติ	100	100	0	ปกติ
43	เสียงไอปกติ	92.31	100	0	ปกติ
44	เสียงไอปกติ	92.31	100	0	ปกติ

ตารางที่ 4.6 การทดลองทำนายผลลัพธ์ของแอปพลิเคชันด้วยไฟล์เสียง (ต่อ)

ลำดับที่	ข้อมูลไฟล์เสียง	ความตรงกันกับเสียงไอ covid (100%)	ความตรงกันกับเสียงไอปกติ (100%)	Error Point	ทำนายผลลัพธ์
45	เสียงไอปกติ	100	100	0	ปกติ
46	เสียงไอปกติ	88.46	100	0	ปกติ
47	เสียงไอปกติ	96.15	100	0	ปกติ
48	เสียงไอปกติ	100	100	0	ปกติ
49	เสียงไอปกติ	92.31	100	0	ปกติ
50	เสียงไอปกติ	92.31	100	0	ปกติ

หมายเหตุ Error Point หรือแต้มค่าความผิดพลาด ถ้าหากว่ามีแต้มค่าความผิดพลาดแสดงว่าไม่ใช่เสียงไอปกติและไม่ใช่เสียงไอโควิด

จากตารางที่ 4.6 แสดงผลการทดลองการทำนายผลของไฟล์เสียง ซึ่งจะแสดงค่าเปอร์เซ็นต์ความตรงกันของไฟล์เสียงที่นำเข้าทดลองเทียบกับฟีเจอร์ของเสียงไอปกติและเสียงไอโควิด และมีค่าความผิดพลาดเกิดขึ้นเมื่อเสียงที่นำเข้าทดสอบไม่ใช่เสียงไอ ซึ่งความแม่นยำในการทำนายผลที่ไม่ใช่เสียงไอมีค่า 100 เปอร์เซ็นต์ ค่าความแม่นยำการทำนายผลเสียงไอคนที่ปกติ 77.78 เปอร์เซ็นต์ และค่าความแม่นยำเสียงไอคนที่ปกติอยู่ที่ 25 เปอร์เซ็นต์

บทที่ 5

บทสรุปและข้อเสนอแนะ

5.1 สรุปผลการทดลอง

5.1.1 สรุปผลการทดลองแอปพลิเคชันบนคอมพิวเตอร์ระบบปฏิบัติการวินโดว์ 11

จากการทดลองสรุปได้ว่าเมื่อใช้งานแอปพลิเคชันบนคอมพิวเตอร์ระบบปฏิบัติการวินโดว์ 11 แอปพลิเคชันสามารถทำงานบนคอมพิวเตอร์ระบบปฏิบัติการวินโดว์ได้ สามารถประมวลผลเสียงที่รับเข้าผ่านอุปกรณ์รับเสียงได้ และสามารถแสดงผลการทำนายบนหน้าต่างแอปพลิเคชันได้

5.1.2 สรุปผลการทดลองหาความแม่นยำของพีจเจอร์จากการเทรน

จากการทดลองสรุปได้ว่าเมื่อเพิ่มชุดข้อมูลในการเทรนข้อมูลสร้างพีจเจอร์แล้วนำพีจเจอร์มาหาค่าความแม่นยำ ชุดข้อมูลที่เพิ่มขึ้นส่งผลต่อความแม่นยำของการทำนายผล หากข้อมูลเสียงที่นำมาเทรนมีขนาดมากกว่า 1 วินาที ส่งผลให้ค่าความแม่นยำในการทำนายผลลดลงอย่างมาก

5.1.3 สรุปผลการทดลองเสียงที่มีลักษณะการโอต่างกัน

จากการทดลองจะสรุปได้ว่าค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลขอบเขตบนและขอบเขตล่างมีค่าต่างกันตามลักษณะการโอ ค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลขอบเขตบนในลำดับที่ 2 มีขีดจำกัดสูงสุดและขีดจำกัดต่ำสุดมากกว่าค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลขอบเขตบนในลำดับที่ 1 และค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลจามีขอบเขตและขีดจำกัดที่เปลี่ยนไปขึ้นอยู่กับจำนวนเสียงที่เข้าสู่กระบวนการเทรน

5.1.4 สรุปผลการทดลองเสียงโอที่มีจำนวนลูกคลื่นต่างกัน

จากการทดลองสรุปได้ว่าในขอบเขตบนค่าสัมประสิทธิ์บนสเกลเมลตำแหน่งที่ 4 มีค่ามากขึ้นตามจำนวนลูกคลื่นที่เพิ่มขึ้น และค่าสัมประสิทธิ์บนสเกลเมลตำแหน่งที่ 8 และ 12 มีค่าน้อยลงตามจำนวนลูกคลื่นที่เพิ่มขึ้น ในขอบเขตล่างค่าสัมประสิทธิ์บนสเกลเมลตำแหน่งที่ 1, 4 และ 5 มีค่ามากขึ้นตามจำนวนลูกคลื่นที่เพิ่มขึ้น ค่าสัมประสิทธิ์บนสเกลเมลตำแหน่งที่ 8 และ 9 มีค่าน้อยลงตามจำนวนลูกคลื่นที่เพิ่มขึ้น และค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลจามีขอบเขตและขีดจำกัดที่เปลี่ยนไปขึ้นอยู่กับจำนวนลูกคลื่นเสียงที่เข้าสู่กระบวนการเทรน

5.1.5 สรุปผลการทดลองเสียงโอแบบมีตัวกรองเสียงและไม่มีตัวกรองเสียง

จากการทดลองแสดงค่าสัมประสิทธิ์บนสเกลเมลของเสียงโอคนเดียวขนาดไฟล์หนึ่งวินาทีจำนวนหนึ่งลูกคลื่นแบบมีและไม่มีตัวกรองเสียงในอุปกรณ์รับเสียง (sound filter) มีการแสดงค่า

สัมประสิทธิ์เซปสตรัมบนสเกลเมล ในรูปแบบของค่าอาเรย์ ซึ่งค่าสัมประสิทธิ์บนสเกลเมลแบบที่มีตัวกรองเสียงจะมีค่าสูงกว่าค่าสัมประสิทธิ์บนสเกลเมลแบบที่ไม่มีตัวกรองเสียงทั้งขอบเขตบนและขอบเขตล่าง ค่าสัมประสิทธิ์เซปสตรัมบนสเกลเมลจำมีขอบเขตและขีดจำกัดที่เปลี่ยนไปขึ้นอยู่กับเสียงที่มีตัวกรองและไม่มีตัวกรองที่ผ่านที่เข้าสู่กระบวนการสร้างฟิจเจอร์ และในการทำนายผลบนแอปพลิเคชันการใช้ตัวกรองหรือไม่มีตัวกรองมีผลต่อการทำนายผลบนหน้าต่างแอปพลิเคชัน

5.1.6 สรุปผลการทดลองทำนายผลของแอปพลิเคชันด้วยไฟล์เสียง

จากการทดลองเมื่อใช้ไฟล์เสียงที่ไม่ใช่เสียงไอเข้าสู่กระบวนการทำนายผล ให้ค่าความแม่นยำการทำนายที่ 100 เปอร์เซ็น เมื่อใช้ไฟล์เสียงไอปกติค่าความแม่นยำการทำนายผลอยู่ที่ 77.78 เปอร์เซ็น เมื่อใช้ไฟล์เสียงโควิดค่าความแม่นยำการทำนายผลอยู่ที่ 25 เปอร์เซ็น ทำให้ค่าความแม่นยำในการทำนายผลของแอปพลิเคชันเท่ากับ 51.93 เปอร์เซ็น

