การเปรียบเทียบตัวแบบอนุกรมเวลาแบบผสมสำหรับการพยากรณ์
ข้อมูลอนุกรมเวลาที่มีปัจจัยเชิงฤดูกาล

นางสาวอนุธิดา อนันต์ทรัพย์สุข

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาสถิติ ภาควิชาสถิติ
คณะพาณิชยศาสตร์และการบัญชี จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2560

A COMPARATIVE STUDY OF HYBRID TIME SERIES MODELS FOR FORECASTING SEASONAL TIME SERIES

Miss Anutida Anansapsuk

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science Program in Statistics
Department of Statistics
Faculty of Commerce and Accountancy
Chulalongkorn University
Academic Year 2017

หัวข้อวิทยานิพนธ์	การเปรียบเทียบตัวแบบอนุกรมเวลาแบบผสมสำหรับการพยากรณ์ข้อมูลอนุกรมเวลาที่มีปัจจัยเชิงฤดูกาล
โดย	
สาขาวิชา	สถิติ
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	อาจารย์ ดร.นัท กุลวานิช

คณะพาณิชยศาสตร์และการบัญชี จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

.....คณบดีคณะพาณิชยศาสตร์และการบัญชี

(รองศาสตราจารย์ ดร.พสุ เดชะรินทร์)

คณะกรรมการสอบวิทยานิพนธ์

.....ประธานกรรมการ

(อาจารย์ ดร.อัครินทร์ ไพบูลย์พานิช)

.....อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

(อาจารย์ ดร.นัท กุลวานิช)

.....กรรมการ

(อาจารย์ ดร.ณัตติฤดี เจริญรักษ์)

.....กรรมการภายนอกมหาวิทยาลัย

(อาจารย์ ดร.อรุณี กำลัง)

อนุธิดา อนันต์ทรัพย์สุข : การเปรียบเทียบตัวแบบอนุกรมเวลาแบบผสมสำหรับการพยากรณ์ข้อมูลอนุกรมเวลาที่มีปัจจัยเชิงฤดูกาล (A COMPARATIVE STUDY OF HYBRID TIME SERIES MODELS FOR FORECASTING SEASONAL TIME SERIES) อ.ที่ปรึกษาวิทยานิพนธ์หลัก: อ. ดร.นัท กุลวานิช, 105 หน้า.

งานวิจัยนี้เป็นการศึกษาเปรียบเทียบความแม่นยำของค่าพยากรณ์ที่ได้จาก 3 ตัวแบบ คือ ตัวแบบ ARIMA ที่มีฤดูกาล(SARIMA), ตัวแบบผสมระหว่างตัวแบบ ARIMA ที่มีฤดูกาลกับตัวแบบโครงข่ายประสาทเทียม(SARIMA-ANN) และตัวแบบผสมระหว่างตัวแบบ ARIMA ที่มีฤดูกาลกับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน(SARIMA-SVM) โดยทำการศึกษาเปรียบเทียบทั้งในส่วนของข้อมูลจริงและข้อมูลจำลอง ในส่วนของข้อมูลจริงนั้นได้มีการนำราคาขายปลีกมะนาวเบอร์ 1-2 (หน่วยเป็นบาท/ผล) จากกรมการค้าภายใน กระทรวงพาณิชย์ ซึ่งเป็นราคาผลผลิตทางการเกษตรซึ่งอยู่ในรูปแบบอนุกรมเวลาที่มีปัจจัยเชิงฤดูกาลมาทำการเปรียบเทียบ โดยใช้เกณฑ์รากของค่าคลาดเคลื่อนกำลังสองเฉลี่ย(Root Mean Square Error : RMSE) เป็นเกณฑ์ในการเปรียบเทียบตัวแบบ ผลการศึกษาพบว่าตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบโครงข่ายประสาทเทียม(SARIMA-ANN) และตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน(SARIMA-SVM) ให้ผลการพยากรณ์ที่แม่นยำกว่าตัวแบบ SARIMA ทั้งในชุดข้อมูลจริง และชุดข้อมูลจำลอง และสำหรับการพยากรณ์ด้วยชุดข้อมูลจริงราคาขายปลีกมะนาวที่มีลักษณะอนุกรมเวลาที่มีปัจจัยเชิงฤดูกาลสอดคล้องกับตัวแบบ ARIMA$(1,1,2)x(0,1,1)_{12}$ ตัวแบบผสมระหว่าง ARIMA$(1,1,2)x(0,1,1)_{12}$ กับตัวแบบโครงข่ายประสาทเทียมให้ค่าพยากรณ์ที่แม่นยำที่สุด รองลงมาคือตัวแบบผสมระหว่าง ARIMA$(1,1,2)x(0,1,1)_{12}$ กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน และตัวแบบ ARIMA$(1,1,2)x(0,1,1)_{12}$ มีความแม่นยำในการพยากรณ์ต่ำที่สุด ซึ่งให้ผลสอดคล้องกับผลการพยากรณ์ด้วยชุดข้อมูลจำลอง

ภาควิชา สถิติ ลายมือชื่อนิสิต .....

สาขาวิชา สถิติ ลายมือชื่อ อ.ที่ปรึกษาหลัก .....

ปีการศึกษา 2560

5981552626 : MAJOR STATISTICS

KEYWORDS: SEASONAL ARIMA MODEL, ARTIFICIAL NEURAL NETWORK MODEL, SUPPORT VECTOR MACHINE MODEL, HYBRID MODEL

ANUTIDA ANANSAPSUK: A COMPARATIVE STUDY OF HYBRID TIME SERIES MODELS FOR FORECASTING SEASONAL TIME SERIES. ADVISOR: NAT KULVANICH, Ph.D., 105 pp.

This research is a comparative study of the prediction accuracy of three models : the seasonal ARIMA model(SARIMA), the hybrid model combining seasonal ARIMA and artificial neuron network model(SARIMA-ANN) and the hybrid model combining seasonal ARIMA and support vector machine model(SARIMA-SVM) using both real and simulated data. The retail prices of lime number 1-2 (in baht/unit) characterized by seasonal time series factor from the Department of Internal Trade of Thailand are used for real data. The Root Mean Square Error(RMSE) is introduced to compare the prediction accuracy among three models. The result of this study shows that hybrid model of SARIMA-ANN and SARIMA-SVM always outperform SARIMA model in both real and simulated data. For the real dataset using retail prices of lime number 1-2 characterized by seasonal time series factor ARIMA$(1,1,2)x(0,1,1)_{12}$, hybrid model combining ARIMA$(1,1,2)x(0,1,1)_{12}$ and ANN provides the most accurate forecast followed by hybrid model combining ARIMA$(1,1,2)x(0,1,1)_{12}$ and SVM and ARIMA$(1,1,2)x(0,1,1)_{12}$,respectively. The result is consistent with the forecasting in simulated data.

Department: Statistics Student's Signature .....

Field of Study: Statistics Advisor's Signature .....

Academic Year: 2017

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลงได้ด้วยความกรุณา และความอนุเคราะห์ช่วยเหลืออย่างดียิ่งจากอาจารย์ ดร.นัท กุลวานิช อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่ได้ให้โอกาสผู้วิจัยได้เป็นลูกศิษย์ในที่ปรึกษา คอยให้ความรู้ คำแนะนำ ตลอดจนชี้แนะแนวทางในการศึกษาแก่ผู้วิจัย อีกทั้งยังช่วยแก้ไขข้อบกพร่องต่างๆด้วยความเอาใจใส่ จนกระทั่งวิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยดี ผู้วิจัยขอกราบขอบพระคุณเป็นอย่างสูงมา ณ โอกาสนี้

ผู้วิจัยขอกราบขอบพระคุณ อาจารย์ ดร.อัครินทร์ ไพบูลย์พานิช ประธานกรรมการสอบวิทยานิพนธ์ อาจารย์ ดร.ณัตติฤดี เจริญรักษ์ กรรมการสอบวิทยานิพนธ์ และอาจารย์ ดร.อรุณี กำลัง กรรมการผู้ทรงคุณวุฒิภายนอก ที่ได้กรุณาสละเวลามาตรวจทานแก้ไขข้อบกพร่องในวิทยานิพนธ์ฉบับนี้ ตลอดจนให้คำแนะนำที่เป็นประโยชน์แก่ผู้วิจัยที่จะช่วยให้วิทยานิพนธ์ฉบับนี้สมบูรณ์ยิ่งขึ้น อีกทั้งคณาจารย์ประจำภาควิชาสถิติ คณะพาณิชยศาสตร์และการบัญชี จุฬาลงกรณ์มหาวิทยาลัยทุกท่านที่ได้ให้โอกาสทางการศึกษา และประสิทธิประสาทวิชาความรู้ทางด้านสถิติให้แก่ผู้วิจัยจนกระทั่งสำเร็จการศึกษาในระดับปริญญาโท

สุดท้ายนี้ผู้วิจัยขอกราบขอบพระคุณบิดา มารดา และครอบครัว ที่คอยสนับสนุนเป็นกำลังใจให้กับผู้วิจัยในการศึกษาในระดับปริญญาโท ตลอดจนการทำวิทยานิพนธ์ฉบับนี้จนสำเร็จลุล่วงไปได้ด้วยดี

คุณประโยชน์ และความดีใดๆอันพึงมีจากวิทยานิพนธ์ฉบับนี้ ผู้วิจัยขอมอบและอุทิศให้แก่บิดา มารดา ครอบครัว และคณาจารย์ประจำภาควิชาสถิติ คณะพาณิชยศาสตร์และการบัญชี จุฬาลงกรณ์มหาวิทยาลัยทุกท่าน ที่ได้ให้ความรู้ และวางรากฐานทางการศึกษาให้แก่ผู้วิจัยต่อไปในอนาคต

สารบัญ

หน้า

หน้า

หน้า

สารบัญภาพ

สารบัญตาราง

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

การพยากรณ์เป็นการคาดคะเนหรือประมาณเหตุการณ์ในอนาคตโดยอาศัยข้อมูลในอดีตหรือปัจจุบัน ตลอดจนวิจารณญาณ ความรู้ และประสบการณ์ของบุคคล เพื่อให้การตัดสินใจมีความถูกต้อง การพยากรณ์มีความสำคัญต่อการดำเนินงานในด้านต่างๆเป็นอย่างมากไม่ว่าจะเป็นด้านธุรกิจการเงิน ด้านการตลาด หรือด้านการผลิต เป็นต้น การพยากรณ์ทำให้ธุรกิจสามารถลดความสูญเสียที่จะเกิดขึ้น ทำให้ทราบถึงปริมาณความต้องการสินค้าของตลาด ทำให้สามารถผลิตหรือตระเตรียมสินค้าหรือวัตถุดิบได้เพียงพอ และสอดคล้องกับความต้องการในอนาคต ซึ่งผลที่ได้จากการพยากรณ์นั้นสามารถนำมาเป็นข้อมูลในการวางแผนกิจกรรมต่างๆให้มีประสิทธิภาพมากยิ่งขึ้น ดังนั้น การพัฒนาตัวแบบพยากรณ์ให้สามารถพยากรณ์ได้แม่นยำมากขึ้นจึงถือเป็นงานที่ท้าทาย ซึ่งมีงานวิจัยต่างๆมากมายเกิดขึ้นเพื่อพัฒนาในจุดนี้

การพยากรณ์ด้วยวิธีของ Box-Jenskins โดย George E.P. Box และ Gwilym M.Jenskins เป็นวิธีวิเคราะห์อนุกรมเวลาทางสถิติที่เกิดขึ้นในปี 1970 ซึ่งได้รับความนิยมจนถึงปัจจุบัน โดยมีการเลือกรูปแบบในการพยากรณ์อย่างเป็นระบบโดยใช้ค่าสัมประสิทธิ์สหสัมพันธ์ในตัวเอง และค่าสัมประสิทธิ์สหสัมพันธ์ในตัวเองบางส่วน โดยมีการทดสอบทางสถิติและการหาช่วงความเชื่อมั่นของค่าสังเกตในอนาคต ซึ่งก่อให้เกิด “ตัวแบบ ARIMA” หรือ Autoregressive Integrated Moving Average และตัวแบบอื่นๆ เช่นตัวแบบ SARIMA หรือ Seasonal Autoregressive Integrated Moving Average ตามมา ซึ่งเป็นตัวแบบที่ใช้สำหรับการพยากรณ์อนุกรมเวลาในส่วนที่เป็นฟังก์ชันเชิงเส้นตรงได้ดี

ในเวลาต่อมาได้มีการพัฒนาตัวแบบในการพยากรณ์ข้อมูลขั้นสูง เช่น ในปีค.ศ. 1991 ได้มีการนำเสนอ “ตัวแบบโครงข่ายประสาทเทียม(Artificial neuron network: ANN)” ซึ่งเป็นตัวแบบที่มีความยืดหยุ่นต่อการใช้งานเนื่องจากไม่จำเป็นต้องมีเงื่อนไข(Assumptions)ในการสร้างตัวแบบ ข้อดีของตัวแบบโครงข่ายประสาทเทียมคือ มีความแม่นยำสูงในการพยากรณ์ข้อมูลอนุกรมเวลาในส่วนที่ไม่เป็นฟังก์ชันเชิงเส้นตรง อีกทั้งยังมีความแกร่ง(Robust)เมื่อมีข้อมูลรบกวน(Noisy data) เช่น ข้อมูลมีค่านอกเกณฑ์(Outlier) เป็นต้น และในปี ค.ศ. 1995 ก็ได้มีการพัฒนาตัวแบบสำหรับการพยากรณ์ขึ้นมาใหม่ คือ ตัวแบบ Support vector machine (SVM) โดยมีแนวคิดมาจากตัว

แบบโครงข่ายประสาทเทียม แต่ SVM นั้นมีข้อดีกว่าตรงที่มักจะไม่เกิดปัญหา “Overfitting” มากนักเหมือนกับตัวแบบ ANN

ทั้งนี้การใช้ตัวแบบเดี่ยวในการพยากรณ์ข้อมูลอนุกรมเวลานั้นยังมีข้อจำกัด กล่าวคือแม้ตัวแบบ ARIMA จะเป็นตัวแบบที่สามารถจับกับข้อมูลอนุกรมเวลาส่วนที่เป็นฟังก์ชันเชิงเส้นตรง(Linear Component)ได้ดี แต่การใช้ตัวแบบ ARIMA เพียงตัวแบบเดียวอาจไม่เพียงพอในการพยากรณ์ข้อมูลอนุกรมเวลาที่มีความซับซ้อนมากขึ้น เนื่องจากในความเป็นจริงนั้นข้อมูลอนุกรมเวลาจะมีทั้งรูปแบบส่วนที่เป็นเส้นตรง และไม่เป็นเส้นตรง น้อยมากที่จะมีส่วนประกอบแค่เชิงเส้นตรงเพียงอย่างเดียว หรือไม่เป็นเชิงเส้นตรงเพียงอย่างเดียว ในขณะที่ตัวแบบ ANN และ SVM นั้นจะจับกับข้อมูลอนุกรมเวลาส่วนที่ไม่เป็นฟังก์ชันเชิงเส้นตรง(Nonlinear Component)ได้ดี ดังนั้น ตัวแบบผสม (Hybrid Model)ซึ่งเกิดจากการนำตัวแบบเดี่ยวมากกว่า 1 ชนิดขึ้นไปมารวมกัน เช่น ตัวแบบผสมระหว่าง ARIMA กับ ANN หรือตัวแบบผสมระหว่าง ARIMA กับ SVM จึงได้ถูกนำมาใช้ในการวิเคราะห์ข้อมูลอนุกรมเวลาเพื่อเพิ่มความถูกต้องของการพยากรณ์ ทำให้การพยากรณ์มีประสิทธิภาพและความแม่นยำมากขึ้น(G. P. ZHANG, 2003)

(ภัทร วรภู, 2556)ได้ทำการเปรียบเทียบความแม่นยำของการพยากรณ์อนุกรมเวลาระหว่างตัวแบบผสมและตัวแบบเดี่ยว โดยทำการเปรียบเทียบตัวแบบทั้งหมด 6 ตัวแบบ คือ ตัวแบบ ARIMA, ตัวแบบANN, ตัวแบบSVM, ตัวแบบผสมระหว่างตัวแบบ ARIMA กับตัวแบบANN, ตัวแบบผสมระหว่างตัวแบบ ARIMA กับตัวแบบSVM และตัวแบบ Hybrid Combined (ARIMA+ANN +SVM) โดยใช้ข้อมูลอัตราแลกเปลี่ยนเงินสกุลปอนด์และเงินดอลลาร์สหรัฐรายปี และเปรียบเทียบความแม่นยำของตัวแบบโดยพิจารณาจากค่าคลาดเคลื่อนกำลังสองเฉลี่ย(MSE), รากของค่าคลาดเคลื่อนกำลังสองเฉลี่ย(RMSE), ค่าคลาดเคลื่อนสมบูรณ์เฉลี่ย(MAE) และเปอร์เซ็นต์ค่าคลาดเคลื่อนสมบูรณ์เฉลี่ย(MAPE) สรุปได้ว่าตัวแบบผสมไม่ได้มีความแม่นยำในการพยากรณ์สูงกว่าตัวแบบเดี่ยวเสมอไป แต่งานวิจัยนี้ก็ยังไม่มีการทำการจำลองข้อมูล(simulation)อนุกรมเวลาเพื่อศึกษาและเปรียบเทียบความแม่นยำของการพยากรณ์

(ชญานิน บุญมานะ และ นัท กุลวานิช, 2560) ได้ศึกษาเปรียบเทียบความแม่นยำของการพยากรณ์ด้วยตัวแบบอนุกรมเวลาแบบผสม โดยเปรียบเทียบความแม่นยำของค่าพยากรณ์ที่ได้จากตัวแบบ ARIMA, ตัวแบบผสมระหว่าง ARIMA กับ ANN และตัวแบบผสมระหว่าง ARIMA กับSVM ในการพยากรณ์ราคาปิดหุ้น SCB ของธนาคารไทยพาณิชย์ จำกัด (มหาชน) โดยใช้ทั้งชุดข้อมูลจริงและชุดข้อมูลจำลอง สำหรับการพยากรณ์ในชุดข้อมูลจริงของราคาปิดหุ้น SCB รายสัปดาห์ของธนาคารไทยพาณิชย์ จำกัด (มหาชน) ที่มีลักษณะอนุกรมเวลาสอดคล้องกับตัวแบบ ARIMA(1,1,1) ผลการศึกษาพบว่าตัวแบบผสมระหว่าง ARIMA และซัพพอร์ทเวกเตอร์แมชชีน มีความแม่นยำในการพยากรณ์สูงที่สุดซึ่งสอดคล้องกับผลการวิเคราะห์ด้วยชุดข้อมูลจำลอง แต่งานวิจัยนี้ยังเป็นเพียง

การศึกษาเปรียบเทียบความแม่นยำของการพยากรณ์ของข้อมูลอนุกรมเวลาซึ่งไม่มีส่วนประกอบเชิงฤดูกาล(Non-Seasonal time series)

ดังนั้น ในการศึกษาครั้งนี้ผู้วิจัยจึงสนใจที่จะทำการศึกษาและเปรียบเทียบความแม่นยำของการพยากรณ์ข้อมูลอนุกรมเวลา โดยขยายขอบเขตงานวิจัยของชญานิน บุญมานะ และนัท กุลวานิช ในการทำการศึกษาในส่วนของข้อมูลจำลองจะทำการเปรียบเทียบค่าพยากรณ์ที่ได้จากตัวแบบทั้ง 3 ตัวแบบ คือ ตัวแบบ ARIMA ที่มีฤดูกาล(SARIMA), ตัวแบบผสมระหว่างตัวแบบตัวแบบ ARIMA ที่มีฤดูกาลกับตัวแบบโครงข่ายประสาทเทียม(SARIMA+ANN) และตัวแบบผสมระหว่างตัวแบบ ARIMA ที่มีฤดูกาลกับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน(SARIMA+SVM) และในส่วนของข้อมูลจริงได้มีการนำราคาขายปลีกมะนาวเบอร์ 1-2 (หน่วยเป็นบาท/ผล) จากกรมการค้าภายใน กระทรวงพาณิชย์ ซึ่งเป็นราคาผลผลิตทางการเกษตรซึ่งอยู่ในรูปแบบอนุกรมเวลาที่มีปัจจัยเชิงฤดูกาลมาทำการเปรียบเทียบ จากนั้นทำการเปรียบเทียบตัวแบบที่ให้ค่าพยากรณ์ที่แม่นยำที่สุดโดยพิจารณาจากรากของค่าคลาดเคลื่อนกำลังสองเฉลี่ย(Root Mean Square Error: RMSE) โดยตัวแบบที่มีความแม่นยำในการพยากรณ์มากที่สุด คือ ตัวแบบที่มีค่า RMSE ต่ำที่สุด

1.2 วัตถุประสงค์ในการวิจัย

1. เพื่อเปรียบเทียบความแม่นยำของค่าพยากรณ์ที่ได้จากตัวแบบ ARIMA ที่มีฤดูกาล (SARIMA), ตัวแบบผสมระหว่างตัวแบบ ARIMA ที่มีฤดูกาลกับตัวแบบโครงข่ายประสาทเทียม(SARIMA+ANN) และตัวแบบผสมระหว่างตัวแบบ ARIMA ที่มีฤดูกาลกับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน(SARIMA+SVM) โดยใช้ข้อมูลจำลอง(Simulation data)
2. เพื่อเปรียบเทียบความแม่นยำของค่าพยากรณ์ที่ได้จากตัวแบบ ARIMA ที่มีฤดูกาล (SARIMA), ตัวแบบผสมระหว่างตัวแบบ ARIMA ที่มีฤดูกาลกับตัวแบบโครงข่ายประสาทเทียม(SARIMA+ANN) และตัวแบบผสมระหว่างตัวแบบ ARIMA ที่มีฤดูกาลกับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน(SARIMA+SVM) โดยใช้ข้อมูลจริง คือ ราคาขายปลีกมะนาวเบอร์ 1-2 (หน่วยเป็นบาท/ผล) จากกรมการค้าภายในกระทรวงพาณิชย์

1.3 สมมติฐานของการวิจัย

การพยากรณ์ข้อมูลราคาขายปลีกมะนาวด้วยตัวแบบผสม(Hybrid Model) คือ ตัวแบบผสมระหว่างตัวแบบ ARIMA ที่มีฤดูกาลกับตัวแบบโครงข่ายประสาทเทียม หรือตัวแบบผสมระหว่างตัวแบบ ARIMA ที่มีฤดูกาลกับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน จะมีความแม่นยำในการพยากรณ์มากกว่าการใช้ตัวแบบ ARIMA ที่มีฤดูกาลเพียงตัวแบบเดียว

1.4 ขอบเขตของการวิจัย

ในการศึกษาวิจัยครั้งนี้จะทำการศึกษาในส่วนของข้อมูลจริงและข้อมูลจำลอง ภายใต้ขอบเขตของการวิจัยดังนี้

1.4.1 ข้อมูลจำลอง

ทำการจำลองชุดข้อมูลด้วยตัวแบบ ARIMA(p,1,q)x(P,1,Q)$_s$ ภายใต้ค่า P=0,1 ,Q=0,1 ,p=1,2 และ q=1,2 กล่าวคือจะทำการจำลองชุดข้อมูลอนุกรมเวลาทั้งหมด 12 ตัวแบบ

โดยกำหนดค่าพารามิเตอร์ดังนี้

Φ = -0.5 และ 0.5

Θ = -0.5 และ 0.5

ϕ = -0.5 และ 0.5

θ = -0.5 และ 0.5

และจะพิจารณาเฉพาะค่าพารามิเตอร์ทำให้ชุดข้อมูลจำลองมีคุณสมบัติ Causality และ Invertibility ซึ่งมีทั้งหมด 128 กรณี ดังนี้

แบบจำลองที่ 1 ARIMA(1,1,1)x(0,1,1)$_{12}$

โดยที่ $\Theta = -0.5, 0.5$, $\phi = -0.5, 0.5$, $\theta = -0.5, 0.5$

มีทั้งสิ้น 8 กรณีย่อย

แบบจำลองที่ 2 ARIMA(1,1,2)x(0,1,1)$_{12}$

โดยที่ $\Theta = -0.5, 0.5$, $\phi = -0.5, 0.5$, $\theta_1 = -0.5, 0.5$, $\theta_2 = -0.5$

มีทั้งสิ้น 8 กรณีย่อย

แบบจำลองที่ 3 ARIMA(2,1,1)x(0,1,1)$_{12}$

โดยที่ $\Theta = -0.5, 0.5$, $\phi_1 = -0.5, 0.5$, $\phi_2 = -0.5$, $\theta = -0.5, 0.5$

มีทั้งสิ้น 8 กรณีย่อย

แบบจำลองที่ 4 ARIMA(2,1,2)x(0,1,1)$_{12}$

โดยที่ $\Theta = -0.5, 0.5$, $\phi_1 = -0.5, 0.5$, $\phi_2 = -0.5$, $\theta_1 = -0.5, 0.5$, $\theta_2 = -0.5$

มีทั้งสิ้น 8 กรณีย่อย

แบบจำลองที่ 5 ARIMA(1,1,1)x(1,1,0)$_{12}$

โดยที่ $\Phi = -0.5, 0.5$, $\phi = -0.5, 0.5$, $\theta = -0.5, 0.5$

มีทั้งสิ้น 8 กรณีย่อย

แบบจำลองที่ 6 ARIMA(1,1,2)x(1,1,0)$_{12}$

โดยที่ $\Phi = -0.5, 0.5$, $\phi = -0.5, 0.5$, $\theta_1 = -0.5, 0.5$, $\theta_2 = -0.5$