5.2 ปัญหาและอุปสรรค

5.2.1 ความแม่นยำของฟิจเจอร์

จากที่แอปพลิเคชันช่วยประเมินโรคจากอาการไอมีความแม่นยำในการประเมินโรคยังไม่มากพอควรเนื่องจากจำนวนชุดข้อมูลที่ใช้ในการเทรนมีจำนวนน้อยจากอุปสรรคในการหาเสียงไอโควิด และปัญหาการทดลองที่ต้องใช้ผลการทดลองจำนวนมาก เช่น การบันทึกเสียงขณะเปิดใช้งานทำได้ยาก จากการที่ต้องประเมินผลคนที่มีการไอโควิด ขั้นตอนนี้มีความเสี่ยงสูงจึงไม่สามารถทดลองได้ ทำให้การพัฒนาฟิจเจอร์ทำได้ยาก เพราะแต่ละเสียงจะมีลักษณะแตกต่างกัน มีจุดที่ชุดข้อมูลมีค่าแตกต่างกันซึ่งจำเป็นต่อการนำไปพัฒนาอัลกอริทึมในการสร้างฟิจเจอร์

5.2.2 ไฟล์เสียงที่ใช้ในการสร้างฟิจเจอร์

ในการสร้างฟิจเจอร์ต้องใช้ไฟล์เสียงที่มีคุณภาพ ไม่มีสัญญาณรบกวนในตัวไฟล์ และต้องใช้ไฟล์เสียงจำนวนมากเพื่อสร้างขอบเขตการประมวลผลที่ชัดเจนแม่นยำ ด้วยช่องทางการหาไฟล์ตัวอย่างเสียงไอโรคและไฟล์เสียงไอปกติมีน้อยเนื่องจากปัญหาโรคติดต่อในปัจจุบัน ความร่วมมือของผู้ให้เสียงไอและอุปกรณ์ในการบันทึกเสียงที่มีคุณภาพ

5.3 แนวทางแก้ไข

5.3.1 ความแม่นยำของฟิจเจอร์

ในการเพิ่มความแม่นยำของฟิจเจอร์ทำได้ด้วยการเพิ่มจำนวนชุดข้อมูลในการเทรนจากการหาไฟล์เสียงไอโควิด และเพิ่มการทดลองการใช้งานแอปพลิเคชันเพื่อหาความผิดพลาดในการ

ประมวลผลแต่ละขั้นตอนนี้แล้วแก้ไขปรับปรุงชุดการประมวลผล รวมถึงต้องคำนึงคุณภาพถึงชุดข้อมูลที่ใช้ในการสร้างพีเจเจอร์

5.3.2 ไฟล์เสียงที่ใช้ในการสร้างพีเจเจอร์

หาแหล่งไฟล์เสียงจำนวนมาก ขอความร่วมมือจากผู้ที่ต้องการให้เสียงไอ หลีกเลียงการสัมผัสผู้ป่วย และใช้อุปกรณ์ในการบันทึกเสียงที่มีคุณภาพ



เอกสารอ้างอิง

- [1] ดร.ศุภวุฒิ สายเชื้อ, **Beating COVID-19 เอาชนะโรคภัยด้วยความเข้าใจ**, openbooks, 2565
- [2] ดร. ณัฐโชติ พรหมฤทธิ์, ดร. สัจจาภรณ์ ไวจรรยา, **Fundamental of Deep Learning in Practice**, ไอทีซี, พรีเมียร์, 2564
- [3] ผศ.ดร. กอบเกียรติ สระอุบล, **เรียนรู้ AI: Deep Learning ด้วย ไพทอน**, อินเตอร์มีเดีย, 2565
- [4] Predro Domingos (โดมิงโกส เพโดร), **สุดยอดอัลกอริทึมพลิกโฉมโลก**, ซีเอ็ดดูเคชั่น, 2564
- [5] ศุภชัย สมพานิช, **คู่มือเขียนโปรแกรมภาษา Visual Basic 2017**, สวิสดี ไอที, 2562
- [6] Sarayut Nonสิริ, PhD. Computer Scientist, **ภาษาโปรแกรม ไพทอน คืออะไร**, [ระบบออนไลน์], แหล่งที่มา <https://www.9expertฝึกอบรมing.com/articles/ไพทอน-คืออะไร>, เข้าถึงครั้งสุดท้าย (7 ก.ย. 2564)
- [7] select2web, 2556, **ทำความรู้จักกับ เจสัน คืออะไร**, [ระบบออนไลน์], แหล่งที่มา <https://www.boxsingle.com/13/ทำความรู้จักกับ-เจสัน-คืออะไร>, เข้าถึงครั้งสุดท้าย (7 ก.ย. 2564)
- [8] Sirasit Boonklang, 2564, **มาทำความรู้จักกับ ลิโบริซ่า**, [ระบบออนไลน์], แหล่งที่มา <https://www.borntodev.com/2021/09/17/มาทำความรู้จักกับ-ลิโบริซ่า/>, เข้าถึงครั้งสุดท้าย (12 ก.ย. 2564)
- [9] Developer Team, 2563, **พื้นฐานการใช้ Numpy ใน ไพทอน 3**, [ระบบออนไลน์], แหล่งที่มา <https://www.borntodev.com/2020/04/16/พื้นฐานการใช้-นัมพาย-ใน-ไพทอน-3/>, เข้าถึงครั้งสุดท้าย (12 ก.ย. 2564)
- [10] wannaphong, 2558, **ฟังก์ชัน แมทซ์ ใน ไพทอน**, [ระบบออนไลน์], แหล่งที่มา <https://www.ไพทอน3.wannaphong.com/2015/04/แมทซ์-ไพทอน.html>, เข้าถึงครั้งสุดท้าย (12 ก.ย. 2564)
- [11] Insipid Theme, 2565, **ไพทอน-อุปกรณ์รับเสียง**, [ระบบออนไลน์], แหล่งที่มา <https://www.ไพทอน-อุปกรณ์รับเสียง.readthedocs.io/en/latest/usage.html>, เข้าถึงครั้งสุดท้าย (27 ส.ค. 2564)

เอกสารอ้างอิง(ต่อ)

- [12] Bastian Bechtold, 2558, **ไฟล์เสียง**, [ระบบออนไลน์], แหล่งที่มา <https://www.pyไฟล์เสียง.readthedocs.io/en/latest/>, เข้าถึงครั้งสุดท้าย (27 ส.ค. 2564)
- [13] stackไพทอน, 2563, **สร้าง GUI ในภาษา ไพทอน โดยใช้ กี้วี่**, [ระบบออนไลน์], แหล่งที่มา <https://www.stackไพทอน.medium.com/สร้าง-gui-ในภาษา-ไพทอน-โดยใช้-กี้วี่-ff2554a6d05a>, เข้าถึงครั้งสุดท้าย (27 ส.ค. 2564)
- [14] ED SPERLING, 2561, **Deep Learning Spreads**, [ระบบออนไลน์], แหล่งที่มา <https://www.semiengineering.com/deep-learning-spreads/>, เข้าถึงครั้งสุดท้าย (27 ส.ค. 2564)
- [15] Sorapong Somsorn, 2564, **Audio Classification: การจำแนกเสียงด้วย โครงข่ายประสาทแบบ คอนโวลูชัน (CNN)**, [ระบบออนไลน์], แหล่งที่มา <https://www.sorapong.medium.com/audio-classification-การจำแนกเสียงด้วย-โครงข่ายประสาทแบบ-คอนโวลูชัน-cnn-2605b2d7ef2e>, เข้าถึงครั้งสุดท้าย (28 ส.ค. 2564)
- [16] Sivadee, 2561, **MACHINE LEARNING – นิยามและตัวอย่างการใช้งาน**, [ระบบออนไลน์], แหล่งที่มา <https://www.toolmakers.co/machine-learning-นิยามและตัวอย่างการใช้/>, เข้าถึงครั้งสุดท้าย (1 ก.ย. 2564)
- [17] mindพีเอ็ดพี, 2560, **รู้จักกับ วิชชนสตูดิโอโค้ด (วิชวล สตูดิโอ โค้ด) โปรแกรมฟรีจากค่าย ไมโครซอฟท์**, [ระบบออนไลน์], แหล่งที่มา <https://www.mindพีเอ็ดพี.com/บทความ/ไมโครซอฟท์/4829-visual-studio-code.html>, เข้าถึงครั้งสุดท้าย (6 ก.ย. 2564)



ภาคผนวก ก
โปรแกรมการทำงาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรมการทำงานของการสร้างชุดข้อมูล

```
# เรียกใช้ไลบรารี
import json
import os
import math
import librosa

# กำหนดชื่อโพลเดอร์ที่ต้องการเรียกอ่าน
DATASET_PATH = "COUGH"
# กำหนดชื่อไฟล์ที่ต้องการบันทึก
JSON_PATH = "datasets.json"

# กำหนดค่าตัวแปรที่จำเป็น
SAMPLE_RATE = 22050
TRACK_DURATION = 1
num_segments=1
SAMPLES_PER_TRACK = SAMPLE_RATE * TRACK_DURATION

# รับคำสั่งบันทึกไฟล์ด้วยการอ่านไฟล์เสียงในโพลเดอร์แล้วบันทึกเป็นไฟล์เจสัน
def save_mfcc(dataset_path, json_path, num_mfcc=13, n_fft=2048, hop_length=512):
    data = {"mapping": [], "labels": [], "mfcc": [] }
    samples_per_segment = int(SAMPLES_PER_TRACK / num_segments)
    num_mfcc_vectors_per_segment = math.ceil(samples_per_segment / hop_length)

# เปิดโพลเดอร์เสียงโควิดและโพลเดอร์เสียงปกติในโพลเดอร์เสียงหลัก
for i, (dirpath, dirnames, filenames) in enumerate(os.walk(dataset_path)):
    if dirpath is not dataset_path:
```

```

semantic_label = dirpath.split("/")[-1]
data["mapping"].append(semantic_label)
print("\nProcessing: {}".format(semantic_label))

```

เรียกอ่านไฟล์ในโฟลเดอร์แต่ละไฟล์

```
for f in filenames:
```

```
    file_path = os.path.join(dirpath, f)
```

```
    signal, sample_rate = librosa.load(file_path, sr=SAMPLE_RATE)
```

```
    for d in range(num_segments):
```

```
        start = samples_per_segment * d
```

```
        finish = start + samples_per_segment
```

```
        if finish == start + samples_per_segment:
```

```
            mfcc = librosa.feature.mfcc(signal[start:finish], sample_rate,
n_mfcc=num_mfcc, n_fft=n_fft, hop_length=hop_length)
```

```
            mfcc = mfcc.T
```

```
            if len(mfcc) == num_mfcc_vectors_per_segment:
```

```
                data["mfcc"].append(mfcc.tolist())
```

```
                data["labels"].append(i-1)
```

```
                print("{} segment:{}".format(file_path, d+1))
```

บันทึกชุดข้อมูลเป็นไฟล์เจสัน

```
with open(json_path, "w") as fp:
```

```
    json.dump(data, fp, indent=4)
```

การทำงานหลัก

```
if __name__ == "__main__":
```

```
    save_mfcc(DATASET_PATH, JSON_PATH)
```

โปรแกรมการทำงานของการสร้างฟิจเจอร์

```
# เรียกใช้ไลบรารี
import json

import numpy as np

import matplotlib.pyplot as plt

# กำหนดชื่อไฟล์ที่ต้องการเรียกอ่าน
DATA_PATH = "datasets.json"

# กำหนดชื่อไฟล์ที่ต้องการบันทึก
JSON_PATH = "feature.json"

# โหลดไฟล์ที่ต้องการอ่าน
def load_data(data_path):
    if data_path == DATA_PATH :
        with open(data_path, "r") as fp:
            data = json.load(fp)
            X = np.array(data["mfcc"])
            y = np.array(data["labels"])
            cough_type_data = data["labels"]
            return X, y, cough_type_data

# นำชุดข้อมูลเสียงสร้างขอบเขตบน
def compare_data_high(X):
    feature = []
    segments = X.shape[0]
    for k in range(segments):
        feature.append(k)
```

```

mfcc_num = []
mfcc_va = 13
for i in range(mfcc_va):
    c = -20000
    mfcc_compare_in_segment = X.shape[1]
    for j in range(mfcc_compare_in_segment):
        a = X[k][j][i]
        if a >= c :
            c = a
    mfcc_num.append(c)
feature[k] = mfcc_num
feature1 = feature
return feature1

```

นำชุดข้อมูลเสียงสร้างขอบเขตล่าง

```

def compare_data_low(X):
    feature = []
    segments = X.shape[0]
    for k in range(segments):
        feature.append(k)
        mfcc_num = []
        mfcc_va = 13
        for i in range(mfcc_va):
            c = 20000
            mfcc_compare_in_segment = 44
            for j in range(mfcc_compare_in_segment):
                a = X[k][j][i]
                if a <= c :

```

```

c = a

mfcc_num.append(c)

feature[k] = mfcc_num

feature2 = feature

return feature2

```

นำชุดขอบเขตบนที่ได้ทั้งหมดสร้างขีดจำกัดสูงสุดของขอบเขตบน

```
def cnn_data_max_top(cnn_data1, y):
```

```
    feature_top_max = []
```

```
    y_num_file = y.shape[0]
```

```
    n = -1
```

```
    for d in range(y_num_file):
```

```
        if y[d] != n:
```

```
            n = y[d]
```

```
            feature_top_max.append(n)
```

```
            mfcc_num = []
```

```
            cnn_mfcc_num = 13
```

```
            for j in range(cnn_mfcc_num):
```

```
                mfcc_num.append(j)
```

```
                a = -20000
```

```
                file_num = y.shape[0]
```

```
                for k in range(file_num):
```

```
                    x = y[k]
```

```
                    b = cnn_data1[k][j]
```

```
                    if x == n:
```

```
                        if a <= b:
```

```
                            a = b
```

```
                mfcc_num[j] = a

```

```

        feature_top_max[n] = mfcc_num
    return feature_top_max

```

นำชุดขอบเขตบนที่ได้ทั้งหมดสร้างขีดจำกัดต่ำสุดของขอบเขตบน

```
def cnn_data_max_bottom(cnn_data1, y):
```

```
    feature_top_min = []
```

```
    y_num_file = y.shape[0]
```

```
    n = -1
```

```
    for d in range(y_num_file):
```

```
        if y[d] != n:
```

```
            n = y[d]
```

```
            feature_top_min.append(n)
```

```
            mfcc_num = []
```

```
            cnn_mfcc_num = 13
```

```
            for j in range(cnn_mfcc_num):
```

```
                mfcc_num.append(j)
```

```
                a = 20000
```

```
                file_num = y.shape[0]
```

```
                for k in range(file_num):
```

```
                    x = y[k]
```

```
                    b = cnn_data1[k][j]
```

```
                    if x == n:
```

```
                        if a >= b:
```

```
                            a = b
```

```
                mfcc_num[j] = a
```

```
            feature_top_min[n] = mfcc_num
```

```
    return feature_top_min
```