มีทั้งสิ้น 8 กรณีย่อย

แบบจำลองที่ 7 ARIMA(2,1,1)x$(1,1,0)_{12}$

โดยที่ $\Phi = -0.5, 0.5$, $\phi_1 = -0.5, 0.5$, $\phi_2 = -0.5$, $\theta = -0.5, 0.5$

มีทั้งสิ้น 8 กรณีย่อย

แบบจำลองที่ 8 ARIMA(2,1,2)x$(1,1,0)_{12}$

โดยที่ $\Phi = -0.5, 0.5$, $\phi_1 = -0.5, 0.5$, $\phi_2 = -0.5$, $\theta_1 = -0.5, 0.5$, $\theta_2 = -0.5$

มีทั้งสิ้น 8 กรณีย่อย

แบบจำลองที่ 9 ARIMA(1,1,1)x$(1,1,1)_{12}$

โดยที่ $\Phi = -0.5, 0.5$, $\Theta = -0.5, 0.5$, $\phi = -0.5, 0.5$, $\theta = -0.5, 0.5$

มีทั้งสิ้น 16 กรณีย่อย

แบบจำลองที่ 10 ARIMA(1,1,2)x$(1,1,1)_{12}$

โดยที่ $\Phi = -0.5, 0.5$, $\Theta = -0.5, 0.5$, $\phi = -0.5, 0.5$, $\theta_1 = -0.5, 0.5$, $\theta_2 = -0.5$

มีทั้งสิ้น 16 กรณีย่อย

แบบจำลองที่ 11 ARIMA(2,1,1)x$(1,1,1)_{12}$

โดยที่ $\Phi = -0.5, 0.5$, $\Theta = -0.5, 0.5$, $\phi_1 = -0.5, 0.5$, $\phi_2 = -0.5$, $\theta = -0.5, 0.5$

มีทั้งสิ้น 16 กรณีย่อย

แบบจำลองที่ 12 ARIMA(2,1,2)x$(1,1,1)_{12}$

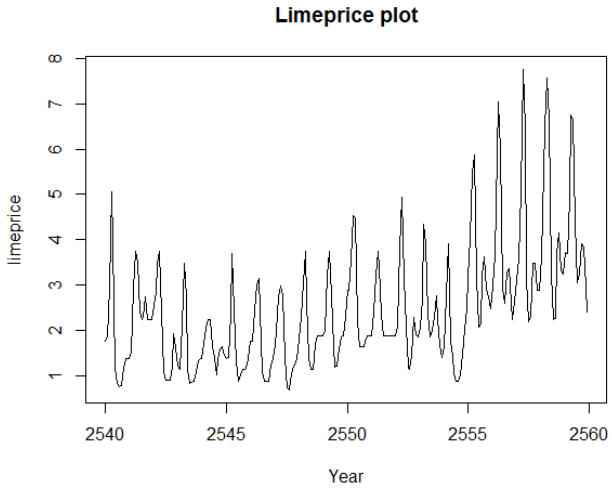
โดยที่ $\Phi = -0.5, 0.5$, $\Theta = -0.5, 0.5$, $\phi_1 = -0.5, 0.5$, $\phi_2 = -0.5$, $\theta_1 = -0.5, 0.5$, $\theta_2 = -0.5$ มีทั้งสิ้น 16 กรณีย่อย

โดยในแต่ละตัวแบบจะทำการจำลองโดยใช้จำนวนรอบในการทำซ้ำเท่ากับ 100 รอบ จากนั้นจะนำค่าส่วนเหลือ(Residuals) ที่ได้จากการพยากรณ์ด้วยตัวแบบ SARIMA ในชุดข้อมูลจำลองไปพยากรณ์ด้วยตัวแบบผสมระหว่าง ARIMA กับโครงข่ายประสาทเทียม(SARIMA+ANN) และตัวแบบผสมระหว่าง ARIMA กับซัพพอร์ทเวกเตอร์แมชชีน(SARIMA+SVM)ตามลำดับ สุดท้ายทำการเปรียบเทียบความแม่นยำของค่าพยากรณ์ที่ได้จาก 3 ตัวแบบ คือ ตัวแบบ SARIMA, ตัวแบบผสม SARIMA+ANN และตัวแบบผสม SARIMA+SVM ในแต่ละแบบจำลอง โดยตัวแบบที่มีค่า RMSE ต่ำสุด จะเป็นตัวแบบที่ดีที่สุด

1.4.2 ข้อมูลจริง

1.4.2.1 นำข้อมูลราคาขายปลีกมะนาวเบอร์ 1-2 (หน่วยเป็นบาท/ผล) จากกรมการค้าภายในกระทรวงพาณิชย์ โดยใช้ข้อมูลเป็นรายเดือนตั้งแต่เดือนมกราคมปี พ.ศ. 2540 ถึง

เดือนธันวาคมปี พ.ศ.2559 จำนวน 240 ตัวอย่าง มาทำการหาตัวแบบอนุกรมเวลาที่มีปัจจัยเชิงฤดูกาลที่เหมาะสมที่สุด พบว่าตัวแบบอนุกรมเวลาที่เหมาะสมที่สุดคือ ตัวแบบARIMA(1,1,2)x$(0,1,1)_{12}$



ภาพที่ 1.1 Time plot ของข้อมูลราคาขายปลีกมะนาว

1.4.2.2 สร้างตัวแบบผสมระหว่าง SARIMA กับตัวแบบโครงข่ายประสาทเทียม (SARIMA+ANN) โดยนำค่าส่วนเหลือ(Residuals)ที่ได้จากตัวแบบ ARIMA(1,1,2)x$(0,1,1)_{12}$ ที่คัดเลือกได้ไปพยากรณ์ด้วยตัวแบบโครงข่ายประสาทเทียมเพอร์เซ็ปตรอนหลายชั้น(MLP) ประกอบด้วยชั้น Input layer 1 ชั้น จำนวน 1 โหนด , Hidden layer 1 ชั้น จำนวน 1 ถึง 5 โหนด โดยใช้ Sigmoid logistics function เป็น Activation function และ Output layer 1 ชั้น ภายใต้จำนวนโหนด 1 โหนด โดยใช้ Linear function เป็น Activation function และใช้เทคนิคการฝึกสอนโครงข่ายแบบ Resilient back propagation(Rprop) โดยกำหนดจำนวนรอบในการทำซ้ำเท่ากับ 1000 รอบ เพื่อหาจำนวน hidden neuron และชุดของ startweights ที่เหมาะสมที่สุด โดยพิจารณาจากค่า RMSE ที่ต่ำที่สุด

1.4.2.3 สร้างตัวแบบผสมระหว่าง SARIMA กับซัพพอร์ทเวกเตอร์แมชชีน (SARIMA+SVM) โดยนำค่าส่วนเหลือ(Residuals)ที่ได้จากตัวแบบ ARIMA(1,1,2)x$(0,1,1)_{12}$ ที่คัดเลือกได้ไปพยากรณ์ด้วยตัวแบบซัพพอร์ทเวกเตอร์แมชชีนสำหรับการพยากรณ์อนุกรมเวลา โดยทำการปรับจูนหาชุดของค่าพารามิเตอร์ของ SVM คือ C และ ε ที่เหมาะสมที่สุด โดยใช้ค่า $C=2^{-5}$, 2^{-3} ,...,2^{3},2^{5} และ ε = 0.001, 0.01, 0.1 และ 1 และใช้ Gaussian radial basis function(RBF) เป็น Kernel function โดยจะหาค่าพารามิเตอร์ σ ที่ดีที่สุดจากการทำซ้ำ 1000 รอบ จากนั้นทำการพิจารณาคัดเลือกตัวแบบซัพพอร์ทเวกเตอร์แมชชีนที่เหมาะสมที่สุด โดยพิจารณาจากค่า RMSE ที่ต่ำที่สุด

1.4.2.4 ทำการเปรียบเทียบความแม่นยำของค่าพยากรณ์ที่ได้จาก 3 ตัวแบบ คือ ตัวแบบ SARIMA, ตัวแบบผสม SARIMA+ANN และตัวแบบผสม SARIMA+SVM โดยตัวแบบที่มีค่า RMSE ต่ำที่สุด จะเป็นตัวแบบที่ดีที่สุด

1.5 ประโยชน์ที่คาดว่าจะได้รับ

เพื่อเป็นแนวทางในการพิจารณาใช้ตัวแบบอนุกรมเวลาแบบผสมสำหรับการพยากรณ์ข้อมูลอนุกรมเวลาที่มีส่วนประกอบเชิงฤดูกาลอื่นๆต่อไป

บทที่ 2
ทฤษฎีและกรอบแนวคิดที่เกี่ยวข้อง

ในบทนี้ ผู้วิจัยจะกล่าวถึงแนวคิดทฤษฎีต่างๆเพื่อเป็นพื้นฐานในการศึกษาวิเคราะห์ และเปรียบเทียบตัวแบบอนุกรมเวลาแบบผสมสำหรับการพยากรณ์ข้อมูลอนุกรมเวลาที่มีปัจจัยเชิงฤดูกาล โดยผู้วิจัยได้แบ่งการนำเสนอออกเป็น 6 ส่วนหลักๆ คือ ตัวแบบ ARIMA, ตัวแบบ ARIMA ที่มีฤดูกาล(SARIMA), ตัวแบบโครงข่ายประสาทเทียม(ANN) ตัวแบบซัพพอร์ตเวกเตอร์แมชชีน(SVM) ตัวแบบผสม(Hybrid Model) และเกณฑ์ที่ใช้ในการตัดสินใจ โดยมีรายละเอียดดังต่อไปนี้

2.1 ตัวแบบ Autoregressive integrated moving average (ARIMA)

ตัวแบบ ARIMA หรือรู้จักกันในชื่อของตัวแบบบ็อกซ์และเจนกินส์(Box-Jenskins model) เป็นตัวแบบที่นิยมนำมาใช้ในการพยากรณ์อนุกรมเวลาเชิงปริมาณซึ่งได้รับความนิยมมากมาจนถึงปัจจุบัน โดยตัวแบบ ARIMA นั้นประกอบไปด้วย 3 ส่วนหลักๆ คือ AR term, MA term และ Differencing operation(I) โดยตัวแบบของ Box&Jenskins นั้นมีเงื่อนไขว่าอนุกรมเวลานั้นต้องมีความนิ่ง หรือคงที่(Stationary) และไม่มีความแปรผันเนื่องจากฤดูกาล(Seasonal Variations)

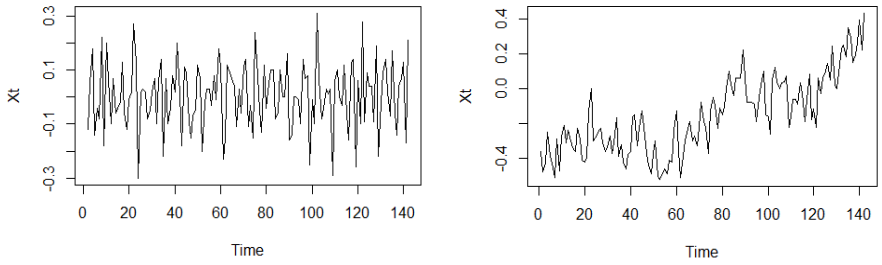
อนุกรมเวลาของตัวแปร X_t จะมีความนิ่ง(stationary) ก็ต่อเมื่อมีคุณสมบัติดังต่อไปนี้(ภูมิฐาน รังคกูลนุวัฒน์, 2556)

1) $E(X_t)=\mu,\ t=1,2,...,T$ หรือค่าเฉลี่ยของตัวแปร X ในแต่ละช่วงเวลา มีค่าคงที่ ไม่ขึ้นกับ t
2) $Var(X_t)=E(X_t-\mu)^2=\sigma_x^2,\ t=1,2,...,T$ หรือความแปรปรวนของตัวแปร X ในแต่ละช่วงเวลา มีค่าคงที่ ไม่ขึ้นกับ t
3) $Cov(X_t,X_{t+h})=\gamma_h,\ h=1,2,...,T$ หรือความแปรร่วมระหว่างข้อมูลอนุกรมเวลา X_t ที่ต่างช่วงเวลากันจะขึ้นกับอยู่ lag h หรือระยะห่างระหว่างช่วงเวลาทั้งสองเท่านั้น ไม่ขึ้นกับ t

   Note $Cov(X_t,X_t)=Var(X_t)=\gamma_0$ เมื่อ h=0

ลักษณะอนุกรมเวลาที่คงที่(Stationary time series) และไม่คงที่(Non-stationary time series) แสดงดังภาพที่ 2.1 กล่าวคือเมื่อนำข้อมูลมาพล็อตลงในกราฟที่มีแกนตั้งคือ X_t และแกนนอน คือ เวลา เริ่มที่ t=1,2,...,T จะพบว่าลักษณะของกราฟจะผันผวนในอัตราคงที่รอบๆค่าคงที่ค่าใดค่า

หนึ่งหากอนุกรมเวลามีลักษณะคงที่ แต่หากอนุกรมเวลามีลักษณะไม่คงที่แล้วนั้นกราฟที่ได้จะผันผวนในอัตราไม่คงที่(อาจผันผวนมากขึ้นหรือน้อยลงรอบๆค่าคงที่ค่าหนึ่งก็ได้)



(ก) (ข)

ภาพที่ 2.1 อนุกรมเวลาที่มีลักษณะคงที่(ก) และไม่คงที่(ข)

เมื่อพบว่าอนุกรมเวลาไม่คงที่(Non-stationary) จะต้องแปลงอนุกรมเวลานั้นให้คงที่เสียก่อน ซึ่งวิธีการหนึ่งที่นิยมนำมาใช้ในการแปลงอนุกรมเวลาที่ไม่คงที่ให้เป็นอนุกรมเวลาแบบคงที่ก็คือวิธีการหาผลต่าง(differencing) ซึ่งจะทำให้เกิด Differencing operation(I) และทำให้เกิดเป็นตัวแบบ Autoregressive integrated moving average(ARIMA)

2.1.1 ตัวแบบ Autoregressive moving average (ARMA)

เป็นตัวแบบซึ่งอธิบายอนุกรมเวลา X_t เป็นฟังก์ชันเชิงเส้นของค่าสังเกตในอดีต และตัวรบกวนขาวทั้งในอดีตและปัจจุบัน โดยมีตัวแบบคือ

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \ldots + \phi_p X_{t-p} + W_t + \theta_1 W_{t-1} + \ldots + \theta_q W_{t-q}$$

เมื่อ X_t คือ อนุกรมเวลาที่คงที่

$\phi_1,..,\phi_p$ และ $\theta_1,..,\theta_p$ คือ พารามิเตอร์ของตัวแบบ

W_t คือ ตัวรบกวนขาวแบบเกาส์เซียน(Gaussian white noise) โดย $W_t \sim (0, \sigma_w^2)$

ซึ่งตัวแบบ ARMA(p,q) นั้นสามารถเขียนได้ในอีกรูปแบบหนึ่งคือ

$$\phi(B) X_t = \theta(B) W_t$$

เมื่อ $\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \ldots - \phi_p B^p$ คือ autoregressive operator

$\theta(B) = 1 + \theta_1 B + \theta_2 B^2 + \ldots + \theta_q B^q$ คือ moving average operator

B คือ Backshift operator ($B^k X_t = X_{t-k}$, k=0,1,2,...)

ตัวแบบ ARMA(p,q) เป็นตัวแบบที่ผสมผสานระหว่าง 2 ตัวแบบ คือ

1. AR(p)-Autoregressive order p

เป็นตัวแบบที่แสดงถึงอนุกรมเวลา X_t ขึ้นอยู่กับค่าของมันเองในอดีตที่ผ่านมา กล่าวคือค่าสังเกต X_t ขึ้นกับค่าสังเกต ณ เวลาก่อนหน้าคือ $X_{t-1},..,X_{t-p}$ โดยมีรูปแบบสมการทางคณิตศาสตร์คือ

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + ... + \phi_p X_{t-p} + W_t$$

2. MA(q)-Moving average order q

เป็นตัวแบบที่แสดงถึงอนุกรมเวลา X_t ขึ้นอยู่กับตัวรบกวนขาว(ε) ตั้งแต่อดีตจนถึงปัจจุบัน กล่าวคือค่าสังเกต X_t ขึ้นกับค่าความคลาดเคลื่อน ณ เวลา $W_t,W_{t-1},.., W_{t-q}$ โดยมีรูปแบบสมการทางคณิตศาสตร์คือ

$$X_t = W_t + \theta_1 W_{t-1} + \theta_2 W_{t-2} + ... + \theta_q W_{t-q}$$

2.1.2 ตัวแบบ Autoregressive integrated moving average (ARIMA)

เป็นตัวแบบ Autoregressive moving average(ARMA) ที่มีการหาผลต่าง differencing อันดับที่ d เพื่อแปลงอนุกรมเวลาที่ไม่คงที่(Non-stationary) ให้เป็นอนุกรมเวลาใหม่ที่คงที่ (stationary) โดยสัญลักษณ์ I คือ differencing operations ซึ่งสามารถเขียนได้ว่า

$$\nabla^d X_t = (1-B)^d X_t$$

โดยตัวแบบ ARIMA มีรูปแบบทั่วไปคือ

$$\phi(B)(1-B)^d X_t = \theta(B) W_t$$

เมื่อ $\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - ... - \phi_p B^p$ คือ autoregressive operator

$\theta(B) = 1 + \theta_1 B + \theta_2 B^2 + ... + \theta_q B^q$ คือ moving average operator

ถ้าอนุกรมเวลา X_t มีค่าเฉลี่ยไม่เท่ากับ 0 กล่าวคือ $E(\nabla^d X_t) = \mu$ เราจะเขียนตัวแบบ ARIMA(p,q) ข้างต้นให้อยู่ในรูปสมการ

$$\phi(B)(1-B)^d X_t = \alpha + \theta(B) W_t$$

เมื่อ $\alpha = \mu(1 - \phi_1 - ... - \phi_p)$

2.1.3 การแปลงอนุกรมเวลาให้คงที่ด้วยวิธีหาผลต่าง(Differencing)

เมื่อพบว่าอนุกรมเวลาไม่มีคุณสมบัติคงที่(stationary) ให้ทำการแปลงให้เป็นอนุกรมเวลาที่คงที่ โดยเริ่มจากการหาผลต่างครั้งที่ 1(First Differencing) ค่า d ที่น้อยที่สุดคือ d=1 ในตัวแบบ ARIMA(p,d,q) แล้วตรวจสอบอนุกรมเวลาใหม่อีกครั้ง ถ้าอนุกรมเวลามีระดับของกราฟคงที่ และ Sample ACF ลดลงอย่างรวดเร็ว แสดงว่าอนุกรมเวลาที่ได้ใหม่นั้นมีคุณสมบัติคงที่แล้ว ให้ทำการเลือกตัวแบบที่เหมาะสมในขั้นต่อไป แต่หากอนุกรมเวลายังไม่คงที่ ให้ทำการหาผลต่างเป็นครั้งที่ 2 (Second Differencing) และครั้งต่อๆไปจนได้อนุกรมเวลาที่คงที่

First Differencing : $\nabla X_t = (1-B)X_t = X_t - X_{t-1}$

Second Differencing : $\nabla^2 X_t = (1-B)^2 X_t = X_t - 2X_{t-1} + X_{t-2}$

2.1.4 ฟังก์ชันสหสัมพันธ์ในตัวเอง(Autocorrelation Function: ACF)

กำหนด $X_1, X_2, \ldots, X_n$ คืออนุกรมเวลาชุดหนึ่งที่คงที่(stationary) มีจำนวนข้อมูลเท่ากับ n ค่าสหสัมพันธ์ในตัวเอง ณ h ช่วงเวลาที่แล้วเขียนแทนด้วยสัญลักษณ์ $\rho(h)$ คำนวณได้จาก

$$\rho(h) = \frac{\text{cov}(X_t, X_{t+h})}{\sqrt{\text{var}(X_t)}\sqrt{\text{var}(X_{t+h})}} = \frac{E[(X_t - E(X_t))(X_{t+h} - E(X_{t+h}))]}{\sqrt{\text{var}(X_t)}\sqrt{\text{var}(X_{t+h})}}$$

เมื่ออนุกรมเวลามีความนิ่ง(Stationary) จะได้ว่า $E(X_t)=E(X_{t+h})$ และ $\text{Var}(X_t)=\text{Var}(X_{t+h})$ จะได้ว่า

$$\rho(h) = \frac{\text{cov}(X_t, X_{t+h})}{\text{var}(X_t)} = \frac{\gamma_h}{\gamma_0}$$

ACF เป็นค่าที่ใช้วัดความสัมพันธ์เชิงเส้นตรงของจุด 2 จุดเวลาที่ต่างกัน กล่าวคือค่า ACF ก็คือค่าสหสัมพันธ์(correlation) ระหว่างอนุกรมเวลา ณ ช่วงเวลาปัจจุบัน(X_t) กับอนุกรมเวลาที่อยู่ห่างกัน h ช่วงเวลา ซึ่งอาจุเป็น h ช่วงเวลาถัดไป (X_{t+h}) หรือ ณ h ช่วงเวลาก่อนหน้า(X_{t-h})ก็ได้ และมีคุณสมบัติดังนี้

- $\rho(h)$ มีค่าอยู่ระหว่าง -1 ถึง 1
- $|\rho(h)|$ มีค่าเข้าใกล้ 1 แสดงว่าค่าสังเกตที่อยู่ห่างกัน h ช่วงเวลามีสหสัมพันธ์กันสูง
- $|\rho(h)|$ มีค่าเข้าใกล้ 0 แสดงว่าค่าสังเกตที่อยู่ห่างกัน h ช่วงเวลามีสหสัมพันธ์กันต่ำ
- $\rho(h) > 0$ แสดงว่าค่าสังเกตที่อยู่ห่างกัน h ช่วงเวลามีสหสัมพันธ์ในทิศทางเดียวกัน
- $\rho(h) < 0$ แสดงว่าค่าสังเกตที่อยู่ห่างกัน h ช่วงเวลามีสหสัมพันธ์ในทิศทางตรงกันข้าม

- $\rho(0)=1$ เสมอ เนื่องจาก $\rho(0)=\frac{\text{cov}(X_t,X_t)}{\text{var}(X_t)}=\frac{\text{var}(X_t)}{\text{var}(X_t)}=1$ หรือจะพิจารณาว่าค่าสังเกตที่อยู่ห่างกัน h=0 ช่วงเวลา ก็คือค่าสังเกตตัวมันเองจะมีความสัมพันธ์กับตัวมันเองมากที่สุดซึ่งเท่ากับ 1 นั่นเอง

แต่เนื่องจากอนุกรมเวลาที่นำมาพิจารณาเป็นเพียงตัวอย่างสุ่ม ดังนั้น ในการวิเคราะห์ข้อมูลอนุกรมเวลาใดๆจึงจะพิจารณาจากค่า Sample Autocorrelation Function(Sample ACF) ซึ่งอาจแตกต่างไปบ้างเมื่อเทียบกับ ACF จากทฤษฎี

การคำนวณค่า Sample ACF : $\hat{\rho}(h)$ นั้นมีจุดประสงค์เพื่อนำมาใช้ประมาณค่า Theoretical autocorrelation Function : $\rho(h)$ และค่า Sample ACF ที่ได้นั้นจะถูกนำไปพล็อตลงใน “Correlogram” ซึ่งเป็นกราฟเพื่อใช้ประเมินว่าอนุกรมเวลาคงที่หรือไม่ นอกจากนี้ค่า Sample ACF ยังช่วยในการตัดสินใจเบื้องต้นว่าควรเลือกแบบจำลองของ Box-Jenskin ชนิดใดกับอนุกรมเวลาที่กำลังพิจารณาอยู่

2.1.5 ฟังก์ชันสหสัมพันธ์ในตัวเองบางส่วน(Partial Autocorrelation Function: PACF)

เป็นค่าที่ใช้วัดความสัมพันธ์เชิงเส้นตรงระหว่าง X_t กับ X_{t+h} โดยไม่มีอิทธิพลของ $X_{t+1},\ldots,X_{t+h-1}$ เข้ามาเกี่ยวข้อง เขียนแทนด้วยสัญลักษณ์ ϕ_{hh} , h=1,2,3,... เช่น ϕ_{33} เป็นความสัมพันธ์เชิงเส้นตรงระหว่าง X_t กับ X_{t-3} โดยไม่มีอิทธิพลของ X_{t-1} และ X_{t-2} เข้ามาเกี่ยวข้อง

$$\phi_{11}=corr(X_1,X_0)=\rho(1)$$

.