นำชุดขอบเขตล่างที่ได้ทั้งหมดสร้างขีดจำกัดสูงสุดของขอบเขตล่าง

```
def cnn_data_min_top(cnn_data2, y):
    feature_bottom_max = []
    y_num_file = y.shape[0]
    n = -1
    for d in range(y_num_file):
        if y[d] != n:
            n = y[d]
            feature_bottom_max.append(n)
            mfcc_num = []
            cnn_mfcc_num = 13
            for j in range(cnn_mfcc_num):
                mfcc_num.append(j)
            a = -20000
            file_num = y.shape[0]
            for k in range(file_num):
                x = y[k]
                b = cnn_data2[k][j]
                if x == n:
                    if a <= b:
                        a = b
            mfcc_num[j] = a
            feature_bottom_max[n] = mfcc_num
    return feature_bottom_max
```

นำชุดขอบเขตล่างที่ได้ทั้งหมดสร้างขีดจำกัดต่ำสุดของขอบเขตล่าง

```
def cnn_data_min_bottom(cnn_data2, y):
    feature_bottom_min = []
```

```

y_num_file = y.shape[0]
n = -1
for d in range(y_num_file):
    if y[d] != n:
        n = y[d]
        feature_bottom_min.append(n)
        mfcc_num = []
        cnn_mfcc_num = 13
        for j in range(cnn_mfcc_num):
            mfcc_num.append(j)
            a = 20000
            file_num = y.shape[0]
            for k in range(file_num):
                x = y[k]
                b = cnn_data2[k][j]
                if x == n:
                    if a >= b:
                        a = b
            mfcc_num[j] = a
        feature_bottom_min[n] = mfcc_num
return feature_bottom_min

```

```

# หาค่าความผิดพลาด ค่าความตรงกันของไฟล์เสียง และค่าความแม่นยำของฟิจเจอร์
def find_accurry(feature_top_max, feature_top_min, feature_bottom_max,
feature_bottom_min, cnn_data1, cnn_data2, y):
    point_for_covid = 0
    point_for_normal = 0
    point_for_error = 0

```

```

point_for_all_covid = 0
point_for_all_normal = 0

for k in range(y.shape[0]):
    covid = 0
    normal = 0
    all_covid = 0
    all_normal = 0

    if y[k] == 0 :
        all_covid += 26
        point_for_all_covid += 26
    elif y[k] == 1 :
        all_normal += 26
        point_for_all_normal += 26

    for j in range(13):

        if cnn_data1[k][j] <= feature_top_max[0][j]:
            if cnn_data1[k][j] >= feature_top_min[0][j]:
                if cnn_data1[k][j] >= feature_bottom_max[0][j]:
                    covid += 1

        if cnn_data2[k][j] <= feature_bottom_max[0][j]:
            if cnn_data2[k][j] >= feature_bottom_min[0][j]:
                if cnn_data2[k][j] <= feature_top_min[0][j]:
                    covid += 1

        if cnn_data1[k][j] <= feature_top_max[1][j]:

```

```

if cnn_data1[k][j] >= feature_top_min[1][j]:
    if cnn_data2[k][j] <= feature_bottom_max[1][j]:
        normal += 1
if cnn_data2[k][j] <= feature_bottom_max[1][j]:
    if cnn_data2[k][j] >= feature_bottom_min[1][j]:
        if cnn_data2[k][j] <= feature_top_min[1][j]:
            normal += 1

if covid == 26 or normal == 26:
    if covid == normal:
        point_for_error += covid
    elif covid > normal:
        point_for_covid += covid
    elif covid < normal:
        point_for_normal += normal

accury_point_for_covid = (point_for_covid / point_for_all_covid)
print("accury of covid cough is {}".format(accury_point_for_covid))
accury_point_for_normal = (point_for_normal / point_for_all_normal)
print("accury of normal cough is {}".format(accury_point_for_normal))
total_point_for_error = (point_for_error / (point_for_all_covid + point_for_all_normal))
print("accury for unknown cough is {}".format(total_point_for_error))
accury_train_cough = ((accury_point_for_covid + accury_point_for_normal) / 2)

return accury_train_cough

```

บันทึกฟิจเจอร์

```
def save_feature_data(json_path, feature_top_max, feature_top_min,
feature_bottom_max, feature_bottom_min, cnn_data1, cnn_data2, cough_type_data):
```

```
    data = {
        "feature_top_max_data": []
        ,"feature_top_min_data": []
        ,"feature_bottom_max_data": []
        ,"feature_bottom_min_data": []
        ,"cnn_top_data": []
        ,"cnn_bottom_data": []
        ,"cough_type_data": []
    }
    data["feature_top_max_data"].append(feature_top_max)
    data["feature_top_min_data"].append(feature_top_min)
    data["feature_bottom_max_data"].append(feature_bottom_max)
    data["feature_bottom_min_data"].append(feature_bottom_min)

    data["cnn_top_data"].append(cnn_data1)
    data["cnn_bottom_data"].append(cnn_data2)
    data["cough_type_data"].append(cough_type_data)
```

บันทึก cnn_data เป็นไฟล์เจสัน

```
with open(json_path, "w") as fp:
    json.dump(data, fp, indent=4)
```

สร้างหน้าต่างกราฟ

```
def plot_feature(feature_top_max, feature_top_min, feature_bottom_max,
feature_bottom_min):
```

```

fig, axs = plt.subplots(2)

axs[0].plot(feature_top_max[0], label="top_max")
axs[0].plot(feature_top_min[0], label="top_min")
axs[0].plot(feature_bottom_max[0], label="bottom_max")
axs[0].plot(feature_bottom_min[0], label="bottom_min")

axs[0].set_ylabel("value")
axs[0].legend(loc="upper right")
axs[0].set_title("covid")

axs[1].plot(feature_top_max[1], label="top_max")
axs[1].plot(feature_top_min[1], label="top_min")
axs[1].plot(feature_bottom_max[1], label="bottom_max")
axs[1].plot(feature_bottom_min[1], label="bottom_min")

axs[1].set_ylabel("value")
axs[1].set_xlabel("mfcc")
axs[1].legend(loc="upper right")
axs[1].set_title("normal")

plt.show()

# ชุดคำสั่งหลัก
if __name__ == "__main__":
    X, y, cough_type_data = load_data(DATA_PATH)

```

```

cnn_data1 = compare_data_high(X)
cnn_data2 = compare_data_low(X)

feature_top_max = cnn_data_max_top(cnn_data1, y)
feature_top_min = cnn_data_max_bottom(cnn_data1, y)
feature_bottom_max = cnn_data_min_top(cnn_data2, y)
feature_bottom_min = cnn_data_min_bottom(cnn_data2, y)

accurry = find_accurry(feature_top_max, feature_top_min, feature_bottom_max,
feature_bottom_min, cnn_data1, cnn_data2, y)
print("accurry is {}".format(accurry))

save_feature_data(JSON_PATH, feature_top_max, feature_top_min,
feature_bottom_max, feature_bottom_min, cnn_data1, cnn_data2, cough_type_data)

plot_feature(feature_top_max, feature_top_min, feature_bottom_max,
feature_bottom_min)

```

โปรแกรมการทำงานของแอปพลิเคชัน

```

# เรียกใช้ไลบรารี
import json
import kivy
from kivy.app import App
from kivy.ui.boxlayout import BoxLayout
import librosa
import numpy as np
import math

```

```

import matplotlib.pyplot as plt

import sounddevice

import soundfile as sf

# กำหนดเวอร์ชันของไคววี่
kivy.require('2.1.0')

# กำหนดชื่อไฟล์ที่ต้องการอ่าน
DATA_FEATURE = "feature.json"
UNKNOWN_COUGH_PATH = "unknown_cough.wav"

# กำหนดค่าระยะเวลาบันทึกเสียง
duration = 3
num_segments = (duration * 2) - 1

# การโหลดหน้าต่างแอปพลิเคชันและคำสั่งหลักการประมวลผล
class MyRoot(BoxLayout):

    def __init__(self):
        super(MyRoot, self).__init__()

    def start_record(self):
        self.state_label.text = str("recording")

    def start_predict_on_app(self):

        recording_cough()

```

```

feature_unknown_cough_high, feature_unknown_cough_low =
load_unknow_cough(UNKNOWN_COUGH_PATH)

accurry_covid, accurry_normal, error = classifier_unknown_cough(W, X, Y ,Z
,top_data ,botton_data
,type_data, feature_unknown_cough_high, feature_unknown_cough_low)

predict_cough = predict(accurry_covid, accurry_normal, error)
if predict_cough == 0:
    self.state_label.text = str("You are infected covid-19")
elif predict_cough == 1:
    self.state_label.text = str("You are normal")
elif predict_cough == 5:
    self.state_label.text = str("this is not cough")

plot_predict(W, X, Y, Z, feature_unknown_cough_high,
feature_unknown_cough_low)

# สร้างหน้าต่างแอปพลิเคชัน
class AHADC(App):

    def build(self):
        return MyRoot()

# คำสั่งบันทึกเสียง
def recording_cough():
    fps = 22050
    print("Recording...")

```

```

recording_unknow_cough = sounddevice.rec(int(duration*fps), samplerate=fps,
channels=2)

sounddevice.wait()

print("Done!")

sf.write("unknow_cough.wav",recording_unknow_cough,fps)

```

คำสั่งอ่านข้อมูลฟีเจอร์

```
def load_data(data_path):
```

```
    if data_path == DATA_FEATURE :
```

```
        with open(data_path, "r") as fp:
```

```
            data = json.load(fp)
```

```
            W = np.array(data["feature_top_max_data"])
```

```
            X = np.array(data["feature_top_min_data"])
```

```
            Y = np.array(data["feature_bottom_max_data"])
```

```
            Z = np.array(data["feature_bottom_min_data"])
```

```
            top_data = np.array(data["cnn_top_data"])
```

```
            botton_data = np.array(data["cnn_bottom_data"])
```

```
            type_data = np.array(data["cough_type_data"])
```

```
            return W, X, Y,Z ,top_data ,botton_data ,type_data
```

คำสั่งอ่านข้อมูลไฟล์เสียง

```
def load_unknow_cough(data_path, num_mfcc=13, n_fft=2048, hop_length=512):
```

```
    data = {
```

```
        "mfcc": []
```

```
    }
```

```

SAMPLE_RATE = 22050

num_mfcc_vectors_per_segment = math.ceil(SAMPLE_RATE / hop_length)

signal, sample_rate = librosa.load(data_path, sr=SAMPLE_RATE)

for d in range(num_segments):

    samples_per_segment = SAMPLE_RATE / 2

    start = int(samples_per_segment * d)

    finish = start + SAMPLE_RATE

    if finish == start + SAMPLE_RATE:

        mfcc = librosa.feature.mfcc(signal[start:finish], sample_rate, n_mfcc=num_mfcc,
n_fft=n_fft, hop_length=hop_length)

        mfcc = mfcc.T

        if len(mfcc) == num_mfcc_vectors_per_segment:

            data["mfcc"].append(mfcc.tolist())

data = np.array(data["mfcc"])

feature_unknown_cough_high = compare_data_high(data)

feature_unknown_cough_high = np.array(feature_unknown_cough_high)

feature_unknown_cough_low = compare_data_low(data)

feature_unknown_cough_low = np.array(feature_unknown_cough_low)

return feature_unknown_cough_high, feature_unknown_cough_low

```

คำสั่งสร้างขอบเขตบนของไฟล์เสียงที่บันทึก

```
def compare_data_high(X):
```

```
    feature = []
```

```
    segments = X.shape[0]
```

```
    for k in range(segments):
```

```
        feature.append(k)
```

```
        mfcc_num = []
```

```
        mfcc_va = X.shape[2]
```

```

for i in range(mfcc_va):
    c = -20000
    mfcc_compare_in_segment = X.shape[1]
    for j in range(mfcc_compare_in_segment):
        a = X[k][j][i]
        if a >= c :
            c = a
    mfcc_num.append(c)
feature[k] = (mfcc_num)
feature_unknow_cough_high = feature
return feature_unknow_cough_high

```

คำสั่งสร้างขอบเขตล่างของไฟล์เสียงที่บันทึก

```

def compare_data_low(X):
    feature = []
    segments = X.shape[0]
    for k in range(segments):
        feature.append(k)
        mfcc_num = []
        mfcc_va = X.shape[2]
        for i in range(mfcc_va):
            c = 20000
            mfcc_compare_in_segment = X.shape[1]
            for j in range(mfcc_compare_in_segment):
                a = X[k][j][i]
                if a <= c :
                    c = a
            mfcc_num.append(c)

```

```

feature[k] = (mfcc_num)
feature_unknown_cough_low = feature
return feature_unknown_cough_low

```

คำสั่งวิเคราะห์ความตรงกันของเสียงที่รับเข้ากับฟิจเจอร์

```

def classifier_unknown_cough(W, X, Y,Z ,top_data ,bottom_data ,type_data,
feature_unknown_cough_high, feature_unknown_cough_low):

```

```

    accurry_covid = 0
    accurry_normal = 0
    cough_point = 0
    error = 100
    data = []
    for d in range(num_segments):
        data.append(d)
        point = []
        for k in range(W.shape[1]):
            if k == 0:
                point.append(k)
                point.append(k+1)
                point[k+1] = 0
            for j in range(13):
                if k == 0:
                    if feature_unknown_cough_high[d][j] <= W[0][k][j]:
                        if feature_unknown_cough_high[d][j] >= X[0][k][j]:
                            if feature_unknown_cough_high[d][j] >= Y[0][k][j]:
                                point[k+1] += 1
                    if feature_unknown_cough_low[d][j] <= X[0][k][j]:
                        if feature_unknown_cough_low[d][j] <= Y[0][k][j]:

```

```

        if feature_unknown_cough_low[d][j] >= Z[0][k][j]:
            point[k+1] += 1
    if feature_unknown_cough_high[d][j] > W[0][k][j]:
        if feature_unknown_cough_high[d][j] > W[0][k+1][j]:
            point[k] += 1
    if feature_unknown_cough_high[d][j] < X[0][k][j]:
        if feature_unknown_cough_high[d][j] < X[0][k+1][j]:
            point[k] += 1
    if feature_unknown_cough_low[d][j] > Y[0][k][j]:
        if feature_unknown_cough_low[d][j] > Y[0][k+1][j]:
            point[k] += 1
    if feature_unknown_cough_low[d][j] < Z[0][k][j]:
        if feature_unknown_cough_low[d][j] < Z[0][k+1][j]:
            point[k] += 1
    if k == 1:
        if feature_unknown_cough_high[d][j] <= W[0][k][j]:
            if feature_unknown_cough_high[d][j] >= X[0][k][j]:
                if feature_unknown_cough_high[d][j] >= Y[0][k][j]:
                    point[k+1] += 1
        if feature_unknown_cough_low[d][j] <= X[0][k][j]:
            if feature_unknown_cough_low[d][j] <= Y[0][k][j]:
                if feature_unknown_cough_low[d][j] >= Z[0][k][j]:
                    point[k+1] += 1

    data[d] = (point)
print(data)
for a in range(num_segments):
    if data[a][0] == 0:
        cough_point += 1