.

$$\phi_{hh}=corr(X_h-X_h^{h-1},X_0-X_0^{h-1}) \quad ,h=2,3,..$$

แต่เนื่องจากอนุกรมเวลาที่นำมาพิจารณาเป็นเพียงตัวอย่างสุ่ม ดังนั้น ในการวิเคราะห์ข้อมูลอนุกรมเวลาใดๆจึงจะพิจารณาจากค่า Sample Partial Autocorrelation Function(Sample PACF) ซึ่งอาจแตกต่างไปบ้างเมื่อเทียบกับ PACF จากทฤษฎี ซึ่งค่า PACF ของตัวอย่างที่ได้ (Sample PACF: $\hat{\phi}_{hh}$) นั้นจะถูกนำไปพล็อตเป็นกราฟ PACF เพื่อช่วยในการตัดสินใจเลือกตัวแบบอนุกรมเวลาประกอบกับกราฟ Sample ACF

2.1.6 ลักษณะของ ACF และ PACF ของตัวแบบ ARMA

ลักษณะของ ACF และ PACF ของตัวแบบ ARMA(p,q) สามารถสรุปได้ดังตารางด้านล่าง

	AR(p)	MA(q)	ARMA(p,q)
ACF	มีค่าลดลงอย่างรวดเร็ว (Tail off)	สิ้นสุดหลังจาก q ช่วงเวลาที่แล้ว	มีค่าลดลงอย่างรวดเร็ว (Tail off)
PACF	สิ้นสุดหลังจาก p ช่วงเวลาที่แล้ว	มีค่าลดลงอย่างรวดเร็ว (Tail off)	มีค่าลดลงอย่างรวดเร็ว (Tail off)

ตารางที่ 2.1 ตารางแสดงลักษณะของ ACF และ PACF ของตัวแบบ ARMA
(ที่มา : Shumway,2010: 108)

2.1.7 Causality และ Invertibility ของตัวแบบ ARMA

- Causality

Causality เป็นเงื่อนไขของพารามิเตอร์ $\phi_1,..,\phi_p$ ในรูปแบบ AR(p) ที่ทำให้อนุกรมเวลาในรูปแบบ AR(p) มีคุณสมบัติคงที่(Stationary) กล่าวคือค่าคาดหวังของ X_t และค่าความแปรปรวนของ X_t ที่เวลา t คงที่ และทำให้ค่าสัมประสิทธิ์สหสัมพันธ์ของตัวอย่าง(Sample ACF) ที่ต่างช่วงเวลากันจะขึ้นกับอยู่กับระยะห่างระหว่างช่วงเวลา(หรือ lag h) เพียงอย่างเดียว(ทรงศิริ แต้สมบัติ, 2553)

การพิจารณาค่าของพารามิเตอร์ $\phi_1,..,\phi_p$ ที่ทำให้รูปแบบ AR(p) มีคุณสมบัติ Causality สามารถพิจารณาได้ดังนี้ พิจารณารูปแบบ AR(p)

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + ... + \phi_p X_{t-p} + W_t$$

เขียนรูปแบบ AR(P) ใหม่ โดยคงเฉพาะเทอม W_t ไว้ทางขวามือ

$$X_t - \phi_1 X_{t-1} - \phi_2 X_{t-2} - ... - \phi_p X_{t-p} = W_t$$

และเขียนให้อยู่ในรูปแบบ Backshift operator

$$(1 - \phi_1 B - \phi_2 B^2 - ... - \phi_p B^p) X_t = W_t$$

สมการ $\phi(B) = 1 - \phi_1 B - ... - \phi_p B^p$ เป็นสมการโพลิโนเมียลลำดับที่ p ของ B คำตอบของสมการเป็นค่าของ B ที่ได้จากการแก้สมการมีจำนวน p ค่า โดยเงื่อนไขที่จะทำให้ตัวแบบอนุกรม

เวลา AR(p) มีคุณสมบัติ Causality ก็คือ รากของสมการ $1-\phi_1 B-...-\phi_p B^p$ ต้องอยู่นอกวงกลม 1 หน่วย(Outside unit circle) กล่าวคือ $\phi(B)=0$ เมื่อ |B|>1 เท่านั้น ตัวอย่างเช่น

- สำหรับรูปแบบ AR(1) คำตอบของสมการ $1-\phi_1 B=0$ คือ $B=\frac{1}{\phi_1}$ เลือกค่า B ที่ |B|>1 หรือ $|\phi|<1$ ซึ่งเป็นเงื่อนไข Stationary สำหรับรูปแบบ AR(1)

- สำหรับรูปแบบ AR(2) คำตอบของสมการ $(1-\phi_1 B-\phi_2 B^2)=0$ จะมี 2 คำตอบ ได้แก่ $B=\frac{\phi_1 \pm \sqrt{\phi_1^2+4\phi_2}}{-2\phi_2}$ เลือกค่า B ที่ |B|>1 หรือ $\phi_2-\phi_1<1$, $\phi_2+\phi_1<1$ และ $|\phi_2|<1$ ซึ่งเป็นเงื่อนไข Stationary สำหรับรูปแบบ AR(2)

- **Invertibility**

Invertibility เป็นเงื่อนไขของพารามิเตอร์ $\theta_1,..,\theta_q$ และเนื่องจากรูปแบบ MA(q) สามารถเขียนได้ในรูปแบบ AR(∞) ทำให้สามารถเขียนค่าความคลาดเคลื่อน(W_t) ในเทอมของค่าสังเกต X_t ได้ พิจารณากระบวนการ MA(1)

$$\begin{aligned}
X_t &= W_t+\theta W_{t-1} \\
W_t &= X_t-\theta W_{t-1} \\
&= X_t-\theta(X_{t-1}-\theta W_{t-2}) \\
&= X_t-\theta X_{t-1}+\theta^2 W_{t-2} \\
&= X_t-\theta X_{t-1}+\theta^2 X_{t-2}-\theta^3 W_{t-3} \\
&\vdots \\
W_t &= \sum_{j=0}^{\infty}(-\theta)^j X_{t-j}
\end{aligned}$$

อย่างไรก็ตามจะมีเฉพาะบางค่าของพารามิเตอร์ $\theta_1,..,\theta_q$ ที่ทำให้หาค่าประมาณ W_t จากค่าสังเกตในอนุกรมเวลาได้ กล่าวคือกระบวนการ MA(1) หากกำหนด $|\theta|<1$ จะทำให้กระบวนการลู่เข้า(converge) หรือค่าพารามิเตอร์(θ)ของกระบวนการ MA(1) ลู่เข้าสู่ 0 กล่าวอีกนัยหนึ่งได้ว่าอิทธิพลของค่าที่ช่วงเวลาก่อนหน้าจะลดลงเมื่อเวลาย้อนหลังไปมากขึ้น โดยจะเรียกค่า $\theta_1,..,\theta_q$ ที่มีคุณสมบัติดังกล่าวว่า “Invertibility”

การพิจารณาค่าของพารามิเตอร์ $\theta_1,..,\theta_p$ ที่ทำให้รูปแบบ MA(q) มีคุณสมบัติ Invertibility ก็คล้ายกับการพิจารณาค่าของพารามิเตอร์ในรูปแบบ AR(p) กล่าวคือ

จากรูปแบบ MA(q) : $X_t=W_t+\theta_1 W_{t-1}+\theta_2 W_{t-2}+...+\theta_q W_{t-q}$ เขียนในรูปแบบ backshift operator ได้เป็น $X_t=\theta(B)W_t=(1+\theta_1 B+\theta_2 B^2+...+\theta_q B^q)W_t$ สมการ

$\theta(B) = 1 + \theta_1 B + ... + \theta_q B^q$ เป็นสมการโพลิโนเมียลลำดับที่ q ของ B คำตอบของสมการเป็นค่าของ B ที่ได้จากการแก้สมการมีจำนวน q ค่า โดยเงื่อนไขที่จะทำให้ตัวแบบอนุกรมเวลา MA(q) มีคุณสมบัติ Invertibility ก็คือ รากของสมการ $1 + \theta_1 B + ... + \theta_q B^q$ ต้องอยู่นอกวงกลม 1 หน่วย (Outside unit circle) กล่าวคือ $\theta(B) = 0$ เมื่อ |B|>1 เท่านั้น ตัวอย่างเช่น

- สำหรับรูปแบบ MA(1) คำตอบของสมการ $1 - \theta_1 B = 0$ คือ $B = \frac{1}{\theta_1}$ เลือกค่า B ที่ |B|>1 หรือ $|\theta_1| < 1$ ซึ่งเป็นเงื่อนไข Invertibility สำหรับรูปแบบ MA(1)

- สำหรับรูปแบบ AR(2) คำตอบชองสมการ $(1 - \theta_1 B - \theta_2 B^2) = 0$ จะมี 2 คำตอบ ได้แก่ $B = \frac{\theta_1 \pm \sqrt{\theta_1^2 + 4\theta_2}}{-2\theta_2}$ เลือกค่า B ที่ |B|>1 หรือ $\theta_2 - \theta_1 < 1$, $\theta_2 + \theta_1 < 1$ และ $|\theta_2| < 1$ ซึ่งเป็นเงื่อนไข Invertibility สำหรับรูปแบบ MA(2)

- **Causality และ Invertibility ของตัวแบบ ARMA(p,q)**

เนื่องจากกระบวนการ ARMA(p,q) เป็นรูปผสมของกระบวนการ AR(p) และ MA(q) ดังนั้น กระบวนการ ARMA(p,q) มีคุณสมบัติ Causality เมื่อสมการ $1 - \phi_1 B - ... - \phi_p B^p = 0$ มีรากของสมการอยู่นอกวงกลมที่มีรัศมีหนึ่งหน่วย และกระบวนการ ARMA(p,q) มีคุณสมบัติ Invertibility เมื่อสมการ $1 + \theta_1 B + ... + \theta_q B^q = 0$ มีรากของสมการอยู่นอกวงกลมรัศมีหนึ่งหน่วย

2.1.8 ขั้นตอนการวิเคราะห์ข้อมูลอนุกรมเวลาด้วยตัวแบบ ARIMA

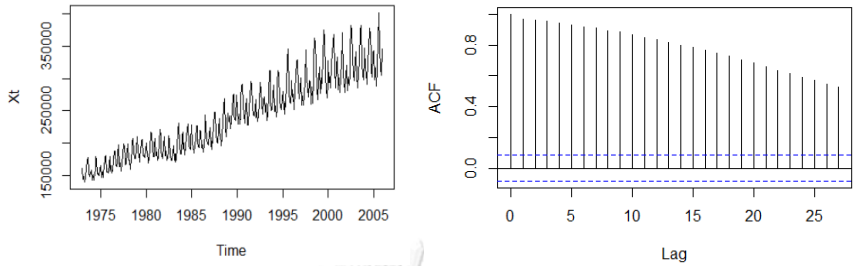
ขั้นที่ 1 : พล็อตกราฟ Time Series เพื่อดูลักษณะของข้อมูล

เมื่อนำข้อมูลมาพล็อตฟกราฟแล้วพบว่าอนุกรมเวลามีคุณสมบัติไม่คงที่(Non-stationary Time Series) จะต้องทำการแปลงอนุกรมเวลาให้คงที่เสียก่อน ซึ่งคุณสมบัติไม่คงที่นั้นอาจเกิดจากอนุกรมเวลามีแนวโน้ม(trend) หรืออนุกรมเวลามีความผันแปร/ความแปรปรวนไม่คงที่ ซึ่งสามารถพิจารณาได้จาก

1) ลักษณะของกราฟข้อมูล เช่น กราฟมีแนวโน้มเชิงเส้นที่เพิ่มขึ้น หรือลดลง(Increasing/ Decreasing trend) หรือกราฟมีลักษณะความแปรปรวนที่เพิ่มขึ้น เป็นต้น ดังแสดงในภาพที่ 2.2 (ก)

2) ฟังก์ชันสหสัมพันธ์ในตัวเองของตัวอย่าง(Sample ACF) หากกราฟ Sample ACF มีลักษณะลดลงค่อนข้างช้าแสดงดังภาพที่ 2.2(ข) แสดงว่าอนุกรมเวลาน่าจะไม่มีคุณสมบัติคงที่

3) การใช้ตัวสถิติทดสอบ “Augmented Dickey Fuller Test(ADF)” หรือเรียกว่าการทดสอบ Unit Root Test หากได้ค่า p-value > α ที่ทำการทดสอบ จะถือว่าอนุกรมเวลาชุดนั้นไม่คงที่



(ก) **(ข)**

ภาพที่ 2.2 (ก) อนุกรมเวลามีลักษณะมีแนวโน้มที่เพิ่มขึ้น และความแปรปรวนไม่คงที่ (ข) Sample ACF มีลักษณะลดลงช้า(Decay)

กรณีอนุกรมเวลาไม่คงที่เนื่องมาจากแนวโน้ม ให้ทำการแปลงอนุกรมเวลาโดยใช้ผลต่าง (Differencing) ส่วนกรณีอนุกรมเวลาไม่คงที่เนื่องมาจากความแปรปรวนไม่คงที่ ให้ทำการแปลงค่าสังเกต(Transformation)ด้วยฟังก์ชันทางคณิตศาสตร์ต่างๆ โดยต้องพิจารณาเลือกใช้ฟังก์ชันทางคณิตศาสตร์ที่เหมาะสมที่ทำให้อนุกรมเวลาใหม่มีความแปรปรวนคงที่ เช่น การใช้ลอการิทึม (Y_t= lnX_t), รากที่สอง ($Y_t=X_t^{1/2}$) เป็นต้น

ขั้นที่ 2 : การระบุแบบจำลอง(Model Identification)

จากการพิจารณาตัวแบบต่างๆที่ผ่านมานั้น จะพบว่าแต่ละตัวแบบมีสหสัมพันธ์ในตัวเอง (ACF) และสหสัมพันธ์ในตัวเองบางส่วน(PACF)ที่แตกต่างกันไปสำหรับตัวแบบที่ต่างกัน ดังนั้น ในการกำหนดตัวแบบที่เหมาะสมให้กับอนุกรมเวลาใดๆจะพิจารณาจากค่าสหสัมพันธ์ในตัวเอง(ACF) และสหสัมพันธ์ในตัวเองบางส่วน(PACF) แต่เนื่องจากอนุกรมเวลาที่นำมาพิจารณาเป็นเพียงตัวอย่างสุ่ม จึงพิจารณาจาก Sample ACF และ Sample PACF แล้วจึงนำมาตัดสินใจเบื้องต้นว่าควรใช้แบบจำลองของ Box-Jenskins แบบใด ซึ่งอาจจะมีตัวแบบที่เหมาะสมมากกว่าหนึ่งตัวแบบ จึงต้องทำการพิจารณาคัดเลือกตัวแบบที่เหมาะสมที่สุด โดยใช้เกณฑ์หรือเงื่อนไขที่รู้จักกันโดยทั่วไป และปรากฏในผลการวิเคราะห์ของโปรแกรมสำเร็จรูปต่างๆ คือ เงื่อนไขของ Akaike(AIC) และเงื่อนไขของ Bayesian(BIC)

ขั้นที่ 3 : การประมาณค่าพารามิเตอร์(Parameter Estimation)

เมื่อได้ตัวแบบ ARIMA(p,d,q) : $(1-\phi_1 B-...-\phi_p B^p)(1-B)^d X_t = (1+\theta_1 B+..+\theta_q B^q)W_t$ ที่เป็นไปได้แล้ว เราจะทราบเพียงค่า p,d,q ที่เหมาะสม แต่ไม่ทราบค่าพารามิเตอร์ $\phi_1,..,\phi_p$ และ $\theta_1,..,\theta_p$ ดังนั้น จะต้องทำการประมาณค่าพารามิเตอร์เหล่านี้ โดยมีวิธีประมาณค่าอยู่หลายวิธี เช่น วิธีกำลังสองน้อยที่สุด วิธีภาวะน่าจะเป็นสูงสุด(MLE)

ขั้นที่ 4 : การตรวจสอบความเหมาะสมของตัวแบบ(Diagnostic checking)

เป็นขั้นตอนที่ทำเพื่อตรวจสอบว่าแบบจำลองที่ประมาณขึ้นในขั้นที่ 2 นั้นมีคุณสมบัติที่เหมาะสมทางสถิติหรือไม่ ถ้าตัวแบบที่เลือกนั้นเหมาะสม ค่าประมาณของพารามิเตอร์ใกล้เคียงกับค่าจริงของพารามิเตอร์ ดังนั้น ส่วนเหลือ(residual) $\varepsilon_t = X_t - \hat{X}_t$ ที่เกิดจากค่าจริงลบด้วยค่าพยากรณ์ควรจะมีคุณสมบัติที่ยอมรับได้ว่า มีการแจกแจงปกติที่ค่าเฉลี่ยเป็นศูนย์และมีความแปรปรวนคงที่ ไม่มีสหสัมพันธ์ในตัวเอง และมีลักษณะสุ่ม หากพบว่าไม่เหมาะสมจะต้องทำการเลือกตัวแบบใหม่ที่เชื่อว่าเหมาะสมกว่า และตรวจสอบความเหมาะสมของตัวแบบจนกว่าจะได้ตัวแบบที่เหมาะสมที่สุดสำหรับอนุกรมเวลาที่พิจารณา

ขั้นที่ 5 : การพยากรณ์(Forecasting)

เมื่อตรวจสอบความเหมาะสมของตัวแบบ ARIMA(p,d,q) แล้วพบว่าตัวแบบมีความเหมาะสมกับข้อมูลอนุกรมเวลาที่พิจารณา ขั้นตอนต่อไปซึ่งเป็นขั้นตอนสุดท้ายของวิธี Box&Jenskins คือการนำสมการพยากรณ์ที่สร้างจากตัวแบบได้ที่ไปหาค่าพยากรณ์ในอนาคต ซึ่งสามารถทำได้ 2 แบบ คือ การพยากรณ์แบบจุด(Point Forecasting) และการพยากรณ์แบบช่วง(Interval Forecasting)

2.2 ตัวแบบ ARIMA ที่มีฤดูกาล

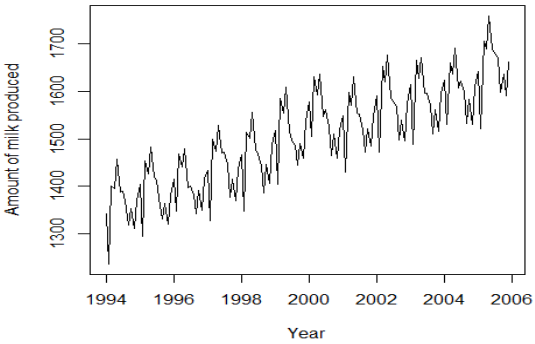
2.2.1 ความแปรผันทางฤดูกาล(Seasonal Variation)

(ภูมิฐาน รังคกูลนุวัฒน์, 2556)กล่าวว่า “ความแปรผันทางฤดูกาล(Seasonal Variation สัญลักษณ์ที่ใช้ S) คือรูปแบบในช่วงเวลาหนึ่งของอนุกรมเวลาที่จะเป็นภายใน 1 ปี และจะเป็นแบบนี้ซ้ำกันทุกปี” ซึ่งอนุกรมเวลาทางเศรษฐศาสตร์ ทางธุรกิจ หรือทางการเงิน ที่มีความถี่เป็นรายเดือนหรือรายไตรมาส อาจจะมีความแปรผันทางฤดูกาลอยู่ด้วย

ระยะเวลาที่สั้นที่สุดที่อนุกรมเวลาจะแสดงให้เห็นว่ามีความผันแปรทางฤดูกาลอีกครั้ง จะเรียกว่า “คาบ(Seasonal period: s) หรือช่วงเวลาของฤดูกาล” เช่น ราคาผลผลิตทางการเกษตรของประเทศหนึ่งในเดือนเมษายนจะสูงกว่าเดือนอื่นๆ และเป็นเช่นนี้ซ้ำกันทุกๆปี ดังนั้น จะมีคาบ s=12 หรือยอดขายเครื่องปรับอากาศในไตรมาสที่ 2 จะสูงกว่าไตรมาสอื่นและเป็นเช่นนี้ทุกๆปี

ดังนั้น จะมีคาบ s=4 ตัวอย่างของข้อมูลอนุกรมเวลาที่มีความผันแปรทางฤดูกาล โดยมีคาบเท่ากับ 12(s=12) แสดงดังรูปภาพที่ 2.3

ส่วนสาเหตุที่ทำให้เกิดความแปรผันทางฤดูกาล คือ สภาพภูมิอากาศ วัฒนธรรม สภาพสังคม หรือเทศกาลต่างๆ เช่น สภาพภูมิอากาศมีผลต่อจำนวนนักท่องเที่ยว หรือเทศกาลปีใหม่จะทำให้ธุรกิจขายของขวัญหรือการ์ดปีใหม่ขยายตัว เป็นต้น(ศิริลักษณ์ สุวรรณวงศ์, 2535)



ภาพที่ 2.3 ลักษณะของข้อมูลอนุกรมเวลาที่มีส่วนประกอบของฤดูกาล(s=12)

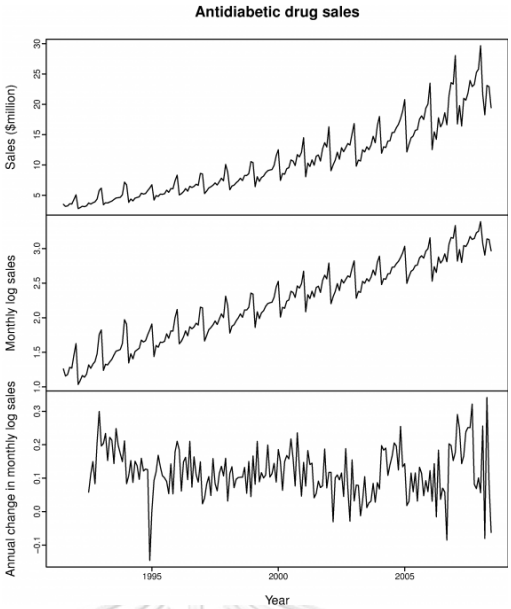
เนื่องจากอนุกรมเวลาที่จะนำไปใช้กับแบบจำลองของ Box-Jenskins ได้นั้น นอกจากจะต้องมีคุณสมบัติคงที่(Stationary)แล้ว ยังต้องไม่มีความแปรผันทางฤดูกาล(Seasonal variation)ด้วย ดังนั้น หากอนุกรมเวลาใดๆที่พิจารณามีความผันแปรทางฤดูกาลอยู่ด้วยจะต้องทำการกำจัดความผันแปรทางฤดูกาลออกไปก่อน ซึ่งวิธีกำจัดความผันแปรทางฤดูกาลนั้นมีหลายวิธี แต่ในที่นี้จะขอกล่าวถึงวิธีที่เป็นที่นิยม และจะนำมาใช้ในการงานวิจัยนี้ คือ การใช้วิธีหาผลต่างของฤดูกาล (Seasonal Differencing)

2.2.2 การกำจัดความผันแปรทางฤดูกาลด้วยวิธีหาผลต่างของฤดูกาล

กำหนดอนุกรมเวลา X_t มีความผันแปรทางฤดูกาล โดยมีช่วงเวลาของฤดูกาล หรือคาบเท่ากับ S การกำจัดความผันแปรทางฤดูกาลสามารถใช้วิธีการหาผลต่างลำดับที่ s ดังนี้

$$\nabla_s X_t = X_t - X_{t-s} = (1-B^s)X_t$$

เช่น หากข้อมูลอนุกรมเวลามีคาบเท่ากับ 12 (s=12) จะสามารถกำจัดความผันแปรทางฤดูกาลโดยใช้สูตร $\nabla_{12} X_t = X_t - X_{t-12}$ หากข้อมูลอนุกรมเวลามีคาบเท่ากับ 4 (s=4) จะสามารถกำจัดความผันแปรทางฤดูกาลโดยใช้สูตร $\nabla_4 X_t = X_t - X_{t-4}$ เป็นต้น ตัวอย่างการใช้ผลต่างของฤดูกาล เพื่อปรับให้ข้อมูลให้คงที่(stationary) แสดงดังภาพที่ 2.4 โดยมีการแปลงข้อมูล (Transformation) โดยใช้ logarithm เพื่อปรับให้ข้อมูลอนุกรมเวลามีความแปรปรวนคงที่ก่อน



ภาพที่ 2.4 การใช้ผลต่างของฤดูกาล(Seasonal Differencing) เพื่อกำจัดความแปรผันทางฤดูกาล(ที่มา : https://www.otexts.org/fpp/8/1)

2.2.3 ตัวแบบ Seasonal Autoregressive Moving Average หรือ SARMA(P,Q)$_s$

2.2.3.1 ตัวแบบ SAR(P)$_s$

เป็นตัวแบบที่แสดงว่าค่าสังเกต X_t ขึ้นอยู่กับค่าสังเกต s , 2s , 3s,.., Ps ช่วงเวลาก่อนหน้า เมื่อ s คือ คาบ(Seasonal period: s) และ P คือ อันดับของรูปแบบ SAR โดยมีรูปแบบสมการคือ

$$\Phi_p(B^s)X_t = W_t$$

หรือเขียนในรูปแบบ

$$X_t = W_t + \Phi_1 X_{t-s} + \Phi_2 X_{t-2s} + ... + \Phi_P X_{t-Ps}$$

เมื่อ $\Phi_p(B^s) = 1 - \Phi_1 B^s - \Phi_2 B^{2s} - ... - \Phi_P B^{Ps}$ คือ seasonal AR characteristic operator

เช่น ตัวแบบ SAR(1)$_{12}$: $X_t = \Phi_1 X_{t-12} + W_t$ เป็นตัวแบบที่ค่าสังเกต X_t ขึ้นอยู่กับค่าสังเกต 12 ช่วงเวลาก่อนหน้า คือ X_{t-12} ตัวแบบ SAR(2)$_{12}$: $X_t = \Phi_1 X_{t-12} + \Phi_1 X_{t-24} + W_t$ เป็นตัวแบบที่ค่าสังเกต X_t ขึ้นอยู่กับค่าสังเกต 12 และ 24 ช่วงเวลาก่อนหน้า คือ X_{t-12} และ X_{t-24}