```

```

else :
    if data[a][0] < error:
        error = data[a][0]
if cough_point > 0:
    error = 0
for a in range(num_segments):
    if data[a][0] == 0:
        if (data[a][1] / 26) >= accurry_covid:
            if (data[a][2] / 26) >= accurry_normal:
                accurry_covid = data[a][1] / 26
                accurry_normal = data[a][2] / 26
            if data[a][1] == data[a][2]:
                if data[a][1] == 26:
                    if data[a][2] == 26:
                        cough_point = 0
                        type_of_data = -1
                        for k in range(type_data.shape):
                            cough = 0
                            for j in range(13):
                                if feature_unknown_cough_high[a][j] <= top_data[0][k][j]:
                                    cough += 1
                                if feature_unknown_cough_low[a][j] <= bottom_data[0][k][j]:
                                    cough += 1
                            if cough_point < cough:
                                cough_point == cough
                                type_of_data == type_data[k]
                        if type_of_data == 0:
                            accurry_covid = 1

```

```

        accurry_normal = 0
        return accurry_covid, accurry_normal, error
    elif type_of_data == 1:
        accurry_covid = 0
        accurry_normal = 1
        return accurry_covid, accurry_normal, error
    else : return accurry_covid, accurry_normal, error
return accurry_covid, accurry_normal, error

```

คำสั่งทำนายผล

```

def predict(accurry_covid, accurry_normal, error):
    x = error
    a = accurry_covid * 100
    b = accurry_normal * 100
    print(error)
    if x == 0 :
        if b >= 100:
            print("you are normal")
            print("accurry to normal cough {} percent".format(b))
            print("accurry to covid cough {} percent".format(a))
            return 1
        elif a >= 100 :
            print("you are infected")
            print("accurry to covid cough {} percent".format(a))
            print("accurry to normal cough {} percent".format(b))
            return 0
        else :
            print("this is not cough")

```

```

    print("accury to covid cough {} percent".format(a))
    print("accury to normal cough {} percent".format(b))
    print("error point = {}".format(x))
    return 5
else :
    print("this is not cough")
    print("accury to covid cough {} percent".format(a))
    print("accury to normal cough {} percent".format(b))
    print("error point = {}".format(x))
    return 5

```

คำสั่งสร้างกราฟ

```

def plot_predict(W, X, Y, Z, feature_unknown_cough_high,
feature_unknown_cough_low):
    fig, axs = plt.subplots(2)
    axs[0].plot(W[0][0], label="Covid-top_max")
    axs[0].plot(X[0][0], label="Covid-top_min")
    axs[0].plot(Y[0][0], label="Covid-bottom_max")
    axs[0].plot(Z[0][0], label="Covid-bottom_min")
    axs[0].plot(feature_unknown_cough_high[0], label="unknown-cough-HIGH")
    axs[0].plot(feature_unknown_cough_low[0], label="unknown-cough-LOW")
    axs[0].set_ylabel("MFCC value")
    axs[0].legend(loc="lower right")
    axs[0].set_title("covid test")
    axs[1].plot(W[0][1], label="Normal-top_max")
    axs[1].plot(X[0][1], label="Normal-top_min")
    axs[1].plot(Y[0][1], label="Normal-bottom_max")
    axs[1].plot(Z[0][1], label="Normal-bottom_min")

```

```

axs[1].plot(feature_unknown_cough_high[0], label="unknown-cough-HIGH")
axs[1].plot(feature_unknown_cough_low[0], label="unknown-cough-LOW")
axs[1].set_ylabel("MFCC value")
axs[1].set_xlabel("Mel-frequency cepstral coefficients : MFCCs")
axs[1].legend(loc="lower right")
axs[1].set_title("normal test")
plt.show()

```

คำสั่งการทำงานหลังของแอปพลิเคชัน

```

if __name__ == "__main__":
    W, X, Y, Z, top_data, bottom_data, type_data = load_data(DATA_FEATURE)
    AHADC().run()

```

โปรแกรมสร้างหน้าต่างแอปพลิเคชัน

<MyRoot>:

canvas.before:

Color:

rgba: 1, 1, 1, 1

Rectangle:

pos: self.pos

size: self.size

state_label: state_label

BoxLayout:

orientation: 'vertical'

Button:

background_color: 97/255, 208/255, 1, 1

color: 1, 1, 1, 1

font_size: 48

text: "Application Helps Assess Disease From Coughing"

Label:

Label:

Label:

color: 0.92, 0.45, 0, 1

font_size: 48

outline_color: 0,0,0

outline_width: 2

text: "press Start button then cough in 3 seconds"

Label:

Label:

Label:

color: 0.25, 1, 0.98, 1

font_size: 72

id: state_label

text: "ready to record"

Label:

Label:

BoxLayout:

orientation: 'horizontal'

Label:

Button:

background_color: 0, 126/255, 181/255, 1

color: 64/255, 1, 35/255, 1

font_size: 32

on_release: root.start_predict_on_app()

on_press: root.start_record()

size: 100, 50

text: "Start"

Label:

Label :

Label :



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ข
คู่มือการใช้งานแอปพลิเคชันช่วยประเมินโรคจากอาการไอ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



คู่มือใช้งานแอปพลิเคชันช่วยประเมินโรคจากอาการไอ

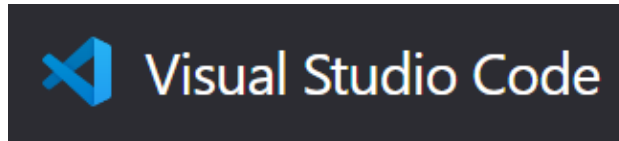
เกรียงศักดิ์ ศรีสุวรรณ
จอนแนชั่น ดอร์วิเนน

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรบัณฑิต
สาขาวิศวกรรมอิเล็กทรอนิกส์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
วิทยาเขตชุมพรเขตรอุดมศักดิ์ จังหวัดชุมพร
ปีการศึกษา 2564

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คู่มือการใช้งานแอปพลิเคชันช่วยประเมินโรคจากอาการไอ

1. ดาวนโหลดและติดตั้ง วิชชวณสตูดิโอโค้ด ลงบนคอมพิวเตอร์ระบบปฏิบัติการวินโดว 8 ,10 ,11



รูปที่ ข.1

2. ดาวนโหลดและติดตั้งภาษาไพทอนเวอร์ชัน 3.10.4 หรือสูงกว่า



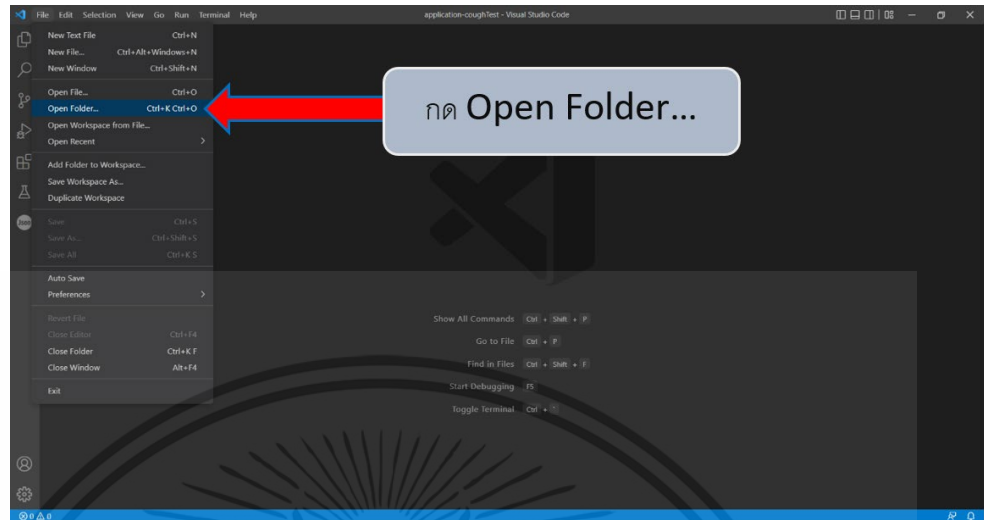
รูปที่ ข.2

3. ดาวนโหลดและติดตั้งไลบรารีที่จำเป็น

- 1) เจสัน
- 2) กวี
- 3) ลิโบรซ่า
- 4) นัมพาย
- 5) แมทซ์
- 6) Matplotlib.pyplot
- 7) อุปกรณ์รับเสียง
- 8) ไฟล์เสียง