2.2.3.2 ตัวแบบ SMA(Q)$_s$

เป็นตัวแบบที่แสดงว่าค่าสังเกต X_t ขึ้นอยู่กับค่าความคลาดเคลื่อน s , 2s , 3s,...,Qs ช่วงเวลาก่อนหน้า เมื่อ s คือ คาบ(Seasonal period: s) และ Q คือ อันดับของรูปแบบ SMA คือ โดยมีรูปแบบสมการคือ

$$X_t = \Theta_Q(B^s)W_t$$

หรือเขียนในรูป

$$X_t = W_t + \Theta_1 W_{t-s} + \Theta_2 W_{t-2s} + ... + \Theta_Q W_{t-Qs}$$

เมื่อ $\Theta_Q(B^s) = 1 + \Theta_1 B^s + \Theta_2 B^{2s} + ... + \Theta_Q B^{Qs}$ คือ seasonal MA characteristic operator

เช่น ตัวแบบ $SMA(1)_{12}$: $X_t = W_t + \Theta_1 W_{t-12}$ เป็นตัวแบบที่ค่าสังเกต X_t ขึ้นอยู่กับค่าความคลาดเคลื่อน 12 ช่วงเวลาก่อนหน้า คือ W_{t-12} เป็นต้น

2.2.3.3 ตัวแบบ SARMA(P,Q)$_s$

เป็นตัวแบบที่ผสมผสานระหว่างตัวแบบ $SAR(P)_s$ และ $SMA(Q)_s$ โดยมีรูปแบบสมการดังนี้

$$\Phi_p(B^s)X_t = \Theta_Q(B^s)W_t$$

เมื่อ $\Phi_p(B^s) = 1 - \Phi_1 B^s - \Phi_2 B^{2s} - ... - \Phi_P B^{Ps}$ = seasonal AR characteristic operator

$\Theta_Q(B^s) = 1 + \Theta_1 B^s + \Theta_2 B^{2s} + ... + \Theta_Q B^{Qs}$ = seasonal MA characteristic operator

เมื่อ Q=0 สมการจะลดรูปเหลือเป็นตัวแบบ $SAR(P)_s$ และเมื่อ P=0 สมการจะลดรูปเหลือเป็นตัวแบบ $SMA(Q)_s$

2.2.3.4 ฟังก์ชันสหสัมพันธ์ในตัวเอง(ACF) และฟังก์ชันสหสัมพันธ์ในตัวเองบางส่วน(PACF) ของตัวแบบ SARMA(P,Q)$_s$

ลักษณะของ ACF และ PACF ของตัวแบบ $SARMA(P,Q)_s$ สามารถสรุปได้ดังตารางด้านล่าง (Shumway. & Stoffer., 2010)

	$SAR(P)_s$	$SMA(q)_s$	$SARMA(P,Q)_s$
ACF	มีค่าลดลงอย่างรวดเร็ว (Tail off) ที่ lags ks k=1,2,...	สิ้นสุดหลังจาก lag Qs ช่วงเวลาที่แล้ว	มีค่าลดลงอย่างรวดเร็ว (Tail off) ที่ lags ks
PACF	สิ้นสุดหลังจาก lag Ps ช่วงเวลาที่แล้ว	มีค่าลดลงอย่างรวดเร็ว (Tail off) ที่ lags ks k=1,2,...	มีค่าลดลงอย่างรวดเร็ว (Tail off) ที่ lags ks

ตารางที่ 2.2 ตารางแสดงลักษณะของ ACF และ PACF สำหรับตัวแบบ $SARMA(P,Q)_s$

2.2.4 ตัวแบบ Seasonal Autoregressive Integrated Moving Average :$SARIMA(P,D,Q)_s$

เป็นตัวแบบ Seasonal Autoregressive Moving Average (SARMA) ที่มีการหาผลต่างของฤดูกาล(Seasonal differencing) อันดับที่ D เพื่อแปลงอนุกรมเวลาที่ไม่คงที่(Non-stationary) เนื่องจากฤดูกาล ให้เป็นอนุกรมเวลาใหม่ที่คงที่(stationary) ดังที่กล่าวไปแล้วในหัวข้อ 2.2.2 โดยตัวแบบ SARIMA นั้นมีรูปแบบโมเดลคือ

$$\Phi_p(B^S)\nabla_S^D X_t = \Theta_Q(B^s)W_t$$

เมื่อ $\nabla_s X_t = X_t - X_{t-s} = (1-B^s)X_t$

2.2.5 ตัวแบบ ARIMA(p,d,q)x$SARIMA(P,D,Q)_s$ หรือ ARIMA(p,d,q)x$(P,D,Q)_s$

เป็นอนุกรมเวลาที่มีรูปแบบร่วมระหว่างรูปแบบ ARIMA(p,d,q) และรูปแบบSARIMA(P,D,Q) โดยมีรูปแบบสมการดังนี้

$$\phi(B)\Phi_p(B^s)\nabla^d\nabla_s^D X_t = \theta(B)\Theta_Q(B^s)W_t$$

เมื่อ $\phi(B) = 1-\phi_1 B-\phi_2 B^2-\ldots-\phi_p B^p$ คือ AR characteristic operator

$\theta(B) = 1+\theta_1 B+\theta_2 B^2+\ldots+\theta_q B^q$ คือ MA characteristic operator

$\Phi_p(B^s) = 1-\Phi_1 B^s-\Phi_2 B^{2s}-\ldots-\Phi_P B^{Ps}$ คือ seasonal AR characteristic operator

$\Theta_Q(B^s) = 1+\Theta_1 B^s+\Theta_2 B^{2s}+\ldots+\Theta_Q B^{Qs}$ คือ seasonal MA characteristic operator

$\nabla^d = (1-B)^d$ คือ การหาผลต่าง(Differencing) อันดับที่ d

$\nabla_S^D = (1-B^s)^D$ คือ การหาผลต่างของฤดูกาล(Seasonal Differencing) อันดับที่ D

2.2.6 Causality และ Invertibility ของตัวแบบ ARMA แบบมีฤดูกาล

สำหรับตัวแบบอนุกรมเวลาที่มีปัจจัยเชิงฤดูกาล สามารถพิจารณาคุณสมบัติ Causality และ Invertibility ได้เช่นเดียวกับตัวแบบอนุกรมเวลา ARMA(p,q) โดยการพิจารณาพารามิเตอร์ Φ ของกระบวนการ SAR เช่นเดียวกับพารามิเตอร์ ϕ ของกระบวนการ AR และพิจารณาพารามิเตอร์ Θ ของกระบวนการ SMA เช่นเดียวกับพารามิเตอร์ θ ของกระบวนการ MA ที่กล่าวไปแล้วในหัวข้อ 2.1.7

2.2.7 ขั้นตอนการวิเคราะห์ข้อมูลอนุกรมเวลาด้วยตัวแบบ ARIMA(p,d,q)x(P,D,Q)$_s$

กรณีอนุกรมเวลาไม่คงที่เนื่องจากแนวโน้ม ฤดูกาล และ/หรือความแปรปรวนไม่คงที่ จะต้องทำการแปลงอนุกรมเวลาเดิมให้เป็นอนุกรมเวลาใหม่(ภัทร วรภู)ที่คงที่เสียก่อน ในลักษณะเช่นเดียวกับการวิเคราะห์ข้อมูลอนุกรมเวลาด้วยตัวแบบ ARIMA โดยใช้ผลต่าง(Differencing)กรณีอนุกรมเวลาไม่คงที่เนื่องจากแนวโน้ม ใช้ผลต่างฤดูกาล(Seasonal Differencing)กรณีอนุกรมเวลาไม่คงที่เนื่องจากฤดูกาล และแปลงค่าสังเกต(Transformation)ด้วยฟังก์ชันต่างๆกรณีอนุกรมเวลาไม่คงที่เนื่องจากความแปรปรวนไม่คงที่

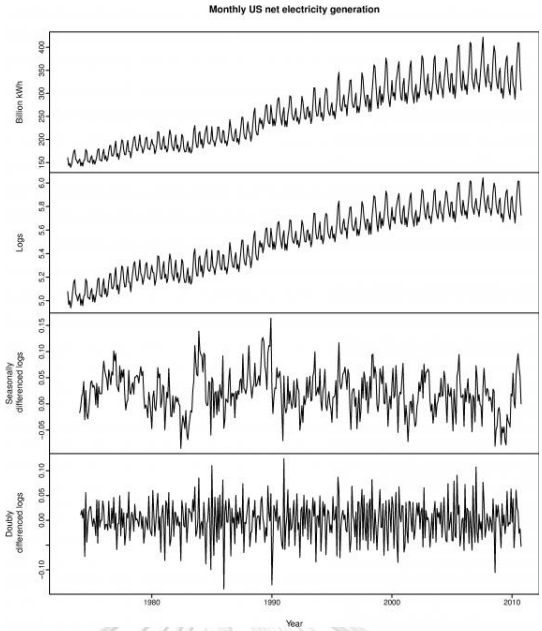
1. กรณีอนุกรมเวลาไม่คงที่เนื่องมาจากทั้ง 3 ส่วนประกอบ คือ ความแปรปรวน ฤดูกาล และแนวโน้มให้ทำการแปลงค่าสังเกตให้มีความแปรปรวนคงที่ก่อน แล้วจึงค่อยทำการแปลงอนุกรมเวลาโดยใช้ผลต่างและผลต่างฤดูกาล แต่หากอนุกรมเวลาไม่คงที่เนื่องมาจากแนวโน้ม และฤดูกาลให้ทำการหาผลต่าง และผลต่างฤดูกาลโดยทำอย่างใดก่อนก็ได้จะให้ผลลัพธ์แบบเดียวกัน(มุกดา แม้นมินทร์, 2549) ตัวอย่างการแปลงอนุกรมเวลาให้คงที่โดยการใช้การแปลงข้อมูล(Transformation) การใช้ผลต่างฤดูกาล(Seasonal Differencing) และการใช้ผลต่าง(Differencing) แสดงดังภาพที่ 2.5

2. กำหนดตัวแบบ ARMA(p,q)xSARMA(P,Q)$_s$ ที่เหมาะสมให้กับอนุกรมเวลาใหม่ที่คงที่แล้วโดยพิจารณารูปแบบ ARMA(p,q) จากค่า Sample ACF : $\hat{\rho}(h)$ และ Sample PACF : $\hat{\phi}_{hh}$ สำหรับ h=1,2,... และพิจารณารูปแบบ SARMA(P,Q)$_s$ จากค่า $\hat{\rho}(h)$ และ $\hat{\phi}_{hh}$ สำหรับ h=s,2s,3s,...

3. ประมาณค่าพารามิเตอร์ในรูปแบบ ARMA(p,q)xSARMA(P,Q)$_s$ จากตัวแบบที่ได้ในข้อ 2 ทำได้เช่นเดียวกันกับการประมาณค่าพารามิเตอร์ของข้อมูลอนุกรมเวลาด้วยตัวแบบ ARIMA ที่ได้กล่าวไว้ในหัวข้อที่ 2.1.8

4. ตรวจสอบความเหมาะสมของตัวแบบ ทำได้เช่นเดียวกับตรวจสอบความเหมาะสมของตัวแบบของข้อมูลอนุกรมเวลาด้วยตัวแบบ ARIMA ที่ได้กล่าวไว้ในหัวข้อที่ 2.1.8

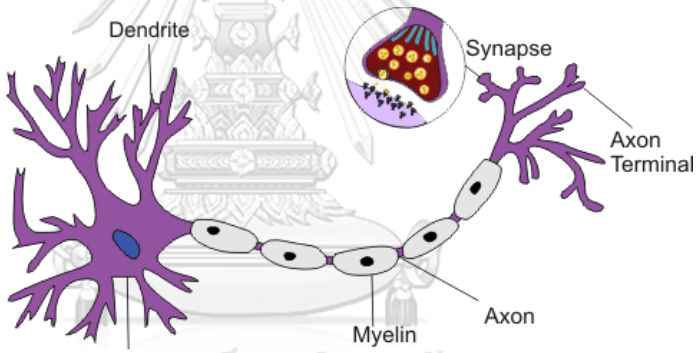
5. สร้างสมการพยากรณ์จากตัวแบบที่เหมาะสมที่สุดที่คัดเลือกได้



ภาพที่ 2.5 ภาพแสดงการปรับข้อมูลอนุกรมเวลาให้คงที่โดยใช้การแปลงข้อมูล ผลต่างฤดูกาล และผลต่าง (ที่มา : https://www.otexts.org/fpp/8/1)

2.3 ตัวแบบโครงข่ายประสาทเทียม(Artificial neural network model: ANN)

ตัวแบบโครงข่ายประสาทเทียมเป็นโมเดลทางคณิตศาสตร์เพื่อจำลองการทำงานของเครือข่ายประสาทในสมองมนุษย์ โดยมีวัตถุประสงค์ที่จะสร้างเครื่องมือซึ่งมีความสามารถในการเรียนรู้การจดจำรูปแบบ(Pattern Recognition)เช่นเดียวกับความสามารถที่มีในสมองมนุษย์ แนวคิดเริ่มต้นของเทคนิคนี้ได้มาจากการศึกษาโครงข่ายไฟฟ้าชีวภาพ(Bioelectric Network) ในสมองซึ่งประกอบด้วยเซลล์ประสาท หรือ "นิวรอน" (Neurons) หลายๆเซลล์ โดยมีจุดเชื่อมระหว่างเซลล์ประสาท เรียกว่า “ซิแนปส์”(Synapses) แต่ละเซลล์ประสาทจะประกอบด้วยส่วนที่รับกระแสประสาทจากเซลล์อื่นเข้ามา เรียกว่า "เดนไดรท์" (Dendrite) ซึ่งเปรียบเสมือน input ของเซลล์ และส่วนที่ส่งกระแสประสาทออกไปเรียกว่า "แอคซอน" (Axon) ซึ่งเปรียบเสมือน output ของเซลล์ โดยกระแสประสาทต่างๆจะผ่านจากเดนไดรท์เข้าสู่เซลล์ประสาท และหากสัญญาณรวมมีความแรงเกินค่าระดับ(Threshold) ของเซลล์ประสาทนั้น เซลล์ประสาทก็จะส่งสัญญาณออกทางแอคซอนต่อไป



ภาพที่ 2.6 เซลล์ประสาทในสมองมนุษย์(Neuron)

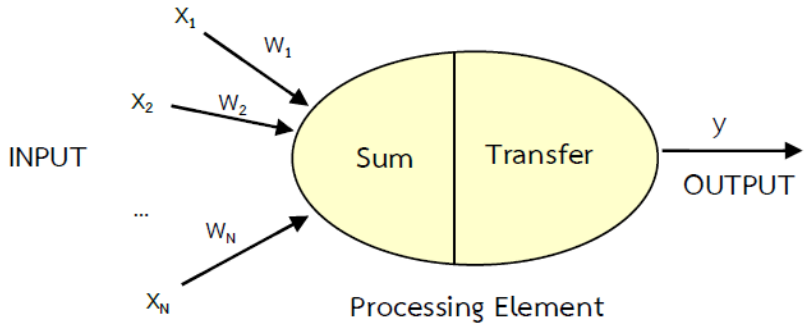
(ที่มา : http://www.4mylearn.org/Neurobiology.html)

โครงข่ายประสาทเทียมก็มีคุณลักษณะคล้ายกับการส่งผ่านสัญญาณประสาทในสมองของมนุษย์ กล่าวคือมีความสามารถในการรวบรวมความรู้(Knowledge) โดยผ่านกระบวนการเรียนรู้(Learning process) และความรู้เหล่านี้จะจัดเก็บอยู่ในโครงข่ายในรูปแบบค่าถ่วงน้ำหนัก(Weight) ซึ่งสามารถปรับเปลี่ยนค่าได้เมื่อมีการเรียนรู้สิ่งใหม่ๆเข้าไป ค่าถ่วงน้ำหนักทำหน้าที่เปรียบเสมือนความรู้ที่รวบรวมไว้เพื่อใช้ในการแก้ปัญหาเฉพาะอย่างของมนุษย์(ชญานิน บุญมานะ และ นัท กุลวานิช, 2560)

ข้อดีของโครงข่ายประสาทเทียมคือ สามารถพยากรณ์ข้อมูลส่วนที่ไม่เป็นฟังก์ชันเชิงเส้นตรงได้ดี อีกทั้งยังมีความแกร่ง(Robust)เมื่อมีข้อมูลรบกวน(Noisy data) เช่น มีค่านอกเกณฑ์(Outlier) เนื่องจากโครงข่ายประสาทเทียมมีโหนดจำนวนมาก เมื่อมีการถ่วงน้ำหนักเพื่อเชื่อมต่อกับโหนดต่างๆ โครงข่ายประสาทเทียมจะสามารถเรียนรู้ในการทำงานในชุดข้อมูลที่มีความคลาดเคลื่อนได้ดี ส่วนข้อเสียของโครงข่ายประสาทเทียม คือ การแปลความหมายที่เข้าใจยาก และใช้ระยะเวลาในการฝึกหัดนานหลายชั่วโมง(ชญานิน บุญมานะ และ นัท กุลวานิช, 2560)

2.3.1 หลักการทำงานของโครงข่ายประสาทเทียม

การประมวลผลของโครงข่ายประสาทเทียมจะเกิดขึ้นในหน่วยประมวลผลย่อย(Node) ซึ่งเปรียบเสมือนเป็นการจำลองลักษณะการทำงานของเซลล์ประสาทในสมองของมนุษย์ โดยเมื่อมีข้อมูลนำเข้า(Input Data) เข้ามายังโครงข่าย ข้อมูลนำเข้าแต่ละค่าจะถูกนำมาคูณกับค่าถ่วงน้ำหนัก(weight)แทนด้วยสัญลักษณ์ W_n จากนั้นผลลัพธ์ที่ได้จากการคูณค่าข้อมูลนำเข้ากับค่าถ่วงน้ำหนักจะถูกนำมารวมกันแล้วส่งผ่านไปยังฟังก์ชันการแปลง(Transfer function) หรือเรียกว่า “ฟังก์ชันกระตุ้น(Activation function)” เพื่อที่จะหาข้อมูลผลลัพธ์(Output Data)ออกมาและส่งไปยังหน่วยประมวลผลย่อยถัดไป ดังภาพที่ 2.7



ภาพที่ 2.7 ภาพแสดงหลักการทำงานของโครงข่ายประสาทเทียม

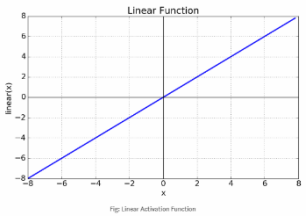
โดยฟังก์ชันกระตุ้น(Activation function)นั้นมีหลายประเภท เช่น Sigmoid logistic function , linear functions, Hyperbolic tangent function *เป็นต้น*

2.3.2 ประเภทของ Activation Function/Transfer Function

Activation Function หรือ Transfer Function นั้นจะมีอยู่ 2 ที่ คือ 1.Hidden neurons – เพื่อทำหน้าที่แปลงค่าข้อมูลนำเข้าจาก Input neurons ให้เป็นข้อมูลส่งออกไปยัง Output neuron และ 2.Output neuron - ทำหน้าที่แปลงค่าข้อมูลนำเข้าจาก Hidden neurons ให้เป็นข้อมูลส่งออกที่สามารถนำไปใช้ได้ ซึ่งก็คือค่าพยากรณ์นั่นเอง โดยประเภทของ Activation Function จะแบ่งได้เป็น 2 ประเภทใหญ่ๆ ดังนี้

- **Linear Activation Function**

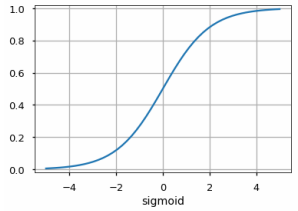
  - ***Linear functions หรือ Identity function***



$f(x) = x$ เป็นฟังก์ชันที่อยู่ใน Output neuron ซึ่งเป็นชั้นสุดท้ายของโครงข่าย ใช้ในการพยากรณ์ค่าที่เป็นตัวเลข การประมาณค่า การพยากรณ์อนุกรมเวลา เป็นต้น

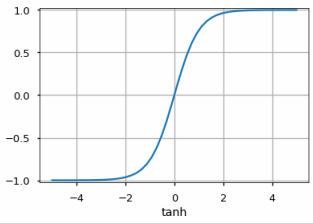
- **Non-linear Activation Functions**

  - **Sigmoid/ Logistic Activation Function**



$f(x) = \frac{1}{1+e^{-x}}$ ทำการแปลงสถานะของข้อมูลนำเข้าให้อยู่ในรูปแบบการจัดแบ่งเชิงกลุ่ม เช่น 0 หรือ 1, ใช่ หรือ ไม่ใช่ เป็นต้น ใช้ในทุก Neurons ใน Hidden layer ยกเว้น Output neuron ค่าที่ได้หลังจากผ่านฟังก์ชันนี้จะอยู่ระหว่าง 0 ถึง 1

  - **Hyperbolic tangent function หรือ Tan-sigmoid function**



$f(x) = \tanh(x) = \frac{\sinh(x)}{\cosh(x)} = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$ คล้ายกับ **Logistic Activation Function** แต่ค่าที่ได้หลังจากผ่านฟังก์ชันนี้จะอยู่ระหว่าง -1 ถึง 1

งานวิจัยชิ้นนี้เลือกใช้ Sigmoid/Logistic Function เป็น Activation Function ในชั้น Hidden Layer เนื่องจากราคาขายปลีกมะนาวจะมีค่ามากกว่า 0 เสมอ การใช้ Sigmoid/Logistic Activation Function นั้นจะทำให้ได้ค่าที่อยู่ระหว่าง 0 กับ 1 ซึ่งไม่เป็นค่าติดลบสอดคล้องกับข้อมูลราคาขายปลีกมะนาว อีกทั้งยังเป็น Activation Function ที่นิยมใช้มากที่สุด ซึ่งให้ผลการพยากรณ์ที่แม่นยำ(Karlik & Olgac)

2.3.3 โครงข่ายประสาทเทียมเพอร์เซ็ปตรอนหลายชั้น (Multi-layer perceptron: MLP)

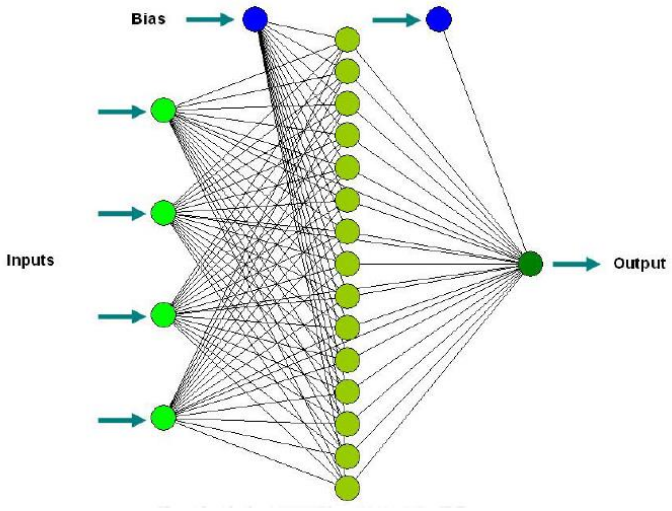
หรือเรียกอีกอย่างหนึ่งว่า โครงข่าย Feedforward neural network เป็นตัวแบบประเภทการพยากรณ์(prediction) ที่นิยมใช้พยากรณ์ข้อมูลอนุกรมเวลา โดยมีโครงสร้าง 3 ชั้น คือ

1. Input Layer มี 1 ชั้น ซึ่งเป็นชั้นที่รับข้อมูลนำเข้า(Input Data) ประกอบด้วย Input neurons โดยจำนวน Input neurons ขึ้นอยู่กับจำนวนตัวแปรอิสระที่ใช้ในการสร้างตัวแบบ

2. Hidden layer มีอย่างน้อย 1 ชั้นขึ้นไป ประกอบด้วย Hidden neurons ไม่มีข้อจำกัดเกี่ยวกับจำนวน Hidden Layer ว่าควรมีจำนวนเท่าไหร่ แต่ปกติจะใช้เพียง 1 หรือ 2 ชั้นเท่านั้น ส่วนจำนวน Hidden neurons นั้นผู้วิจัยต้องทำการทดลองหาจำนวนที่เหมาะสมเอง บ้างกำหนดให้ใช้ 2n+1 หรือ 2n (G. Zhang, Patuwo, & Hu., 1988) โดยที่ n คือจำนวน input node เป็นต้น

3. Output layer มี 1 ชั้น เป็นชั้นสุดท้ายที่ส่งข้อมูลผลลัพธ์(Output Data)ออกไปซึ่งได้ค่าพยากรณ์ไปใช้จริง ประกอบด้วย Output neurons หากเป็นเรื่อง Time series จำนวน Output neurons จะเท่ากับจำนวนค่าพยากรณ์ล่วงหน้าที่ต้องการทำนาย เช่น 1 neuron กรณีพยากรณ์ค่าล่วงหน้าค่าเดียว 2 neuron กรณีต้องการพยากรณ์ 3 ค่า และ 5 ค่าล่วงหน้า หากเป็นเรื่องการพยากรณ์เชิงกลุ่ม จำนวน Output neurons จะเท่ากับจำนวนกลุ่มที่เราตั้งค่าไว้ เป็นต้น (ภัทร วรภู, 2556)

ค่า Inputs ที่เข้ามายังโครงข่ายประสาทเทียม MLP จะถูกส่งผ่านจาก Input neurons ไปยัง Hidden neurons แบบทั่วถึงกันหมด (Fully connected) จากนั้นจึงเข้าสู่ Output neurons ในชั้นสุดท้าย ดังภาพที่ 2.8

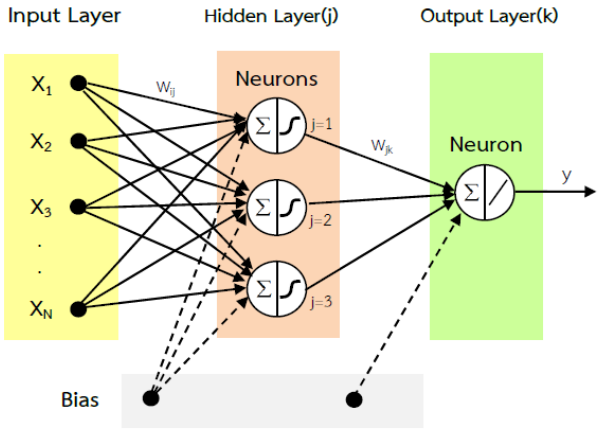


ภาพที่ 2.8 โครงข่ายประสาทเทียมเพอร์เซ็ปตรอนหลายชั้น

(ที่มา : ภัทร วรภู, 2556)

<u>การทำงานของโหนดในชั้นซ่อน และชั้นผลลัพธ์</u>

กำหนดข้อมูลนำเข้า $x=(x_1,...,x_N)$ เพื่อใช้ในการคำนวณหาค่า Output ของโครงข่ายประสาทเทียม



ภาพที่ 2.9 การทำงานของโหนดในชั้นซ่อน และชั้นผลลัพธ์

<u>พิจารณาโหนดในชั้นซ่อน</u>

- คำนวณค่าผลรวมของโหนดในชั้นซ่อน โดยนำค่าของข้อมูลนำเข้ามาคูณกับค่าถ่วงน้ำหนักในแต่ละเส้นเชื่อมโยงที่เข้ามายังโหนดที่ j แล้วนำมารวมกัน ดังนั้น ค่าผลรวมของโหนดที่ j ในชั้นซ่อน คือ

$$net_j = \sum_{i=1}^{N} x_i w_{ij} + bias$$

เมื่อ X_i แทนข้อมูลนำเข้าที่ i , i=1,..,N

W_{ij} แทนค่าถ่วงน้ำหนักของเส้นเชื่อมโยงกับข้อมูลนำเข้าที่ i ไปยังโหนดที่ j , j=1,..,J

- ทำการปรับค่าผลรวมด้วย Sigmoid/logistic Activation Function ซึ่งจะได้ค่าของโหนดในชั้นซ่อนที่อยู่ในช่วง (0,1)

$$y_j = f(net_j) = \frac{1}{1+e^{-net_j}}$$

<u>พิจารณาโหนดในชั้นผลลัพธ์</u>

- คำนวณค่าผลรวมของโหนดในชั้นผลลัพธ์(Output Layer) ซึ่งมี k โหนด

$$net_k = \sum_{j=1}^{J} y_j w_{jk} + bias$$

เมื่อ w_{jk} แทนค่าถ่วงน้ำหนักของเส้นเชื่อมโยงจากโหนดที่ j ในชั้น hidden layer ไปยังโหนดที่ k ในชั้น Output layer , k=1,..,K

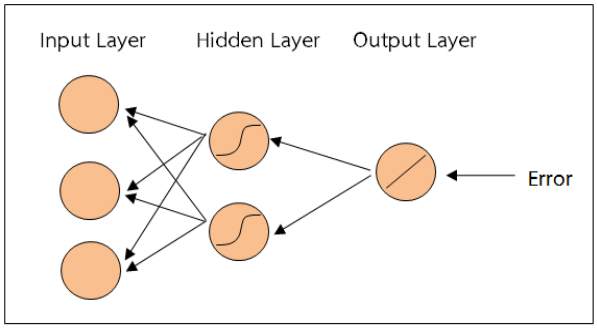
- ทำการปรับค่าผลรวมด้วยฟังก์ชันกระตุ้นโดยใช้ Linear Activation Function ซึ่งจะได้ค่าของโหนดในชั้น Output

$$y_k = f(net_k) = net_k$$

2.3.4 อัลกอริทึมในการเรียนรู้

2.3.4.1 เทคนิคการแพร่แบบย้อนกลับ(Back-propagation)

การฝึกสอน(Training)โครงข่ายประสาทเทียมเป็นกระบวนการในการปรับปรุงค่า weight เพื่อให้ได้ข้อมูลผลลัพธ์(Output Data)ใกล้เคียงกับข้อมูลเป้าหมาย(Target Data) (Riedmiller & Braun, 1993)กล่าวว่าอัลกอริทึมที่นิยมใช้ในการเทรนโครงข่ายประสาทเทียมแบบ MLP ก็คือเทคนิค “Back-propagation: BP” หรือเทคนิคการแพร่แบบย้อนกลับ



ภาพที่ 2.10 เทคนิคการแพร่แบบย้อนกลับ(Back-propagation)

หลักการของเทคนิคนี้คือเมื่อใส่ข้อมูลนำเข้า(Input Data)สำหรับฝึกสอนให้แก่โครงข่ายแล้ว ข้อมูลผลลัพธ์(Output Data)จากโครงข่ายจะถูกนำไปเปรียบเทียบกับข้อมูลเป้าหมาย(Target Data) แล้วทำการคำนวณหาค่าความผิดพลาด ซึ่งค่าความผิดพลาดหรือความคลาดเคลื่อน(Error)นี้จะถูกส่ง/แพร่กลับเข้าสู่โครงข่ายจากชั้นผลลัพธ์ย้อนกลับมายังชั้นข้อมูลนำเข้า เพื่อใช้แก้ไขค่าน้ำหนักที่เชื่อมระหว่างโหนดต่างๆต่อไปจนกระทั่งได้น้ำหนัก(weight)ที่ทำให้ฟังก์ชันความคลาดเคลื่อน(Error function:E) ของกระบวนการมีค่าน้อยที่สุด(G. Zhang et al., 1988) โดยก่อนที่จะทำการสอนโครงข่ายประสาทเทียมแบบหลายชั้นนั้นจะมีการกำหนดค่าเริ่มต้นให้กับค่าถ่วงน้ำหนักที่เชื่อมโยงระหว่างชั้นทุกชั้น โดยค่านี้ได้มาจากการสุ่ม(Randomness) และเมื่อเข้าสู่กระบวนการฝึกสอนโครงข่าย ค่าถ่วงน้ำหนักใหม่จะถูกปรับดังสมการด้านล่าง

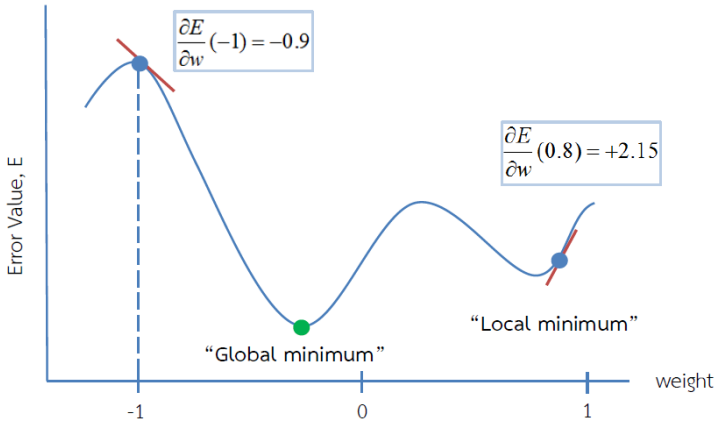
$$w_{ij}(t+1) = w_{ij}(t) - \eta \frac{\partial E}{\partial w_{ij}}(t)$$

โดย w_{ij} คือ ค่าถ่วงน้ำหนักจากneuron j ไปยัง neuron i

η คือ อัตราการเรียนรู้(learning rate)

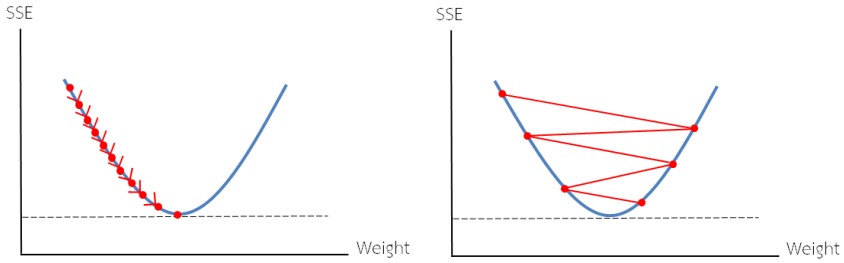
$\frac{\partial E}{\partial w_{ij}}(t)$ คือ partial derivative ของฟังก์ชันค่าความคลาดเคลื่อนเทียบกับค่าถ่วงน้ำหนัก

อัตราการเรียนรู้(η) เป็นพารามิเตอร์ใน BP ที่ผู้วิจัยต้องระบุค่า ซึ่งจะส่งผลต่อประสิทธิภาพของเทคนิค BP อย่างมาก เนื่องจากเทคนิค BP นั้นจะใช้ขนาด(magnitude) ของ partial derivative ในการปรับค่าถ่วงน้ำหนักด้วย ตัวอย่างเช่น หากค่า partial derivative ของฟังก์ชันค่าคลาดเคลื่อนเมื่อค่าถ่วงน้ำหนักปัจจุบันเท่ากับ -1 มีค่าเท่ากับ -0.9 ค่าถ่วงน้ำหนักใหม่จะเท่ากับ $w_{ij}(t+1) = -1 + 0.9\eta$ ซึ่งถ้ากำหนดค่าอัตราเรียนรู้ที่มีค่ามาก เช่น η=2 จะทำให้ค่าถ่วงน้ำหนักใหม่ไปอยู่ที่ 0.8 ซึ่งเลยจุดที่เป็น Global minimum ของฟังก์ชันความคลาดเคลื่อน ดังภาพด้านล่าง



ภาพที่ 2.11 ตัวอย่างการปรับค่าถ่วงน้ำหนักเมื่อกำหนดอัตราการเรียนรู้ η=2

กล่าวคือการกำหนดอัตราการเรียนรู้ที่มากเกินเกินไปอาจทำให้เกิด oscillation ทำให้ไม่ได้ค่าน้ำหนักที่ทำให้คลาดเคลื่อน(Error) ของกระบวนการที่ต่ำสุดเสียที ซึ่งกระบวนการนั้นอาจไม่ลู่เข้าเลยก็เป็นไปได้(Riedmiller & Braun, 1993) ดังนั้น การฝึกสอนโครงข่ายประสาทเทียมโดยทั่วไปจะกำหนดค่า learning rate ให้มีค่าน้อยๆ เพื่อให้การฝึกสอนนั้นมีประสิทธิภาพมาก แต่กระบวนการ optimization จะใช้เวลานานมากเช่นกัน เนื่องจากแต่ละ step การเรียนรู้จนกว่าจะได้ค่า weight ที่ต่ำที่สุดนั้นมีค่าน้อย ทำให้ใช้เวลานานกว่ากระบวนการนั้นจะลู่เข้า(Converge) ดังภาพที่ 2.12



ภาพที่ 2.12 ลักษณะของการกำหนดอัตราการเรียนรู้ที่ต่อระยะเวลาในการเรียนรู้ของโครงข่าย

2.3.4.2 เทคนิค Resilient back propagation(Rprop)

ในปี 1993 Martin Riedmiller และ Heinrich Braun ได้คิดค้นอัลกอริทึมใหม่ขึ้นมาเรียกว่า “Resilient back propagation: Rprop” ซึ่งพัฒนามาจากเทคนิค Back-Prop เดิม ซึ่งเทคนิค Rprop นั้นไม่ต้องใช้ขนาด(magnitude)ของ partial derivative ในการปรับค่าถ่วงน้ำหนัก แต่จะใช้เพียงเครื่องหมาย(sign)ในการปรับค่า โดยมีข้อดีหลักๆอยู่ 2 ประการคือ

1. การฝึกสอนโครงข่ายประสาทเทียมด้วย Rprop จะใช้เวลาน้อยกว่า BP
2. Rprop ไม่จำเป็นต้องระบุค่า Learning rate ดังเช่น เทคนิค Back-prop

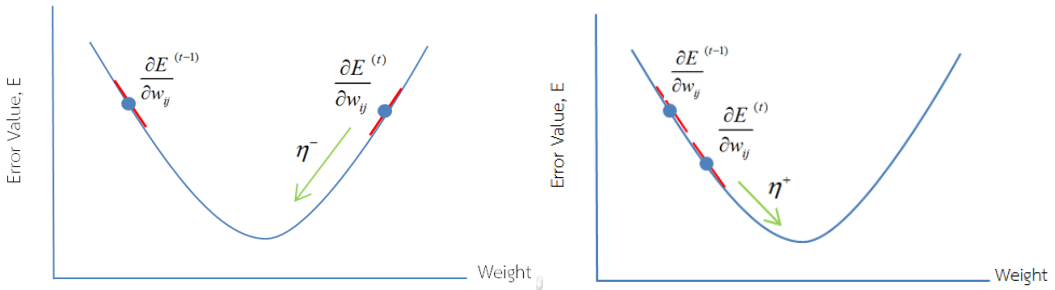
หลักการของเทคนิค Rprop คือ ในขั้นตอนเริ่มต้นแต่ละ weight จะมีค่า update value(Δ_{ij}) เป็นของตนเอง (โดยทั่วไปจะถูกกำหนดให้ initial update value Δ_{ij} (Δ_0) =0.1 ซึ่งความมาก-น้อยของพารามิเตอร์นี้ไม่ได้มีผลกระทบต่อความเร็วในการลู่เข้า)(Riedmiller & Braun, 1993) จากนั้นค่า update value ใหม่($\Delta_{ij}^{(t)}$) จะถูกปรับค่าดังนี้

$$\Delta_{ij}^{(t)} = \begin{cases} \eta^+ \cdot \Delta_{ij}^{(t-1)} & , \text{ ถ้า } \dfrac{\partial E}{\partial w_{ij}}^{(t-1)} \cdot \dfrac{\partial E}{\partial w_{ij}}^{(t)} > 0 \\ \eta^- \cdot \Delta_{ij}^{(t-1)} & , \text{ ถ้า } \dfrac{\partial E}{\partial w_{ij}}^{(t-1)} \cdot \dfrac{\partial E}{\partial w_{ij}}^{(t)} < 0 \\ \Delta_{ij}^{(t-1)} & , \text{ กรณีอื่น} \end{cases}$$

โดย $0 < \eta^- < 1 < \eta^+$

กล่าวคือหากค่า partial derivative ของค่าถ่วงน้ำหนัก w_{ij} เปลี่ยนเครื่องหมายนั่นหมายความว่าการปรับค่าถ่วงน้ำหนักครั้งที่แล้วมากจนเกินค่าถ่วงน้ำหนักที่ทำให้ฟังก์ชันความ

คลาดเคลื่อนต่ำที่สุด ค่า update value(Δ_{ij}) จะถูกปรับ(scale)ขนาดลดลงด้วย η^- ในทางตรงกันข้ามหากค่า partial derivative ของค่าถ่วงน้ำหนัก w_{ij} ไม่เปลี่ยนเครื่องหมายแสดงว่าค่าถ่วงน้ำหนักนั้นยังไม่ใช่ค่าที่ทำให้ฟังก์ชันความคลาดเคลื่อนต่ำที่สุด จึงถูกปรับเพิ่มขนาดด้วย η^+



ภาพที่ 2.13 ภาพแสดงการปรับค่า update value(Δ_{ij})

เมื่อ update value ใหม่($\Delta_{ij}^{(t)}$) ของทุก weight ถูกปรับค่าแล้ว จะมีการกำหนดเครื่องหมาย(sign)ให้กับ $\Delta_{ij}^{(t)}$ เพื่อปรับทิศทางให้ถูกต้อง ซึ่งค่าเปลี่ยนแปลงค่าถ่วงน้ำหนัก(weight update: $\Delta w_{ij}^{(t)}$) ณ ตำแหน่งปัจจุบันจะถูกปรับดังสมการ

$$\Delta w_{ij}^{(t)} = \begin{cases} -\Delta_{ij}^{(t)} & , \text{ ถ้า } \frac{\partial E}{\partial w_{ij}}^{(t)} > 0 \\ +\Delta_{ij}^{(t)} & , \text{ ถ้า } \frac{\partial E}{\partial w_{ij}}^{(t)} < 0 \\ 0 & , \text{ กรณีอื่น} \end{cases}$$

สุดท้ายค่าถ่วงน้ำหนักใหม่ของแต่ละ weight จะถูกปรับโดยการนำค่าถ่วงน้ำหนักเดิมบวกด้วยค่าเปลี่ยนแปลงค่าถ่วงน้ำหนัก(weight update: $\Delta w_{ij}^{(t)}$) ซึ่งเป็นไปดังสมการด้านล่าง

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} + \Delta w_{ij}^{(t)}$$

Note

โดยทั่วไปจะกำหนด ค่า $\eta^- = 0.5$ และ $\eta^+ = 1.2$ (Riedmiller & Braun, 1993) ซึ่งเป็น Default ของโปรแกรมสำเร็จรูป R ซึ่งจะนำมาใช้ในงานวิจัยนี้

2.3.5 การแปลงข้อมูลก่อนและหลังกระบวนการเรียนรู้(Data preprocessing and postprocessing)

ข้อมูลดิบที่นำมาวิเคราะห์ควรจะถูกทำการแปลงก่อนนำเข้าสู่กระบวกการฝึกฝน เนื่องจากข้อมูลดิบนั้นอาจจะมีข้อมูลบางค่าที่เป็นค่านอกเกณฑ์(outlier) เป็นต้น เพื่อทำให้การสร้างตัวแบบใช้เวลาน้อยลง และทำให้ค่าพยากรณ์ที่ได้มีประสิทธิภาพมากยิ่งขึ้น ซึ่งวิธีการแปลงข้อมูลนั้นมีอยู่หลายวิธี เช่น การแปลงข้อมูลให้อยู่ในรูปปรกติมาตรฐานน้อยที่สุด-มากที่สุด(Min-Max normalization), การแปลงข้อมูลให้อยู่ในรูปคะแนน Z (Zscore standardization), การแปลงข้อมูลให้อยู่ในช่วง -1 ถึง 1 เป็นต้น(ชญานิน บุญมานะ และ นัท กุลวานิช, 2560)

ในการวิจัยนี้เลือกใช้วิธีการแปลงข้อมูลให้อยู่ในรูปปรกติมาตรฐานน้อยที่สุด-มากที่สุด (Min-Max normalization) เนื่องจากการแปลงข้อมูลในลักษณะนี้ข้อมูลที่ถูกแปลงแล้วจะมีค่าอยู่ระหว่าง 0 ถึง 1 ซึ่งจะสอดคล้องกับราคาขายปลีกมะนาวซึ่งมีค่าเป็นบวกเสมอ หากใช้การแปลงข้อมูลด้วยวิธีอื่นอาจทำให้ราคาขายปลีกมะนาวมีค่าติดลบได้

- **การแปลงข้อมูลให้อยู่ในรูปปรกติมาตรฐานน้อยที่สุด-มากที่สุด(Min-Max normalization)**

เป็นการแปลงข้อมูลจากช่วงที่เป็นไปได้เดิมของค่า input ให้เป็นช่วงข้อมูลใหม่ที่มีค่าอยู่ในช่วง 0-1

$$z_i = \frac{x_i - \min(x)}{\max(x) - \min(x)}$$

เมื่อ $x = (x_1, ..., x_n)$

z_i คือ ข้อมูลตัวที่ i ที่ถูกทำ normalization แล้ว

หมายเหตุ ภายหลังจากการทำการฝึกฝนตัวแบบโครงข่ายประสาทเทียมแล้ว จะต้องทำการแปลงข้อมูลที่ได้กลับไปให้อยู่ในช่วงเดิมของค่า input ตอนแรก(Denormalization) เพื่อที่จะสามารถนำค่าไปใช้ในการพยากรณ์ต่อไปได้

2.3.6 ตัวแบบ ANN สำหรับการพยากรณ์อนุกรมเวลา

ตัวแบบ ANN สามารถนำมาประยุกต์ใช้กับการพยากรณ์อนุกรมเวลาได้ โดยที่ตัวแปรที่ใช้ในการพยากรณ์นั้นเป็นเพียงตัวแปรเดียวซึ่งเรียกว่า “Univariate time series” เช่น ต้องการพยากรณ์ระคามะนาวในเดือนถัดไป ก็นำตัวแปรราคามะนาว ณ เวลาย้อนหลังของเดือนก่อนๆหน้ามาใช้เป็นตัวแปรอิสระในการพยากรณ์

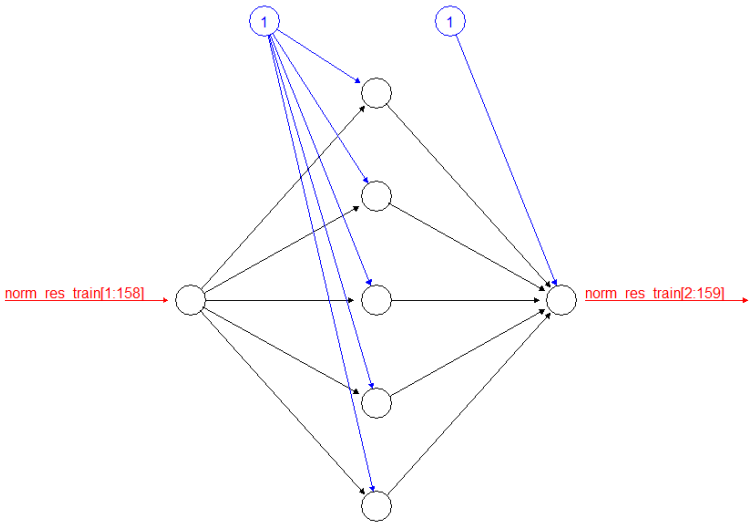
(G. Zhang et al., 1988)กล่าวไว้ว่าสมมติว่ามีข้อมูลอนุกรมเวลาอยู่ N ค่า $y_1,y_2,...,y_N$ ในชุดข้อมูลฝึกสอน(Training set) และต้องการพยากรณ์ 1 ช่วงเวลาไปข้างหน้า(1-step-ahead) คือ Y_{N+1} เมื่อกำหนดจำนวน lag หรือจุดเวลาย้อนหลัง เช่น lag=n=1 คือทำการพยากรณ์ Y_{N+1} ด้วย Y_N , lag=n=2 คือทำการพยากรณ์ Y_{N+1} ด้วย Y_N และ Y_{N-1} ดังนั้น จะมีจำนวนรูปแบบการฝึกสอน (Training pattern) หรือชุดของข้อมูลนำเข้า(Input) เท่ากับ N-n ชุด รูปแบบการฝึกสอนที่ 1 จะประกอบด้วยข้อมูลนำเข้าคือ $y_1,y_2,...,y_n$ ซึ่งจะได้ข้อมูลเป้าหมายคือ y_{n+1} รูปแบบการฝึกสอนที่2 จะประกอบด้วยข้อมูลนำเข้าคือ $y_2,y_3,...,y_{n+1}$ ซึ่งจะได้ข้อมูลเป้าหมายคือ y_{n+2} ท้ายที่สุดจะได้รูปแบบการฝึกสอนสุดท้ายคือรูปแบบการสอนที่ N-n ซึ่งประกอบด้วยข้อมูลนำเข้าคือ $y_{N-n},y_{N-n+1},...,y_{N-1}$ และข้อมูลเป้าหมายคือ y_N จากตารางที่ 2.3 เป็นตัวอย่างกำหนดจำนวน lag=n=3 ดังนั้น ทุกๆ รูปแบบการฝึกสอนจะมี Input 3 ตัว นำเข้าเพื่อทำการฝึกสอน และจะได้ค่าพยากรณ์(Predicted Value) 1 ค่าสำหรับทุกๆชุดของ Input ที่ใส่เข้าไป

ส่วนจำนวน Lag นั้นปัจจุบันยังไม่มีกฎเกณฑ์ตายตัวว่าจะต้องใช้จำนวนเท่าไหร่จึงจะได้ค่าพยากรณ์ที่ดีที่สุด ทั้งนี้ขึ้นอยู่กับความซับซ้อนของข้อมูล ความแปรผันของข้อมูล ลักษณะเฉพาะของข้อมูล เช่น มีแนวโน้ม มีปัจจัยเชิงฤดูกาล มีการส่ายไปมาไร้รูปแบบ (Noise) ผู้วิจัยจำเป็นต้องทำการทดลองหาจำนวน lag ที่เหมาะสมเอง สำหรับข้อมูลอนุกรมอนุกรมเวลาที่มีปัจจัยเชิงฤดูกาล การกำหนดจำนวน lag ให้คลุมรอบของปัจจัยฤดูกาล จะมีโอกาสเพิ่มความแม่นยำในการพยากรณ์ให้สูงขึ้น(Chen & Wang, 2007)

Training pattern	Input 1	Input 2	Input 3	Target Output
1	y1	y2	y3	$\hat{y}_4$
2	y2	y3	y4	$\hat{y}_5$
3	y3	y4	y5	$\hat{y}_6$
...	...	...	...	...
...	...	...	...	...
96	y96	y97	y98	$\hat{y}_{99}$
97	y97	y98	y99	$\hat{y}_{100}$

ตารางที่ 2.3 ตารางแสดงตัวแปรอิสระและตัวแปรตามสำหรับตัวแบบ ANN ในการพยากรณ์ข้อมูลอนุกรมเวลา กรณี N=100 , n=lag=3