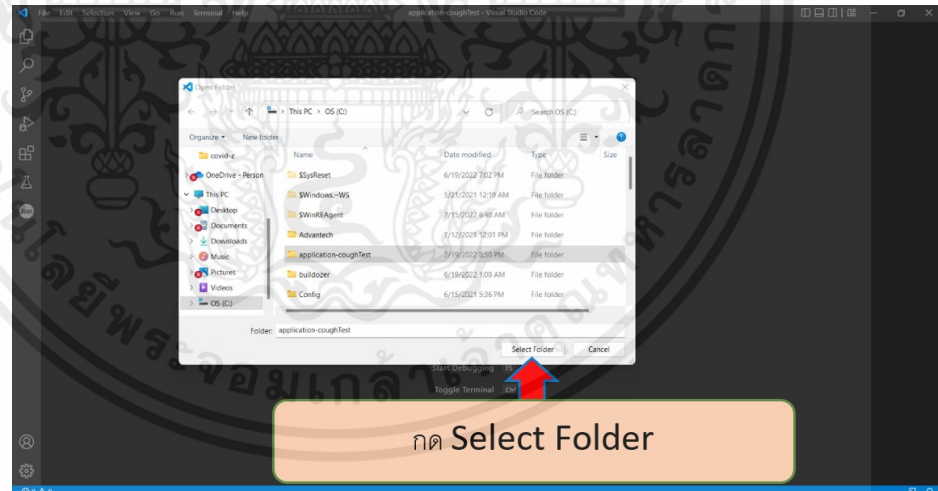
4. กดใช้งานแอปพลิเคชัน

4.1. กด Open Folder...



รูปที่ ข.3

4.2. เลือกโฟลเดอร์ที่มีไฟล์แอปพลิเคชัน แล้วกดปุ่ม Select Folder

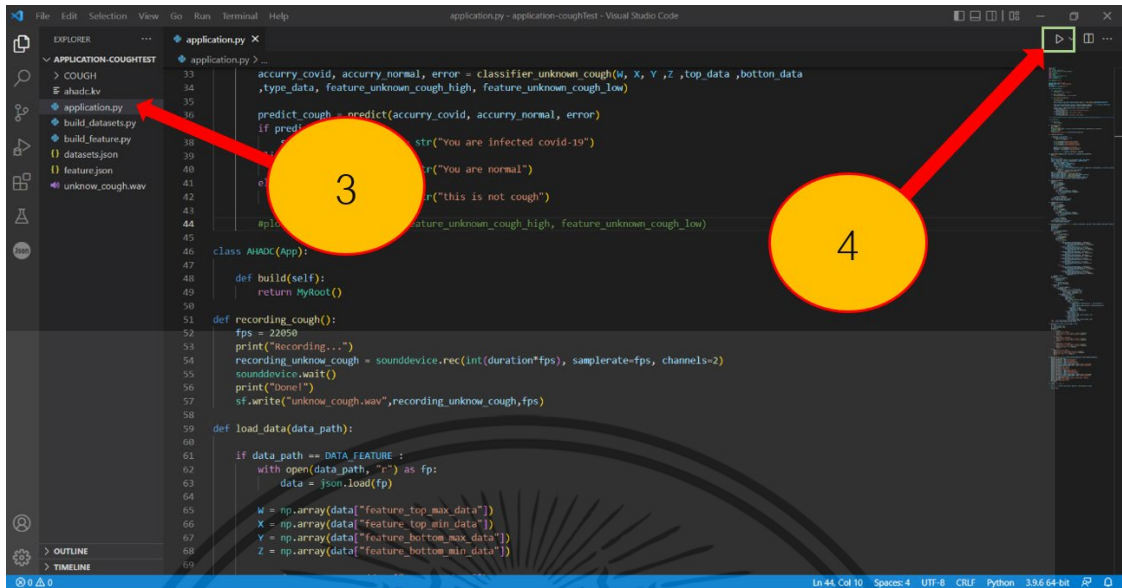


รูปที่ ข.4

4.3. เปิดไฟล์แอปพลิเคชัน

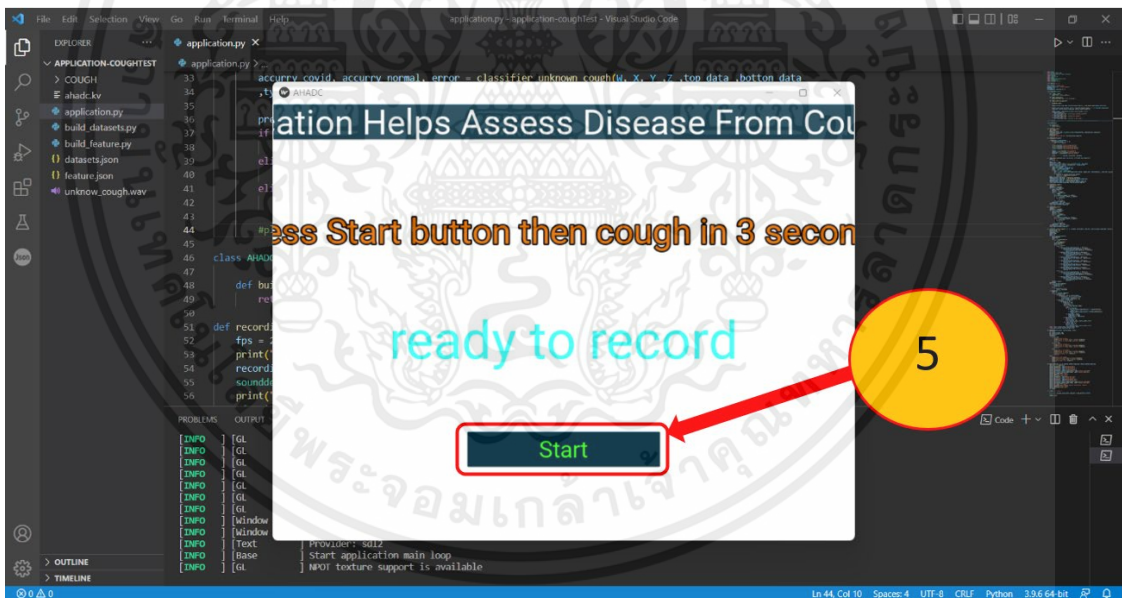
4.4. กดปุ่มเริ่มการทำงานแอปพลิเคชัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ ข.5

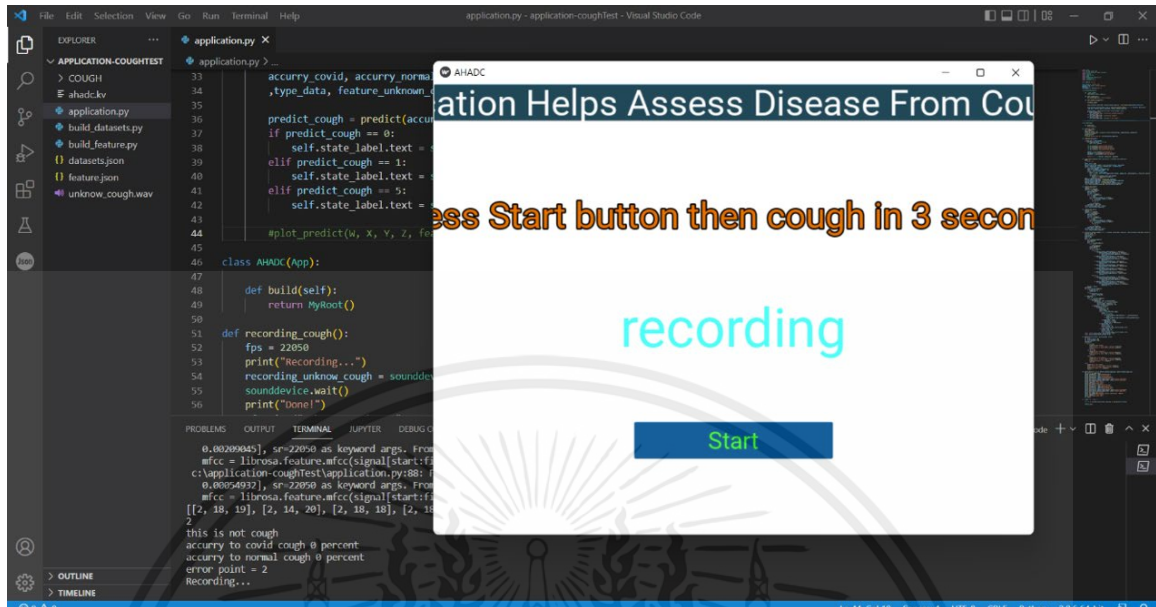
4.5 กดปุ่ม Start เพื่อเริ่มบันทึกเสียงเป็นเวลา 3 วินาที



รูปที่ ข.6

4.6 ระหว่างทำการบันทึกเสียงจะขึ้นว่า recording

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้คัดลอกเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

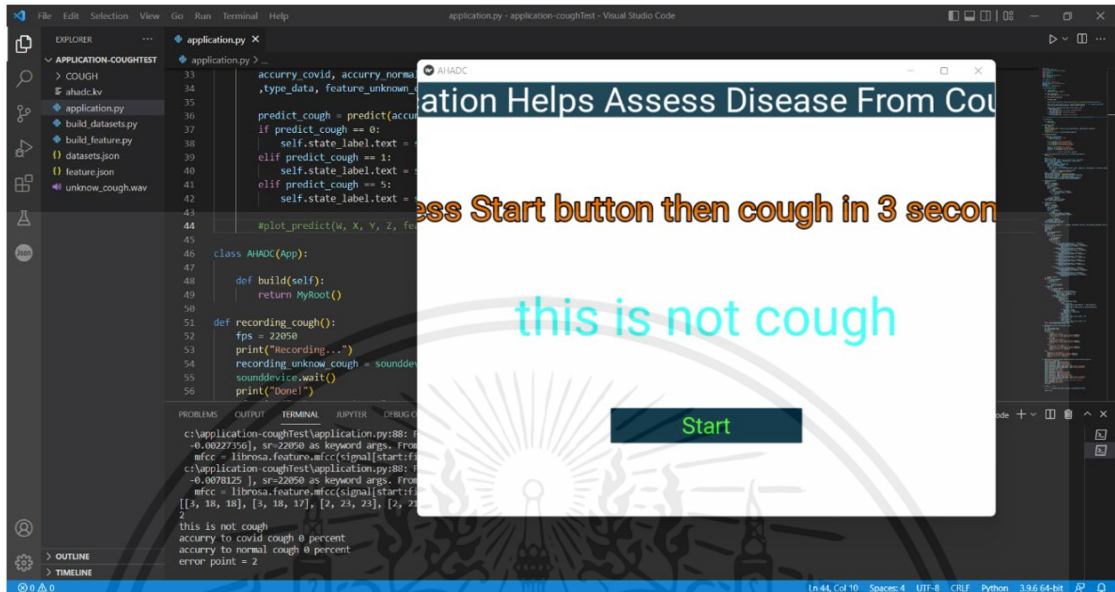


รูปที่ ข.7

4.7 แสดงผลเป็นอักษรหลังการบันทึกเสียงและประมวลผลในทันที โดยมี 3 แบบที่แสดงคือ

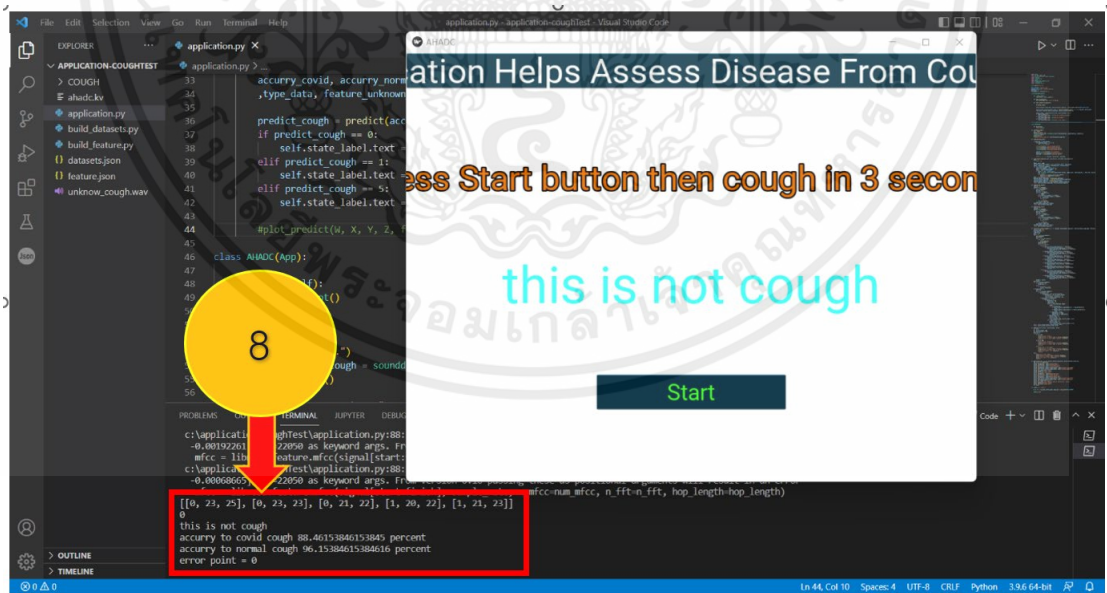
- 1) you are normal (คุณเป็นปกติ)
- 2) you are infected โรคโควิด-19 (คุณได้ติดเชื้อโควิด19)
- 3) this is not cough (นี่ไม่ใช่เสียงไอ)

4.8 การแสดงผล



รูปที่ ข.8

4.9 แสดงผลความเข้ากันของเสียงที่รับเข้ากับฟิจเจอร์



รูปที่ ข.9

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ประวัติผู้เขียน



ชื่อ-นามสกุล

นายเกรียงศักดิ์ ศรีสุวรรณ

วันเดือนปีเกิด

3 พฤษภาคม 2542

ที่อยู่

90 หมู่ 1 ต.ทุ่ง อ.ไชยา จ.สุราษฎร์ธานี 84110

ประวัติการศึกษา

พ.ศ.2560 จบการศึกษาระดับมัธยมศึกษาจาก
โรงเรียนไชยวิทยา

Tel.0937374493

Email mak4657@hotmail.com



ชื่อ-นามสกุล

นายจอนแนชั่น ดอร์วิน

วันเดือนปีเกิด

24 มกราคม 2543

ที่อยู่

82/1 หมู่ 4 ต.ปากคลอง อ.ปะทิว จ.ชุมพร 86210

ประวัติการศึกษา

พ.ศ.2560 จบการศึกษาระดับมัธยมศึกษาจาก
โรงเรียนศรีราษฎร์จังหวัดชุมพร

Tel.0952939240

Email jonathan09086@gmail.com