ตัวอย่างโครงสร้างตัวแบบ ANN สำหรับการพยากรณ์อนุกรมเวลาที่ประกอบด้วย Input layer 1 ชั้น มีจำนวนโหนด 1 โหนด , Hidden layer 1 ชั้น มีจำนวนโหนด 5 โหนด และ Output layer 1 ชั้น มีจำนวนโหนด 1 โหนด เมื่อทำการพยากรณ์ 1 ช่วงเวลาไปข้างหน้า(1-step-ahead) กำหนดจำนวนข้อมูลเข้าเท่ากับ 159 ค่า แสดงดังภาพที่ 2.14



ภาพที่ 2.14 ภาพแสดงโครงสร้างตัวแบบ ANN สำหรับการพยากรณ์อนุกรมเวลา

2.4 ตัวแบบซัพพอร์ตเวกเตอร์แมชชีน(Support vector machine model: SVM)

ตัวแบบซัพพอร์ตเวกเตอร์แมชชีน(SVM) เป็นตัวแบบพยากรณ์ที่จัดอยู่ในกลุ่ม Machine Learning ถูกคิดค้นขึ้นเมื่อปี 1995 โดย Vapnik สามารถทำงานได้ดีกับฐานข้อมูลตัวอย่างที่ไม่รู้จัก (Unknown database) ด้วยกระบวนการปรับรูปแบบข้อมูลจากข้อมูลที่มีมิติต่ำ (Low dimension dataset) บนพื้นที่ข้อมูลนำเข้า(Input space) ให้อยู่ในรูปแบบของข้อมูลที่มีมิติสูง(High dimension dataset) บนพื้นที่ข้อมูลคุณลักษณะ(Feature space หรือ High dimensional space) โดยมีแนวคิดมาจากตัวแบบโครงข่ายประสาทเทียม(ANN) แต่ SVM นั้นมีข้อดีกว่าตรงที่มักจะไม่เกิดปัญหา “Overfitting” มากนัก

SVM มีลักษณะคล้ายกับตัวแบบ ANN แต่แตกต่างกันตรงหลักการ Minimization กล่าวคือ ANN นั้นจะใช้หลักการลดความเสี่ยงเชิงทดลองให้ต่ำที่สุด(Empirical Risk Minimization: ERM) ANN จะพยายามทำให้กระบวนการฝึกฝนในชุดข้อมูล Training เกิด Error ต่ำที่สุด นั่นคือโครงสร้างของตัวแบบ ANN อาจจะมีหลายจำนวน Hidden layer หรือมีจำนวน Hidden neurons มาก จนอาจทำให้ตัวแบบเรียนรู้ดีเกินไปในข้อมูล Training หรือโมเดลจดจำรูปแบบของข้อมูล Training มากเกินไปซึ่งอาจทำให้ตัวแบบพยากรณ์ได้ไม่แม่นยำในชุดข้อมูลTesting เรียกว่าเกิดปัญหา “Overfitting” ในขณะที่ Support vector machine จะใช้หลักการลดความเสี่ยงเชิงโครงสร้างให้ต่ำที่สุด (Structural Risk Minimization:SRM) โดย SVM จะมีเทอมหรือฟังก์ชันที่เรียกว่า “Regularization penalty(C)” มาคอยควบคุม เสมือนเป็นตัวเบรกไม่ให้ตัวแบบเรียนรู้ดีเกินไป (ภัทร วรภู, 2556)

2.4.1 ตัวแบบซัพพอร์ทเวกเตอร์แมชชีนสำหรับการถดถอย (Support vector machine model for regression)

(Thissena, Brakela, Weijerb, Melssena, & Buydensa, 2003)กล่าวว่าตัวแบบ SVM สามารถนำมาประยุกต์ใช้สำหรับการวิเคราะห์การถดถอย(Regression analysis) และการพยากรณ์อนุกรมเวลา(Time series prediction)ได้

หลักการของซัพพอร์ทเวกเตอร์แมชชีนสำหรับการถดถอย(Support vector regression :SVR) คือ การหา linear function :f(x) บนพื้นที่ข้อมูลคุณลักษณะ(Feature space) เมื่อกำหนดเซตข้อมูลนำเข้า $\{(x_1, y_1),\ldots,(x_n, y_n)\} \subset R^d xR$ โดยที่ฟังก์ชัน f(x) นั้นจะต้องมีระยะ ε มากที่สุด ในขณะเดียวกันมีค่าคลาดเคลื่อน(Minimize error) น้อยที่สุด กล่าวคือเราต้องการหาฟังก์ชัน

$$f(x) = \langle w, x\rangle + b \quad , \qquad w \in R^d, b \in R,$$

w คือ เวกเตอร์ค่าถ่วงน้ำหนัก

b คือ bias

ซึ่งสมการเส้นตรงที่ดีที่สุดคือ สมการเส้นตรงที่ minimize cost function : $R_{SVM}(C)$

Minimize $$R_{SVM}(C) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} L_{\varepsilon}(y_i - f(x_i, w)) \quad \text{-------- (1)}$$

$$L_{\varepsilon}(y_i, f(x_i, w)) = \begin{cases} 0 & , \ |y_i - f(x_i, w)| \le \varepsilon \\ |y_i - f(x_i, w)| - \varepsilon & , \ otherwise \end{cases}$$

Subject to $$\begin{cases} y_i - \langle w, x \rangle - b \le \varepsilon \\ \langle w, x \rangle + b - y_i \le \varepsilon \end{cases}$$

- เทอมแรก $\frac{1}{2}\|w\|^2$ เรียกว่า “regularization term” ซึ่งเป็นตัวกำหนด flatness of solution

- เทอมที่สอง เรียกว่า “empirical error(risk)” ซึ่งประกอบไปด้วยฟังก์ชันที่เรียกว่า “ε-intensive loss function(L) โดย ε จะหมายถึงค่าที่เรายอมรับการเกิดความผิดพลาด (error)ได้ จะเป็น 0 หากระยะระหว่างค่าจริง และเส้นสมการ regression มีค่าน้อยกว่า ε

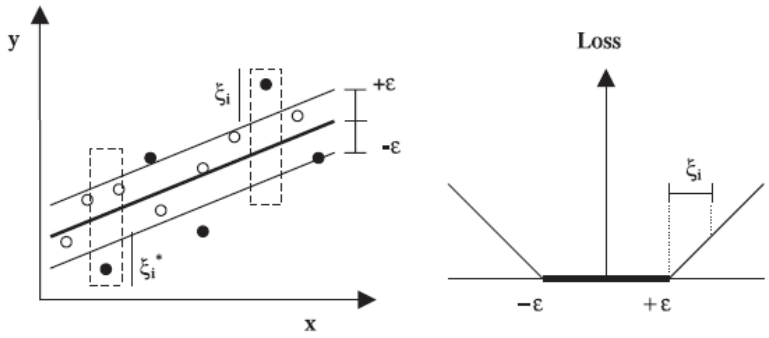
พารามิเตอร์ ε เป็นตัวควบคุมความกว้างของ ε-intensive zone ซึ่งใช้ในการ fit ข้อมูล training ซึ่งจะส่งผลกระทบต่อการสร้างสมการ regression กล่าวคือหาก ε มีค่าน้อยก็จะทำให้ training error มีค่าต่ำ แต่อาจทำให้เกิดปัญหา overfitting ซึ่งทำให้การพยากรณ์ในชุด testing data ไม่แม่นนัก แต่หาก ε มีค่ามากเกินไป จะทำให้ training error มีค่าสูง แต่ก็จะไม่เกิดปัญหา overfitting มากนัก(Chen & Wang, 2007)

ส่วน errors ที่มีค่ามากกว่า $\pm\varepsilon$ จะถูกเรียก หรือกำหนดให้เป็น “Slack variable : $\xi^{(*)}$” ξ_i(เหนือ ε) และ ξ_i^*(ต่ำกว่า ε) ซึ่งแสดงระยะห่างระหว่างค่าจริงที่อยู่นอก ε-intensive zone และสมการ regression ดังนั้นจากสมการที่ (1) จะเปลี่ยนไปเป็นสมการที่ (2)

Minimize $$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \quad \text{-------- (2)}$$

Subject to $$\begin{cases} y_i - \langle w, x \rangle - b \le \varepsilon + \xi_i \\ \langle w, x \rangle + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0, \quad i = 1, 2, ..., n \end{cases}$$

- ค่าคงที่ $C \geq 0$ เรียกว่า “regularized constant” คือค่าที่ปรับสมดุล(trade off)ระหว่างความผิดพลาด(training error) และความซับซ้อนของการรันโมเดล(model’s complexity) เช่น หาก C มีค่ามาก(infinity) เป้าหมายของ SVR จะเป็นการ minimize เทอม empirical risk เพียงอย่างเดียว โดยไม่คำนึงถึง regularization term



ภาพที่ 2.15 (ซ้าย) แสดง tube of ε accuracy จุดสีดำ คือ support vector
(ขวา) แสดง ε-intensive loss function
(ที่มา : Using support vector machines for time series prediction, 2003: 37)

Note ในทางปฏิบัติค่า C และ ε เป็นพารามิเตอร์ของ SVM ที่ต้องกำหนดให้เหมาะสมโดยผู้วิจัย

จากสมการที่ (2) นั้นจะนำ Lagrangian theory มาใช้ในการแก้ปัญหา optimization ที่มี constraint ดังกล่าว และจะได้ค่าเวกเตอร์น้ำหนักที่เหมาะสมคือ

$$w = \sum_{i=1}^{n_{SV}} (\alpha_i - \alpha_i^*) x_i$$

- α_i และ α_i^* คือ “ตัวคูณลากรานจ์” ที่อยู่เหนือและใต้เส้นสมการ regression ตามลำดับ โดย $\alpha_i \times \alpha_i^* = 0$, $\alpha_i \geq 0$, $\alpha_i^* \geq 0$, $i = 1, 2, ..., n$ ซึ่ง Training point ที่มีตัวคูณลากรานจ์ไม่เท่ากับศูนย์จะถูกเรียกว่า “support vector”

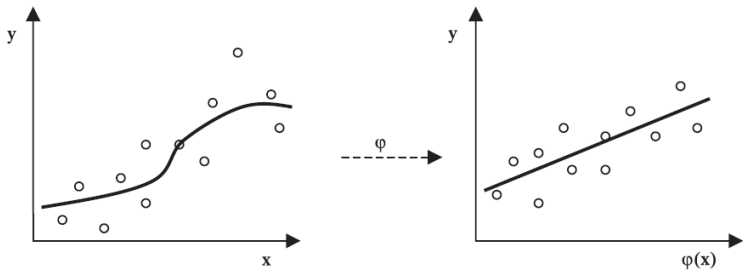
และจะได้ว่าสมการการถดถอยที่เหมาะสมสำหรับพยากรณ์ปัญหา regression คือ

$$f(x) = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) \cdot \langle x_i, x \rangle + b$$

สุดท้ายจะทำการแปลงข้อมูลนำเข้าที่มีลักษณะไม่เป็นเชิงเส้นตรงบนพื้นที่ข้อมูลนำเข้า x_i ไปสู่พื้นที่คุณลักษณะหลายมิติ(higher-dimensional feature space) โดยการใช้ "kernel function" เพื่อให้สามารถหาสมการเชิงเส้นตรงบนพื้นที่ข้อมูลคุณลักษณะนี้ได้ ซึ่งจะได้สมการ regression ที่เหมาะสมในการพยากรณ์ คือ

$$f(x) = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) \cdot K\langle x_i, x\rangle + b$$

โดย $K\langle x_i, x\rangle = \langle \phi(x_i), \phi(x)\rangle$



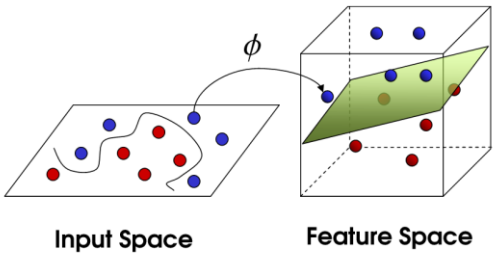
ภาพที่ 2.16 การแปลง(Mapping) ข้อมูลนำเข้า x_i ไปสู่พื้นที่คุณลักษณะหลายมิติ
(ที่มา : Using support vector machines for time series prediction, 2003: 38)

2.4.2 ตัวแบบ Support vector machine สำหรับพยากรณ์ข้อมูลอนุกรมเวลา (Support vector machine model for time series)

การพยากรณ์ข้อมูลอนุกรมเวลาด้วย SVM นั้นมีหลักการเช่นเดียวกันกับการพยากรณ์ปัญหา Regression ที่มีตัวแปรอิสระและตัวแปรตาม แต่ต้องปรับข้อมูลอนุกรมเวลาให้เป็นตัวแปร Lag เหมือนกับการพยากรณ์ข้อมูลอนุกรมเวลาด้วยตัวแบบ ANN ส่วนค่าพารามิเตอร์ต่างๆนั้นผู้วิจัยต้องทดลองหาค่าที่เหมาะสมจนกระทั่งได้ตัวแบบที่ให้ผลการพยากรณ์ที่น่าพอใจ

2.4.3 Kernel Function

เป็นฟังก์ชันที่ทำให้ SVM สามารถแก้ปัญหาที่ไม่เป็นเชิงเส้นตรง(Nonlinear)ได้ หลักการของ kernel function นั้นจะเป็นการเพิ่มมิติของข้อมูลเพื่อที่จะแปลงจุดพิกัดจากพื้นที่ข้อมูลนำเข้า(Input space) ให้มีการจัดเรียงใหม่(Mapping)ในพื้นที่หลายมิติ(Higher dimensional space) หรือพื้นที่คุณลักษณะ(Feature space) ทำให้สามารถค้นหา Hyperplane แนวเส้นตรงได้ดังภาพ



ภาพที่ 2.17 การแปลงข้อมูลโดยใช้ kernel Function จากพื้นที่ข้อมูลนำเข้า ให้ข้อมูลมีการเรียงตัวใหม่ในพื้นที่คุณลักษณะ

ประเภทของ kernel function นั้นมีหลายประเภท(Wang, 2014) เช่น

1. Linear

$$K(x,x') = \langle x, x' \rangle$$

2. Gaussian Radial Basis Function (RBF)

$$K(x,x') = \exp\left(-\frac{\| x - x' \|^2}{2\sigma^2}\right)$$

3. Polynomial

$$K(x,x') = \langle x, x' + 1 \rangle^d$$

4. Sigmoid

$$K(x_i,x_j) = \tanh(s\langle x_i, x_j \rangle + c$$

โดยที่ c,d,s เป็นค่า Parameter สำหรับ Kernel ในแต่ละฟังก์ชัน

(ภัทร วรภู, 2556)กล่าวว่าจนกระทั่งถึงปัจจุบันนี้ ยังไม่มีกฎเกณฑ์ตายตัวในการเลือกชนิดของ Kernel function ที่ดีที่สุดสำหรับทุกชุดข้อมูล

ดังนั้น ในงานวิจัยชิ้นนี้จึงเลือก Gaussian Radial Basis Function(RBF) เป็น Kernel Function เนื่องจากง่ายต่อการนำไปใช้เนื่องจากมีพารามิเตอร์ที่ต้องใส่ค่าตัวเดียว คือ σ และมีประสิทธิภาพที่ดีในการแปลง Nonlinear mapping จาก Input space ไปสู่ High dimensional feature space (Chen and Wang, 2007: 257)

2.4.4 การกำหนดค่าพารามิเตอร์ต่างๆของ SVM ในงานวิจัย

- ค่าพารามิเตอร์ของ SVM คือ C และ ε

(Hsu, Chang, & Lin, 2016)ได้ทำการวิจัยและสรุปว่าการทดลองเลือกใช้ค่า C เป็นแบบ exponentially growing sequence กล่าวคือ $C = 2^{-5}, 2^{-3}, ..., 2^{-15}$ จะทำให้ได้ค่าพารามิเตอร์ที่เหมาะสม

ดังนั้น ในงานวิจัยนี้จะใช้คำสั่ง tune.svm ในการปรับจูนค่า C และ ε ที่ดีที่สุดสำหรับข้อมูลชุดนั้นๆ โดยใช้ค่า c=2^{-5},2^{-3},...,2^{3},2^{5} และ ε = 0.001, 0.01, 0.1 และ 1

- Kernel function ที่เลือกใช้คือ Gaussian Radial Basis Function (RBF)

โดยพารามิเตอร์ของ Kernel Function RBF คือ σ จะใช้คำสั่ง sigest ใน library “kernlab” ช่วยในการหาค่า sigma ที่ดีที่สุดสำหรับข้อมูลในชุดนั้นๆ และหา σ ที่ดีที่สุดจากการทำซ้ำทั้งหมด 1000 รอบ โดยพิจารณาจาก RMSE ที่ต่ำที่สุด

- จำนวน Lag

ในงานวิจัยนี้จะใช้จำนวน Lag=1 เพื่อให้สอดคล้องกับการพยากรณ์ด้วยตัวแบบ ANN ที่เลือกใช้จำนวน Lag=1 เช่นเดียวกัน

2.5 ตัวแบบผสม(Hybrid Model)

(G. P. ZHANG, 2003)ได้สรุปหลักในการสร้างตัวแบบผสมไว้ดังนี้ กล่าวคือการสร้างตัวแบบผสมจะกำหนดว่าข้อมูลที่นำมาวิเคราะห์นั้นมี 2 องค์ประกอบ คือ องค์ประกอบที่เป็นเส้นตรง (linear component) และองค์ประกอบที่ไม่เป็นเส้นตรง(nonlinear Component) ดังสมการ

$$y_t = L_t + N_t$$

โดย L_t แทนองค์ประกอบที่เป็นเส้นตรง(linear component)

N_t แทนองค์ประกอบที่ไม่เป็นเส้นตรง(linear component)

1. วิเคราะห์ข้อมูลอนุกรมเวลาด้วยตัวแบบ ARIMA เพื่อทำการพยากรณ์ข้อมูลในส่วนที่เป็นฟังก์ชันเชิงเส้นตรง คือ L_t
2. คำนวณค่าส่วนเหลือ(residuals) จากการพยากรณ์ด้วยตัวแบบ ARIMA โดยส่วนเหลือนั้นจะเหลือเพียงฟังก์ชันส่วนที่ไม่เป็นเส้นตรง นั่นคือ $e_t = y_t - \hat{L}_t$ โดย $\hat{L}_t$ เป็นค่าพยากรณ์ที่ได้จากการพยากรณ์ข้อมูลอนุกรมเวลาในส่วนที่เป็นฟังก์ชันเชิงเส้นตรงด้วยตัวแบบ ARIMA

3. นำค่าส่วนเหลือที่ได้จากตัวแบบ ARIMA ไปพยากรณ์ข้อมูลส่วนที่ไม่เป็นฟังก์ชันเชิงเส้นตรง($\hat{N}_t$) ด้วยตัวแบบ ANN หรือ SVM
4. คำนวณค่าพยากรณ์รวม $y_t = \hat{L}_t + \hat{N}_t$

2.6. เกณฑ์ที่ใช้ในการตัดสินใจ

เกณฑ์ที่ใช้ในการคัดเลือกตัวแบบในการพยากรณ์ที่ดีที่สุดมีอยู่หลายเกณฑ์ดังต่อไปนี้

- ค่าคลาดเคลื่อนกำลังสองเฉลี่ย(Mean Square Error: MSE) เป็นค่าวัดความถูกต้องของการพยากรณ์ที่วัดจากขนาดของค่าความคลาดเคลื่อนของการพยากรณ์ที่ได้จากกำลังสองของค่าความคลาดเคลื่อน

$$MSE = \frac{\sum_{t=1}^{n}(x_t - \hat{x}_t)^2}{n}$$

- รากของค่าคลาดเคลื่อนกำลังสองเฉลี่ย(Root Mean Square Error: RMSE) เป็นค่าวัดความถูกต้องของการพยากรณ์ โดยเป็นรากกำลังสองของค่า MSE

$$RMSE = \sqrt{MSE} = \sqrt{\frac{\sum_{t=1}^{n}(x_t - \hat{x}_t)^2}{n}}$$

- ค่าความคลาดเคลื่อนสัมบูรณ์เฉลี่ย(Mean Absolute Error: MAE) เป็นค่าวัดความถูกต้องของการพยากรณ์ที่วัดจากขนาดของค่าความคลาดเคลื่อนของการพยากรณ์โดยไม่คำนึงถึงทิศทางของความคลาดเคลื่อน

$$MAE = \frac{\sum_{t=1}^{n}| x_t - \hat{x}_t |}{n}$$

ในงานวิจัยนี้เลือกใช้ RMSE เป็นเกณฑ์ที่ใช้ในการคัดเลือกตัวแบบเนื่องจากง่ายต่อการวัดเพราะมีหน่วยเดียวกันกับข้อมูล ซึ่งจะทำให้สามารถเปรียบเทียบตัวแบบได้สะดวกมากขึ้น โดยตัวแบบที่มีค่า RMSE ที่ต่ำที่สุด จะเป็นตัวแบบที่มีความแม่นยำในการพยากรณ์มากที่สุด

บทที่ 3
วิธีดำเนินงานวิจัย

งานวิจัยนี้มีวัตถุประสงค์เพื่อเปรียบเทียบความแม่นของค่าพยากรณ์ที่ได้จากตัวแบบ SARIMA, ตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบโครงข่ายประสาทเทียม(SARIMA+ANN) และตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน(SARIMA+SVM) โดยใช้โปรแกรม R เวอร์ชัน 3.4.3 ในการวิเคราะห์ข้อมูล

3.1 ขั้นตอนในการดำเนินงานวิจัย

ส่วนที่ 1 : ศึกษาโดยใช้ข้อมูลจำลอง

ขั้นตอนที่ 1 จำลองชุดข้อมูลด้วยตัวแบบ ARIMA(p,1,q)x(P,1,Q)s จำนวน 240 จุดเวลา ภายใต้ค่า P=0-1 ,Q=0-1 ,p=1-2 และ q=1-2 รวมทั้งสิ้น 12 ตัวแบบ กำหนดค่าพารามิเตอร์ $\Phi, \phi, \Theta, \theta$ เท่ากับ -0.5 และ 0.5 ในแต่ละตัวแบบกำหนดจำนวนรอบในการทำซ้ำเท่ากับ 100 รอบ

ขั้นตอนที่ 2 ทำการแบ่งข้อมูลในแต่ละรอบของการจำลองออกเป็นชุดข้อมูลฝึกสอน 70% (จำนวน 159 จุดเวลา)สำหรับสร้างตัวแบบ และชุดข้อมูลทดสอบ 30%(จำนวน 68 จุดเวลา) สำหรับประเมินความถูกต้องของตัวแบบ

ขั้นตอนที่ 3 นำชุดข้อมูลฝึกสอนมาสร้างตัวแบบ ARIMA(p,1,q)x(P,1,Q)$_{12}$ ภายใต้ค่า P=0-1, Q=0-1, p=1-2 และ q=1-2 แล้วทำการพิจารณาคัดเลือกตัวแบบที่เหมาะสมที่สุดโดยพิจารณาจากค่า AIC ที่ต่ำที่สุด จากนั้นทำการพยากรณ์ข้อมูลด้วยตัวแบบที่คัดเลือกได้ในชุดข้อมูลฝึกสอน และชุดข้อมูลทดสอบ ซึ่งจัดเป็นข้อมูลส่วนที่เป็นฟังก์ชันเชิงเส้นตรง คำนวณหาค่า RMSE ในแต่ละรอบของการจำลอง และค่าเฉลี่ยของ RMSE จากการจำลองทั้งหมด 100 รอบ

ขั้นตอนที่ 4 นำค่าส่วนเหลือ(Residuals) ที่ได้จากการพยากรณ์ด้วยตัวแบบ SARIMA ในชุดข้อมูลฝึกสอนจากข้อมูลจำลองในแต่ละรอบมาสร้างตัวแบบโครงข่ายประสาทเทียมเพอร์เซ็ปตรอนหลายชั้น(MLP) ประกอบด้วย Input layer 1 ชั้น 1 โหนด Hidden layer 1 ชั้น 1-5 โหนด และ Output layer 1 ชั้น 1 โหนด โดยใช้ Sigmoid logistics function เป็น Activation function และ Output layer 1 ชั้น ภายใต้จำนวนโหนด 1 โหนด โดยใช้ Linear function เป็น Activation function และใช้เทคนิคการฝึกสอนโครงข่ายแบบ Resilient back propagation(Rprop) โดยกำหนด $\eta^- = 0.5$, $\eta^+ = 1.2$ และ $\Delta_0 = 0.1$ ใช้จำนวนรอบของการทำซ้ำ(Iterations) เท่ากับ

100 รอบ จากนั้นทำการพิจารณาคัดเลือกตัวแบบโครงข่ายประสาทเทียมที่เหมาะสมที่สุด โดยพิจารณาจากค่า RMSE ที่ต่ำที่สุด และทำการพยากรณ์ข้อมูลส่วนที่ไม่เป็นฟังก์ชันเชิงเส้นตรงในเทอมของค่าพารามิเตอร์ด้วยตัวแบบโครงข่ายประสาทเทียมในชุดข้อมูลทดสอบจากข้อมูลจำลองในแต่ละรอบ จากนั้นคำนวณค่าพยากรณ์รวม(Total forecasting) ซึ่งเป็นการรวมข้อมูลส่วนที่เป็นฟังก์ชันเชิงเส้นตรงที่ได้จากการพยากรณ์ด้วยตัวแบบ SARIMA และข้อมูลที่ไม่เป็นฟังก์ชันเชิงเส้นตรงที่ได้จากการพยากรณ์ด้วยตัวแบบโครงข่ายประสาทเทียมเข้าด้วยกัน คำนวณค่า RMSE ในแต่ละรอบของการจำลอง และคำนวณหาค่าเฉลี่ยของ RMSE จากการจำลองทั้งหมด 100 รอบ

ขั้นตอนที่ 5 นำค่าส่วนเหลือ(Residuals) ที่ได้จากการพยากรณ์ด้วยตัวแบบ SARIMA ในชุดข้อมูลฝึกสอนจากข้อมูลจำลองในแต่ละรอบมาสร้างตัวแบบซัพพอร์ทเวกเตอร์แมชชีน(SVM) โดยทำการปรับจูนหาชุดของค่าพารามิเตอร์ของ SVM คือ C และ ε ที่เหมาะสมที่สุด โดยใช้ค่า $C=2^{-5}$, $2^{-3},\ldots,2^{3},2^{5}$ และ ε = 0.001, 0.01, 0.1, และ 1 และใช้ Gaussian radial basis function(RBF) เป็น Kernel function โดยจะหาค่าพารามิเตอร์ σ ที่ดีที่สุดโดยใช้คำสั่ง “sigest” จากนั้นทำการพิจารณาคัดเลือกตัวแบบซัพพอร์ทเวกเตอร์แมชชีนที่เหมาะสมที่สุดจากการทำซ้ำ 100 รอบ โดยพิจารณาจากค่า RMSE ที่ต่ำที่สุด และทำการพยากรณ์ข้อมูลส่วนที่ไม่เป็นฟังก์ชันเชิงเส้นตรงด้วยตัวแบบซัพพอร์ทเวกเตอร์แมชชีนในชุดข้อมูลทดสอบจากข้อมูลจำลองในแต่ละรอบ จากนั้นคำนวณค่าพยากรณ์รวม(Total forecasting) ซึ่งเป็นการรวมข้อมูลส่วนที่เป็นฟังก์ชันเชิงเส้นตรง ที่ได้จากการพยากรณ์ด้วยตัวแบบ ARIMA ที่มีฤดูกาล และข้อมูลที่ไม่เป็นฟังก์ชันเชิงเส้นตรงที่ได้จากการพยากรณ์ด้วยตัวแบบซัพพอร์ทเวกเตอร์แมชชีนเข้าด้วยกันด้วยกัน คำนวณค่า RMSE ในแต่ละรอบของการจำลอง และคำนวณหาค่าเฉลี่ยของ RMSE จากการจำลองทั้งหมด 100 รอบ

ขั้นตอนที่ 6 สุดท้ายทำการเปรียบเทียบความแม่นยำของค่าพยากรณ์ที่ได้จาก 3 ตัวแบบ คือ ตัวแบบ ARIMA ที่มีฤดูกาล(SARIMA), ตัวแบบผสมระหว่างตัวแบบ ARIMA ที่มีฤดูกาลกับตัวแบบโครงข่ายประสาทเทียม(SARIMA+ANN) และตัวแบบผสมระหว่างตัวแบบ ARIMA ที่มีฤดูกาลกับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน(SARIMA+SVM) สำหรับข้อมูลจำลอง โดยตัวแบบที่ดีที่สุดจะมีค่า RMSE ที่ต่ำที่สุด

ส่วนที่ 2 : ศึกษาโดยใช้ข้อมูลจริง

ขั้นตอนที่ 1 เก็บรวบรวมข้อมูลจริงราคาขายปลีกมะนาวเบอร์ 1-2 (หน่วยเป็นบาท/ผล) จากกรมการค้าภายใน กระทรวงพาณิชย์ ข้อมูลเป็นรายเดือนตั้งแต่เดือนมกราคม พ.ศ.2540 ถึงเดือนธันวาคม พ.ศ.2559 จำนวน 240 ตัวอย่าง ใช้ผลต่าง(Differencing) และผลต่างฤดูกาล(Seasonal Differencing) เพื่อปรับข้อมูลให้คงที่(Stationary) และไม่มีแนวโน้มเนื่องจากฤดูกาล

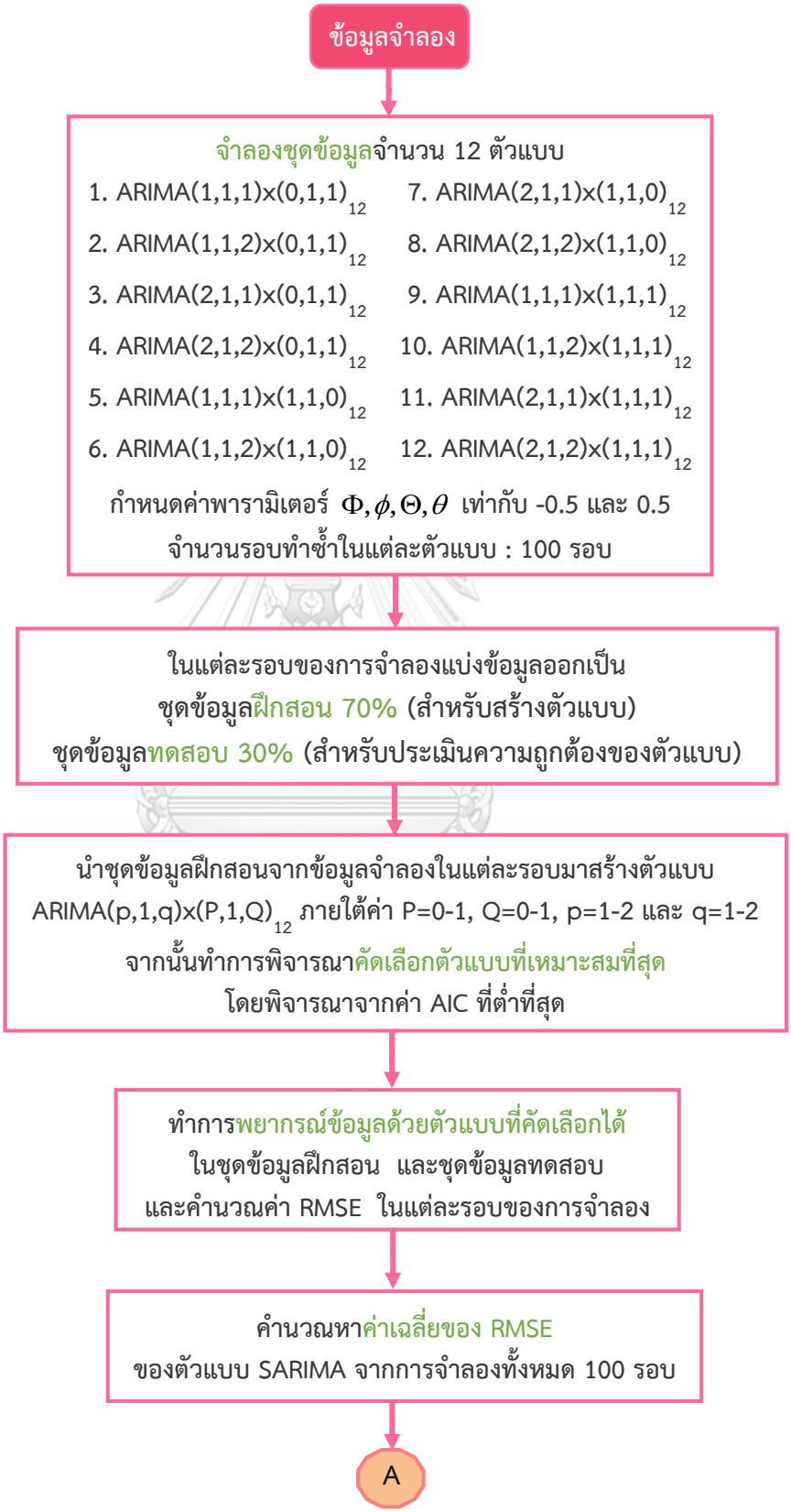
ขั้นตอนที่ 2 ทำการแบ่งข้อมูลออกเป็นชุดข้อมูลฝึกสอน 70%(จำนวน 159 จุดเวลา)สำหรับสร้างตัวแบบ และชุดข้อมูลทดสอบ 30%(จำนวน 68 จุดเวลา) สำหรับประเมินความถูกต้องของตัวแบบ

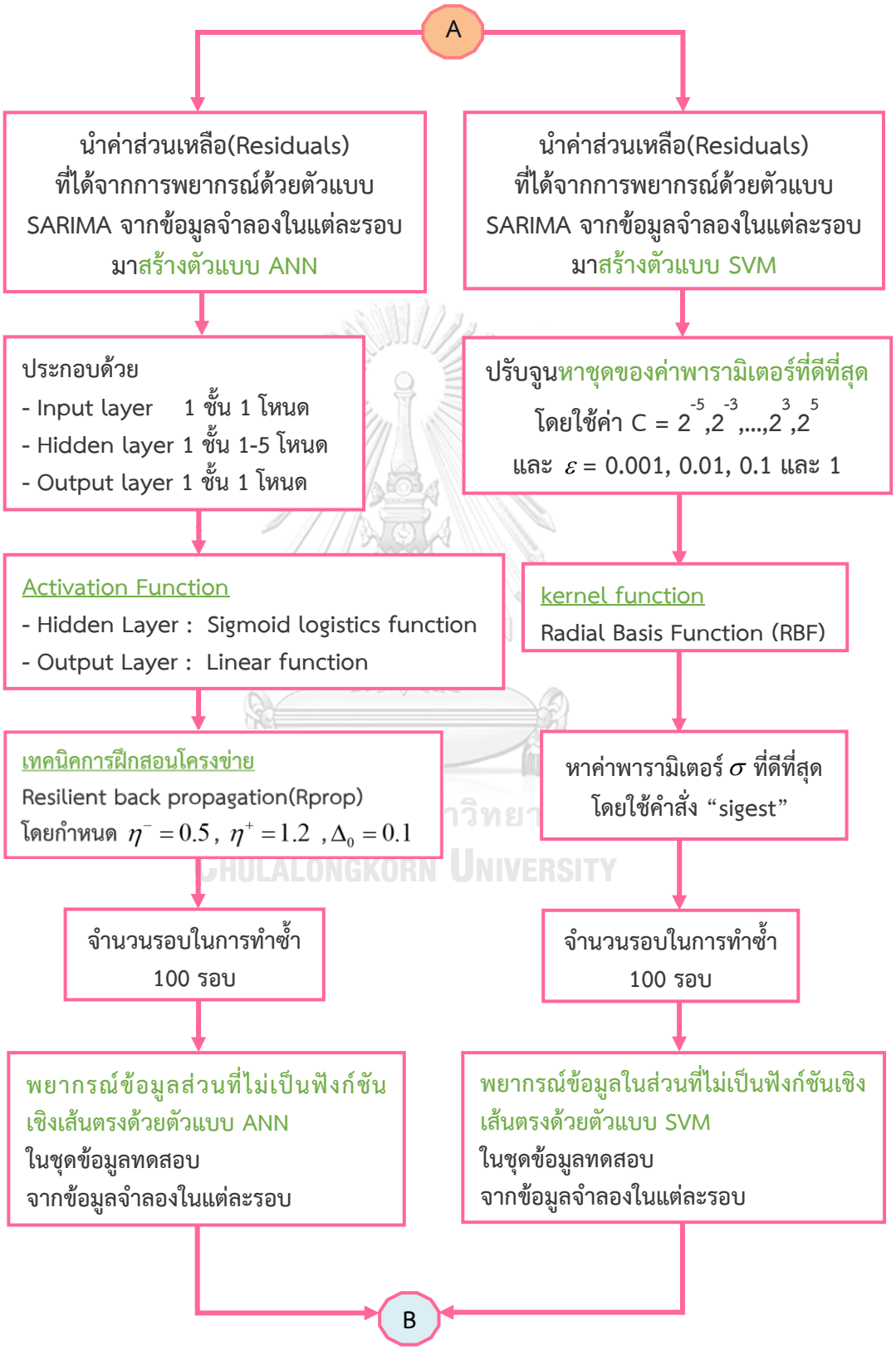
ขั้นตอนที่ 3 นำชุดข้อมูลฝึกสอนจากข้อมูลจริงมาสร้างตัวแบบ ARIMA(p,1,q)x(P,1,Q)$_{12}$ ภายใต้ค่า P=0-1, Q=0-1 , p=1-2 และ q=1-2 แล้วทำการพิจารณาคัดเลือกตัวแบบที่เหมาะสมที่สุดโดยพิจารณาจากค่า AIC ที่ต่ำที่สุด จากนั้นทำการพยากรณ์ราคาราคาขายปลีกมะนาวด้วยตัวแบบที่คัดเลือกได้ในชุดข้อมูลฝึกสอน และชุดข้อมูลทดสอบ ซึ่งจัดเป็นข้อมูลส่วนที่เป็นฟังก์ชันเชิงเส้นตรง และคำนวณค่า RMSE จากชุดข้อมูลทดสอบ

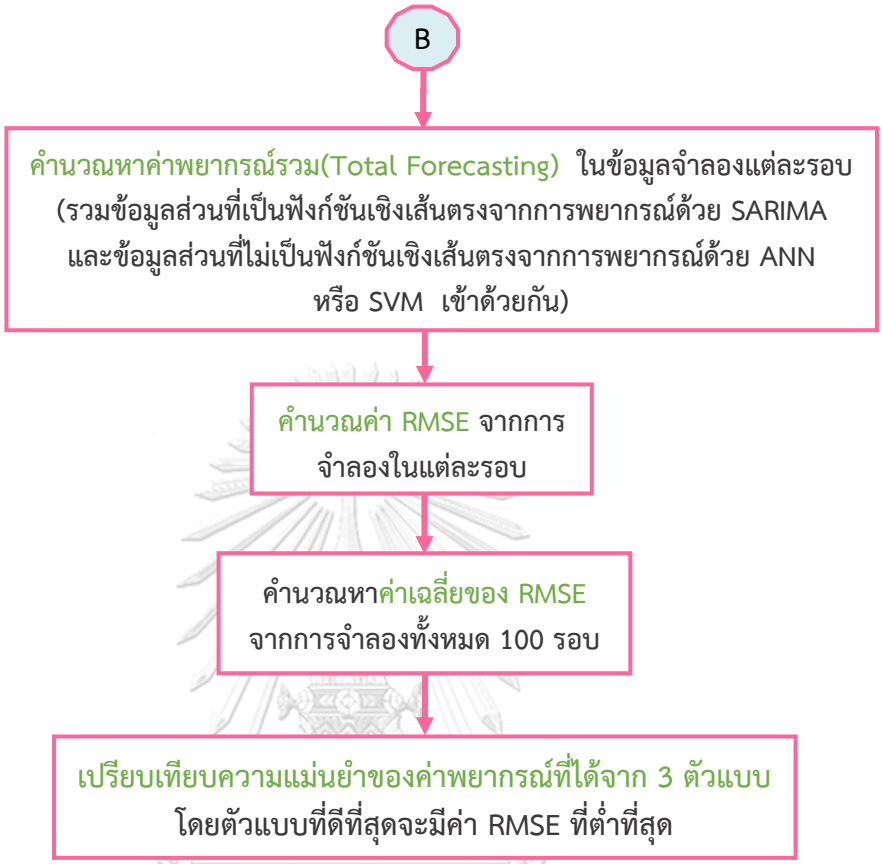
ขั้นตอนที่ 4 นำค่าส่วนเหลือ(Residuals)ที่ได้จากการพยากรณ์ด้วยตัวแบบ SARIMA ในชุดข้อมูลฝึกสอนจากข้อมูลจริงมาสร้างตัวแบบ ANN และ SVM ภายใต้ขอบเขตการวิจัยในข้อมูลจำลอง จากนั้นทำการพยากรณ์ข้อมูลในส่วนที่ไม่เป็นฟังก์ชันเชิงเส้นตรงด้วยตัวแบบ ANN หรือ SVM ในชุดข้อมูลทดสอบ แล้วคำนวณหาค่าพยากรณ์รวม(Total Forecasting) ซึ่งเป็นการรวมข้อมูลในส่วนที่เป็นฟังก์ชันเชิงเส้นตรงที่ได้จากการพยากรณ์ด้วยตัวแบบ SARIMA และข้อมูลในส่วนที่ไม่เป็นฟังก์ชันเชิงเส้นตรงที่ได้จากการพยากรณ์ด้วยตัวแบบ ANN หรือ SVM เข้าด้วยกัน และคำนวณหาค่า RMSE ในชุดข้อมูลทดสอบ

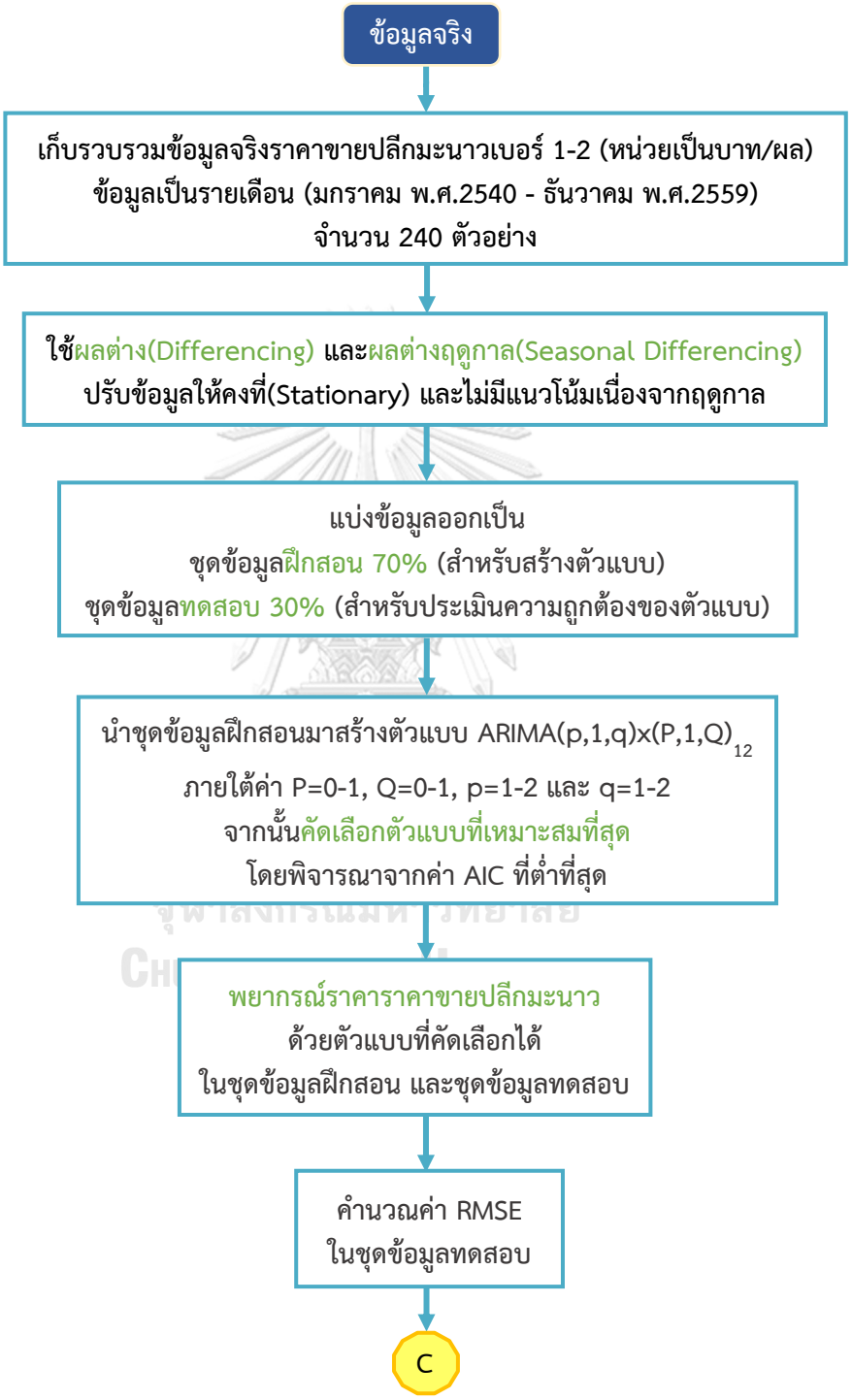
ขั้นตอนที่5 สุดท้ายทำการเปรียบเทียบความแม่นยำของค่าพยากรณ์ที่ได้จาก 3 ตัวแบบ โดยตัวแบบที่ดีที่สุดจะมีค่า RMSE ที่ต่ำที่สุด

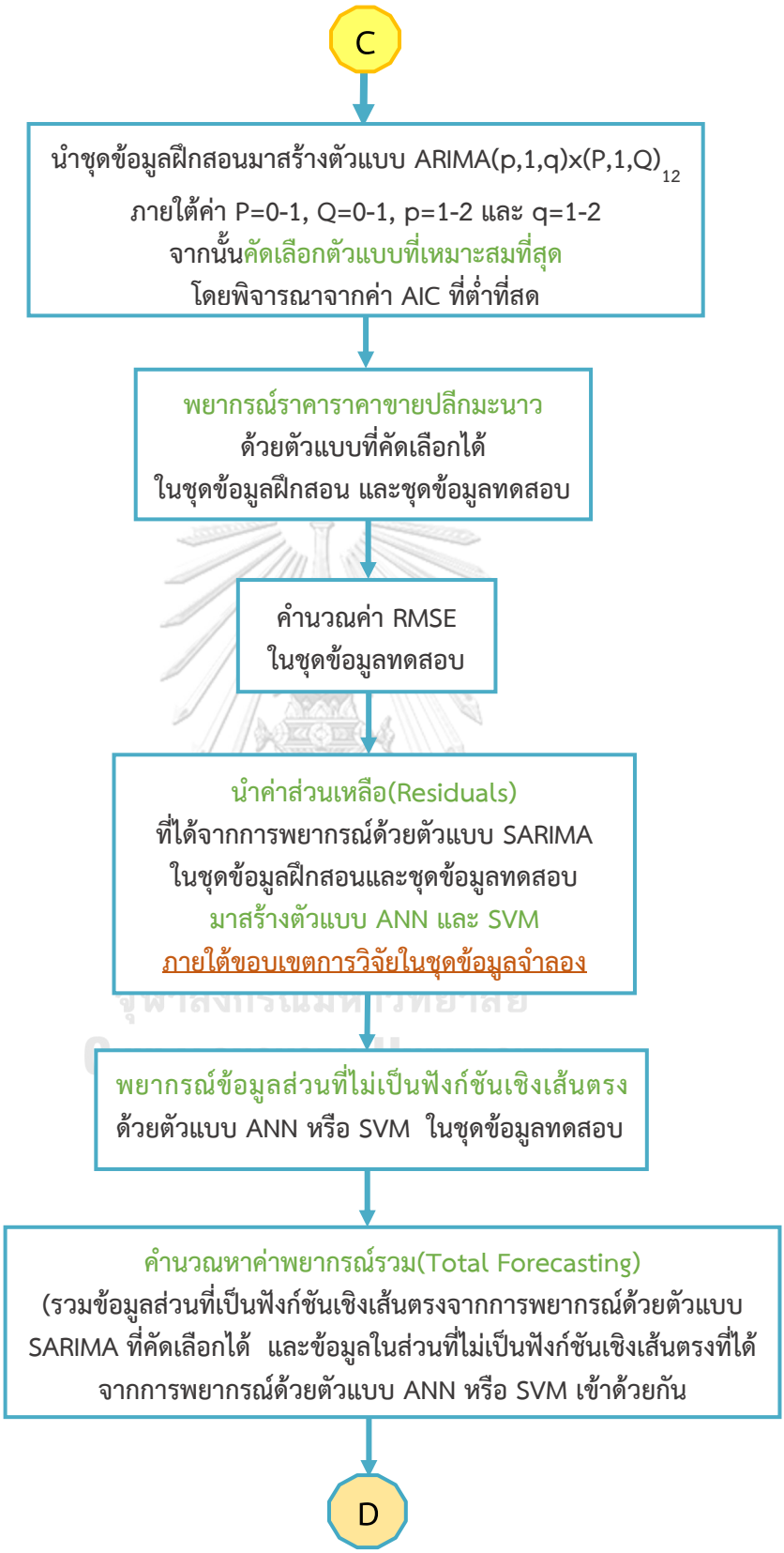
3.2 แผนผังการดำเนินงานวิจัย

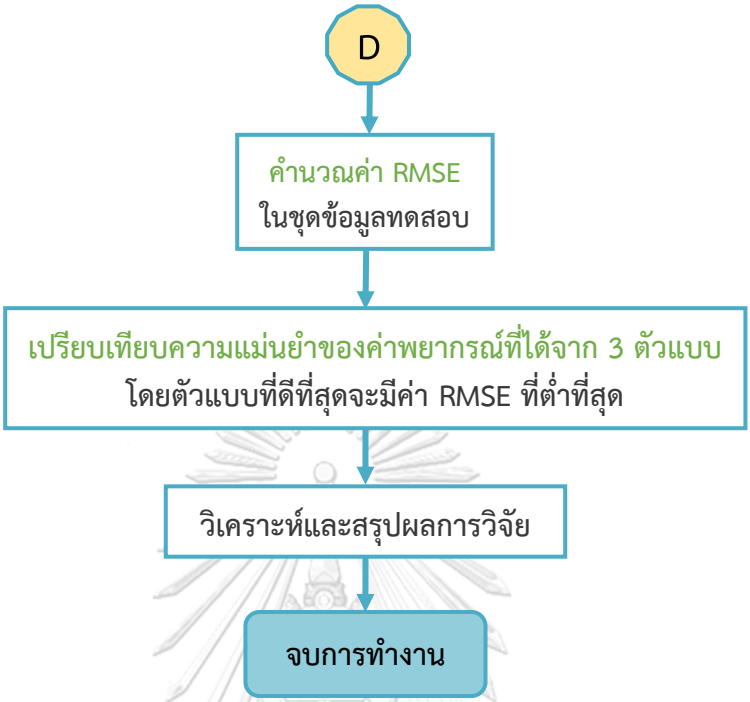



A
นำค่าส่วนเหลือ(Residuals) ที่ได้จากการพยากรณ์ด้วยตัวแบบ SARIMA จากข้อมูลจำลองในแต่ละรอบ มาสร้างตัวแบบ ANN
นำค่าส่วนเหลือ(Residuals) ที่ได้จากการพยากรณ์ด้วยตัวแบบ SARIMA จากข้อมูลจำลองในแต่ละรอบ มาสร้างตัวแบบ SVM
ประกอบด้วย
- Input layer 1 ชั้น 1 โหนด
- Hidden layer 1 ชั้น 1-5 โหนด
- Output layer 1 ชั้น 1 โหนด
ปรับจูนหาชุดของค่าพารามิเตอร์ที่ดีที่สุด
โดยใช้ค่า C = 2^-5,2^-3,...,2^3,2^5
และ ε = 0.001, 0.01, 0.1 และ 1
Activation Function
- Hidden Layer : Sigmoid logistics function
- Output Layer : Linear function
kernel function
Radial Basis Function (RBF)
เทคนิคการฝึกสอนโครงข่าย
Resilient back propagation(Rprop)
โดยกำหนด η⁻ = 0.5, η⁺ = 1.2, Δ0 = 0.1
หาค่าพารามิเตอร์ σ ที่ดีที่สุด
โดยใช้คำสั่ง “sigest”
จำนวนรอบในการทำซ้ำ
100 รอบ
จำนวนรอบในการทำซ้ำ
100 รอบ
พยากรณ์ข้อมูลส่วนที่ไม่เป็นฟังก์ชันเชิงเส้นตรงด้วยตัวแบบ ANN
ในชุดข้อมูลทดสอบ
จากข้อมูลจำลองในแต่ละรอบ
พยากรณ์ข้อมูลในส่วนที่ไม่เป็นฟังก์ชันเชิงเส้นตรงด้วยตัวแบบ SVM
ในชุดข้อมูลทดสอบ
จากข้อมูลจำลองในแต่ละรอบ
B




ข้อมูลจริง
เก็บรวบรวมข้อมูลจริงราคาขายปลีกมะนาวเบอร์ 1-2 (หน่วยเป็นบาท/ผล)
ข้อมูลเป็นรายเดือน (มกราคม พ.ศ.2540 - ธันวาคม พ.ศ.2559)
จำนวน 240 ตัวอย่าง
ใช้ผลต่าง(Differencing) และผลต่างฤดูกาล(Seasonal Differencing)
ปรับข้อมูลให้คงที่(Stationary) และไม่มีแนวโน้มเนื่องจากฤดูกาล
แบ่งข้อมูลออกเป็น
ชุดข้อมูลฝึกสอน 70% (สำหรับสร้างตัวแบบ)
ชุดข้อมูลทดสอบ 30% (สำหรับประเมินความถูกต้องของตัวแบบ)
นำชุดข้อมูลฝึกสอนมาสร้างตัวแบบ ARIMA(p,1,q)x(P,1,Q)12
ภายใต้ค่า P=0-1, Q=0-1, p=1-2 และ q=1-2
จากนั้นคัดเลือกตัวแบบที่เหมาะสมที่สุด
โดยพิจารณาจากค่า AIC ที่ต่ำที่สุด
พยากรณ์ราคาราคาขายปลีกมะนาว
ด้วยตัวแบบที่คัดเลือกได้
ในชุดข้อมูลฝึกสอน และชุดข้อมูลทดสอบ
คำนวณค่า RMSE
ในชุดข้อมูลทดสอบ
C


C
นำชุดข้อมูลฝึกสอนมาสร้างตัวแบบ ARIMA(p,1,q)x(P,1,Q)12
ภายใต้ค่า P=0-1, Q=0-1, p=1-2 และ q=1-2
จากนั้นคัดเลือกตัวแบบที่เหมาะสมที่สุด
โดยพิจารณาจากค่า AIC ที่ต่ำที่สด
พยากรณ์ราคาราคาขายปลีกมะนาว
ด้วยตัวแบบที่คัดเลือกได้
ในชุดข้อมูลฝึกสอน และชุดข้อมูลทดสอบ
คำนวณค่า RMSE
ในชุดข้อมูลทดสอบ
นำค่าส่วนเหลือ(Residuals)
ที่ได้จากการพยากรณ์ด้วยตัวแบบ SARIMA
ในชุดข้อมูลฝึกสอนและชุดข้อมูลทดสอบ
มาสร้างตัวแบบ ANN และ SVM
ภายใต้ขอบเขตการวิจัยในชุดข้อมูลจำลอง
พยากรณ์ข้อมูลส่วนที่ไม่เป็นฟังก์ชันเชิงเส้นตรง
ด้วยตัวแบบ ANN หรือ SVM ในชุดข้อมูลทดสอบ
คำนวณหาค่าพยากรณ์รวม(Total Forecasting)
(รวมข้อมูลส่วนที่เป็นฟังก์ชันเชิงเส้นตรงจากการพยากรณ์ด้วยตัวแบบ
SARIMA ที่คัดเลือกได้ และข้อมูลในส่วนที่ไม่เป็นฟังก์ชันเชิงเส้นตรงที่ได้
จากการพยากรณ์ด้วยตัวแบบ ANN หรือ SVM เข้าด้วยกัน
D



บทที่ 4
ผลการวิจัย

งานวิจัยนี้มีวัตถุประสงค์เพื่อเปรียบเทียบความแม่นยำของค่าพยากรณ์ที่ได้จาก 3 ตัวแบบ คือ ตัวแบบ SARIMA, ตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบโครงข่ายประสาทเทียม (SARIMA+ANN) และตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน (SARIMA+SVM) โดยทำการเปรียบเทียบทั้งในส่วนของข้อมูลจริง และข้อมูลจำลอง ซึ่งจะถูกแบ่งออกเป็นชุดข้อมูลฝึกสอน(Training data)จำนวน 70% สำหรับสร้างตัวแบบ และชุดข้อมูลทดสอบ (Testing Data)จำนวน 30% สำหรับตรวจสอบความถูกต้องของตัวแบบ และในขั้นตอนสุดท้ายจะทำการเปรียบเทียบความแม่นยำของค่าพยากรณ์โดยใช้เกณฑ์รากของค่าคลาดเคลื่อนกำลังสองเฉลี่ย (Root Mean Square Error: RMSE) เป็นเกณฑ์ในการเปรียบเทียบตัวแบบ

งานวิจัยนี้จะนำเสนอผลการเปรียบเทียบความแม่นยำของค่าพยากรณ์ โดยแบ่งออกเป็น 2 ส่วน คือ ส่วนของข้อมูลจำลอง และส่วนของข้อมูลจริง ดังนี้

4.1 ผลการเปรียบเทียบความแม่นยำของค่าพยากรณ์ที่ได้จากตัวแบบ SARIMA, ตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบโครงข่ายประสาทเทียม(SARIMA+ANN) และตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน(SARIMA+SVM) โดยใช้<u>ชุดข้อมูลจำลอง</u> รวมทั้งสิ้น 12 ตัวแบบ(128 กรณี) คือ

แบบจำลองที่ 1 ARIMA(1,1,1)x$(0,1,1)_{12}$ มีทั้งสิ้น 8 กรณีย่อย
แบบจำลองที่ 2 ARIMA(1,1,2)x$(0,1,1)_{12}$ มีทั้งสิ้น 8 กรณีย่อย
แบบจำลองที่ 3 ARIMA(2,1,1)x$(0,1,1)_{12}$ มีทั้งสิ้น 8 กรณีย่อย
แบบจำลองที่ 4 ARIMA(2,1,2)x$(0,1,1)_{12}$ มีทั้งสิ้น 8 กรณีย่อย
แบบจำลองที่ 5 ARIMA(1,1,1)x$(1,1,0)_{12}$ มีทั้งสิ้น 8 กรณีย่อย
แบบจำลองที่ 6 ARIMA(1,1,2)x$(1,1,0)_{12}$ มีทั้งสิ้น 8 กรณีย่อย
แบบจำลองที่ 7 ARIMA(2,1,1)x$(1,1,0)_{12}$ มีทั้งสิ้น 8 กรณีย่อย
แบบจำลองที่ 8 ARIMA(2,1,2)x$(1,1,0)_{12}$ มีทั้งสิ้น 8 กรณีย่อย
แบบจำลองที่ 9 ARIMA(1,1,1)x$(1,1,1)_{12}$ มีทั้งสิ้น 16 กรณีย่อย
แบบจำลองที่ 10 ARIMA(1,1,2)x$(1,1,1)_{12}$ มีทั้งสิ้น 16 กรณีย่อย
แบบจำลองที่ 11 ARIMA(2,1,1)x$(1,1,1)_{12}$ มีทั้งสิ้น 16 กรณีย่อย
แบบจำลองที่ 12 ARIMA(2,1,2)x$(1,1,1)_{12}$ มีทั้งสิ้น 16 กรณีย่อย

<u>แบบจำลองที่ 1</u> **ARIMA(1,1,1)x$(0,1,1)_{12}$**

ตารางที่ 4.1 ตารางแสดงค่าเฉลี่ยของ RMSE ของชุดข้อมูลทดสอบที่ได้จากการจำลองชุดข้อมูลอนุกรมเวลาด้วยตัวแบบ ARIMA(1,1,1)x$(0,1,1)_{12}$ โดยที่ $\Theta = -0.5, 0.5$, $\phi = -0.5, 0.5$, $\theta = -0.5, 0.5$ ซึ่งมีทั้งหมด 8 กรณี เป็นดังนี้

MODEL	กรณีที่	Φ	Θ	ϕ	θ	ค่าเฉลี่ยของ RMSE		
						SARIMA	SARIMA+ANN	SARIMA+SVM
ARIMA(1,1,1)x$(0,1,1)_{12}$	1	-	-0.5	-0.5	-0.5	3.3285	1.4391	1.4518
	2	-	-0.5	-0.5	0.5	2.5580	1.5541	1.5618
	3	-	-0.5	0.5	-0.5	2.4483	1.5840	1.5870
	4	-	-0.5	0.5	0.5	3.1483	1.9786	1.9870
	5	-	0.5	-0.5	-0.5	3.3084	1.7651	1.7753
	6	-	0.5	-0.5	0.5	2.2862	1.8857	1.8886
	7	-	0.5	0.5	-0.5	2.2610	1.9303	1.9345
	8	-	0.5	0.5	0.5	2.8872	2.3349	2.3443

**<u>หมายเหตุ</u> ช่องที่ระบายสีคือตัวแบบที่เหมาะสมที่สุด เมื่อใช้ค่าเฉลี่ยของ RMSE เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ

สรุปได้ว่า การพยากรณ์สำหรับแบบจำลอง ARIMA(1,1,1)x$(0,1,1)_{12}$ ในชุดข้อมูลทดสอบเมื่อใช้รากของค่าคลาดเคลื่อนกำลังสองเฉลี่ย(Root Mean Square Error: RMSE) เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ ตัวแบบผสมระหว่าง ARIMA(1,1,1)x$(0,1,1)_{12}$ กับตัวแบบโครงข่ายประสาทเทียม ให้ค่าพยากรณที่แม่นยำที่สุด<u>สำหรับทุกกรณีของค่าพารามิเตอร์</u> เนื่องจากมีค่า RMSE ที่ต่ำที่สุด รองลงมาคือตัวแบบผสมระหว่าง ARIMA(1,1,1)x(0,1,1)12 กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน และตัวแบบ ARIMA(1,1,1)x$(0,1,1)_{12}$ มีความแม่นยำในการพยากรณ์ต่ำที่สุด

<u>แบบจำลองที่ 2</u> **ARIMA(1,1,2)x(0,1,1)$_{12}$**

ตารางที่ 4.2 ตารางแสดงค่าเฉลี่ยของ RMSE ของชุดข้อมูลทดสอบที่ได้จากการจำลองชุดข้อมูลอนุกรมเวลาด้วยตัวแบบ ARIMA(1,1,2)x(0,1,1)$_{12}$ โดยที่ $\Theta = -0.5, 0.5$, $\phi = -0.5, 0.5$, $\theta_1 = -0.5, 0.5$, $\theta_2 = -0.5$ ซึ่งมีทั้งหมด 8 กรณี เป็นดังนี้

MODEL	กรณีที่	Φ	Θ	ϕ	θ_1	θ_2	ค่าเฉลี่ยของ RMSE		
							SARIMA	SARIMA+ANN	SARIMA+SVM
ARIMA(1,1,2)x(0,1,1)$_{12}$	1	-	-0.5	-0.5	-0.5	-0.5	3.2836	1.4363	1.4413
	2	-	-0.5	-0.5	0.5	-0.5	2.7625	1.5898	1.5970
	3	-	-0.5	0.5	-0.5	-0.5	2.7018	1.6108	1.6202
	4	-	-0.5	0.5	0.5	-0.5	2.8217	1.9587	1.9741
	5	-	0.5	-0.5	-0.5	-0.5	3.1720	1.7123	1.7130
	6	-	0.5	-0.5	0.5	-0.5	2.5511	1.9151	1.9161
	7	-	0.5	0.5	-0.5	-0.5	2.4909	1.9211	1.9353
	8	-	0.5	0.5	0.5	-0.5	2.6901	2.3400	2.3424

**<u>หมายเหตุ</u> ช่องที่ระบายสีคือตัวแบบที่เหมาะสมที่สุด เมื่อใช้ค่าเฉลี่ยของ RMSE เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ

สรุปได้ว่า การพยากรณ์สำหรับแบบจำลอง ARIMA(1,1,2)x(0,1,1)$_{12}$ ในชุดข้อมูลทดสอบเมื่อใช้รากของค่าคลาดเคลื่อนกำลังสองเฉลี่ย (Root Mean Square Error: RMSE) เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ ตัวแบบผสมระหว่าง ARIMA(1,1,2)x(0,1,1)$_{12}$ กับตัวแบบโครงข่ายประสาทเทียมให้ค่าพยากรณ์ที่แม่นยำที่สุด<u>สำหรับทุกกรณีของค่าพารามิเตอร์</u> เนื่องจากมีค่า RMSE ที่ต่ำที่สุด รองลงมาคือตัวแบบผสมระหว่าง ARIMA(1,1,2)x(0,1,1)$_{12}$ กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน และตัวแบบ ARIMA(1,1,2)x(0,1,1)$_{12}$ มีความแม่นยำในการพยากรณ์ต่ำที่สุด

<u>แบบจำลองที่ 3</u> ARIMA(2,1,1)x(0,1,1)$_{12}$

ตารางที่ 4.3 ค่าเฉลี่ยของ RMSE ของชุดข้อมูลทดสอบที่ได้จากการจำลองชุดข้อมูลอนุกรมเวลาด้วยตัวแบบ ARIMA(2,1,1)x(0,1,1)$_{12}$ โดยที่ $\Theta = -0.5, 0.5$, $\phi_1 = -0.5, 0.5$, $\phi_2 = -0.5$, $\theta = -0.5, 0.5$ ซึ่งมีทั้งหมด 8 กรณี เป็นดังนี้

MODEL	กรณีที่	Φ	Θ	ϕ_1	ϕ_2	θ	ค่าเฉลี่ยของ RMSE		
							SARIMA	SARIMA+ANN	SARIMA+SVM
ARIMA(2,1,1)x(0,1,1)$_{12}$	1	-	-0.5	-0.5	-0.5	-0.5	3.4250	1.4503	1.4624
	2	-	-0.5	-0.5	-0.5	0.5	2.8465	1.5731	1.5687
	3	-	-0.5	0.5	-0.5	-0.5	2.7577	1.5772	1.5885
	4	-	-0.5	0.5	-0.5	0.5	3.0986	1.9602	1.9675
	5	-	0.5	-0.5	-0.5	-0.5	3.3090	1.7864	1.7851
	6	-	0.5	-0.5	-0.5	0.5	2.5610	1.9798	1.9805
	7	-	0.5	0.5	-0.5	-0.5	3.1538	1.9315	1.9488
	8	-	0.5	0.5	-0.5	0.5	2.9205	2.4484	2.4558

**<u>หมายเหตุ</u> ช่องที่ระบายสีคือตัวแบบที่เหมาะสมที่สุด เมื่อใช้ค่าเฉลี่ยของ RMSE เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ

สรุปได้ว่า การพยากรณ์สำหรับแบบจำลอง ARIMA(2,1,1)x(0,1,1)$_{12}$ ในชุดข้อมูลทดสอบเมื่อใช้รากของค่าคลาดเคลื่อนกำลังสองเฉลี่ย(Root Mean Square Error: RMSE) เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ ตัวแบบผสมระหว่าง ARIMA(2,1,1)x(0,1,1)$_{12}$ กับตัวแบบโครงข่ายประสาทเทียม ให้ค่าพยากรณ์ที่แม่นยำที่สุด เนื่องจากมีค่า RMSE ที่ต่ำที่สุด รองลงมาคือตัวแบบผสมระหว่าง ARIMA(2,1,1)x(0,1,1)$_{12}$ กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน และตัวแบบ ARIMA(2,1,1)x(0,1,1)$_{12}$ มีความแม่นยำในการพยากรณ์ต่ำที่สุด **<u>ยกเว้นเพียง 2 กรณี</u>** คือ กรณีที่ 2 ซึ่งมีพารามิเตอร์ Θ = -0.5, ϕ_1 = -0.5, ϕ_2 =-0.5, θ = 0.5 และกรณีที่ 5 ที่มีพารามิเตอร์ Θ = 0.5, ϕ_1 = -0.5, ϕ_2 = -0.5, θ = -0.5 ที่ตัวแบบผสมระหว่าง ARIMA(2,1,1)x(0,1,1)$_{12}$ กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีนให้ค่าพยากรณ์ที่แม่นยำที่สุด รองลงมาคือตัวแบบผสมระหว่าง ARIMA(2,1,1)x(0,1,1)$_{12}$ กับตัวแบบโครงข่ายประสาทเทียม และตัวแบบ ARIMA(2,1,1)x(0,1,1)$_{12}$ มีความแม่นยำในการพยากรณ์ต่ำที่สุด

<u>แบบจำลองที่ 4</u> **ARIMA(2,1,2)x(0,1,1)$_{12}$**

ตารางที่ 4.4 ตารางแสดงค่าเฉลี่ยของ RMSE ของชุดข้อมูลทดสอบที่ได้จากการจำลองชุดข้อมูลอนุกรมเวลาด้วยตัวแบบ ARIMA(2,1,2)x(0,1,1)$_{12}$ โดยที่ $\Theta = -0.5, 0.5$, $\phi_1 = -0.5, 0.5$, $\phi_2 = -0.5$, $\theta_1 = -0.5, 0.5$, $\theta_2 = -0.5$ ซึ่งมีทั้งหมด 8 กรณี เป็นดังนี้

MODEL	กรณีที่	Φ	Θ	ϕ_1	ϕ_2	θ_1	θ_2	ค่าเฉลี่ยของ RMSE		
								SARIMA	SARIMA+ANN	SARIMA+SVM
ARIMA(2,1,2)x(0,1,1)$_{12}$	1	-	-0.5	-0.5	-0.5	-0.5	-0.5	3.7933	1.4538	1.4501
	2	-	-0.5	-0.5	-0.5	0.5	-0.5	3.4214	1.6897	1.6957
	3	-	-0.5	0.5	-0.5	-0.5	-0.5	3.3806	1.6352	1.6479
	4	-	-0.5	0.5	-0.5	0.5	-0.5	3.5256	2.0017	2.0041
	5	-	0.5	-0.5	-0.5	-0.5	-0.5	3.7178	1.7397	1.7413
	6	-	0.5	-0.5	-0.5	0.5	-0.5	3.2993	1.9874	1.9914
	7	-	0.5	0.5	-0.5	-0.5	-0.5	3.3392	1.9270	1.9329
	8	-	0.5	0.5	-0.5	0.5	-0.5	3.3592	2.3854	2.3978

**<u>หมายเหตุ</u> ช่องที่ระบายสีคือตัวแบบที่เหมาะสมที่สุด เมื่อใช้ค่าเฉลี่ยของ RMSE เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ

สรุปได้ว่า การพยากรณ์สำหรับแบบจำลอง ARIMA(2,1,2)x(0,1,1)$_{12}$ ในชุดข้อมูลทดสอบเมื่อใช้รากของค่าคลาดเคลื่อนกำลังสองเฉลี่ย(Root Mean Square Error: RMSE) เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ ตัวแบบผสมระหว่าง ARIMA(2,1,2)x(0,1,1)$_{12}$ กับตัวแบบโครงข่ายประสาทเทียม ให้ค่าพยากรณ์ที่แม่นยำที่สุด เนื่องจากมีค่า RMSE ที่ต่ำที่สุด รองลงมาคือตัวแบบผสมระหว่าง ARIMA(2,1,2)x(0,1,1)$_{12}$ กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน และตัวแบบ ARIMA(2,1,2)x(0,1,1)$_{12}$ มีความแม่นยำในการพยากรณ์ต่ำที่สุด **<u>ยกเว้นเพียง 1 กรณี</u>** คือ กรณีที่ 1 ซึ่งมีพารามิเตอร์ Θ = -0.5 , ϕ_1 = -0.5, ϕ_2 = -0.5, θ_1 = -0.5 และ θ_2 = -0.5 ที่ตัวแบบผสมระหว่าง ARIMA(2,1,2)x(0,1,1)$_{12}$ กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีนให้ค่าพยากรณ์ที่แม่นยำที่สุด รองลงมาคือตัวแบบผสมระหว่าง ARIMA(2,1,2)x(0,1,1)$_{12}$ กับตัวแบบโครงข่ายประสาทเทียม และตัวแบบ ARIMA(2,1,2)x(0,1,1)$_{12}$ มีความแม่นยำในการพยากรณ์ต่ำที่สุด

<u>แบบจำลองที่ 5</u> ARIMA(1,1,1)x(1,1,0)$_{12}$

ตารางที่ 4.5 ตารางแสดงค่าเฉลี่ยของ RMSE ของชุดข้อมูลทดสอบที่ได้จากการจำลองชุดข้อมูลอนุกรมเวลาด้วยตัวแบบ ARIMA(1,1,1)x(1,1,0)$_{12}$ โดยที่ $\Phi = -0.5, 0.5$, $\phi = -0.5, 0.5$, $\theta = -0.5, 0.5$ ซึ่งมีทั้งหมด 8 กรณี เป็นดังนี้

MODEL	กรณีที่	Φ	Θ	ϕ	θ	ค่าเฉลี่ยของ RMSE		
						SARIMA	SARIMA+ANN	SARIMA+SVM
ARIMA(1,1,1)x(1,1,0)$_{12}$	1	-0.5	-	-0.5	-0.5	3.4984	1.5009	1.5053
	2	-0.5	-	-0.5	0.5	2.4722	1.6106	1.6100
	3	-0.5	-	0.5	-0.5	2.5032	1.6072	1.6164
	4	-0.5	-	0.5	0.5	3.0982	2.0159	2.0224
	5	0.5	-	-0.5	-0.5	3.3847	1.7901	1.8005
	6	0.5	-	-0.5	0.5	2.3851	1.9738	1.9831
	7	0.5	-	0.5	-0.5	2.3452	1.9184	1.9248
	8	0.5	-	0.5	0.5	2.9680	2.4583	2.4658

**<u>หมายเหตุ</u> ช่องที่ระบายสีคือตัวแบบที่เหมาะสมที่สุด เมื่อใช้ค่าเฉลี่ยของ RMSE เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ

สรุปได้ว่า การพยากรณ์สำหรับแบบจำลอง ARIMA(1,1,1)x(1,1,0)$_{12}$ ในชุดข้อมูลทดสอบเมื่อใช้รากของค่าคลาดเคลื่อนกำลังสองเฉลี่ย(Root Mean Square Error: RMSE) เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ ตัวแบบผสมระหว่าง ARIMA(1,1,1)x(1,1,0)$_{12}$ กับตัวแบบโครงข่ายประสาทเทียมให้ค่าพยากรณ์ที่แม่นยำที่สุด เนื่องจากมีค่า RMSE ที่ต่ำที่สุด รองลงมาคือตัวแบบผสมระหว่าง ARIMA(1,1,1)x(1,1,0)$_{12}$ กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน และตัวแบบ ARIMA(1,1,1)x(1,1,0)$_{12}$ มีความแม่นยำในการพยากรณ์ต่ำที่สุด **<u>ยกเว้นเพียง 1 กรณี</u>** คือ กรณีที่2 ซึ่งมีพารามิเตอร์ Φ = -0.5, ϕ = -0.5, θ = 0.5 ที่ตัวแบบผสมระหว่าง ARIMA(1,1,1)x(1,1,0)$_{12}$ กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีนให้ค่าพยากรณ์ที่แม่นยำที่สุด รองลงมาคือตัวแบบผสมระหว่าง ARIMA(1,1,1)x(1,1,0)$_{12}$ กับตัวแบบโครงข่ายประสาทเทียม และตัวแบบ ARIMA(1,1,1)x(1,1,0)$_{12}$ มีความแม่นยำในการพยากรณ์ต่ำที่สุด

<u>แบบจำลองที่ 6</u> ARIMA(1,1,2)x(1,1,0)$_{12}$

ตารางที่ 4.6 ตารางแสดงค่าเฉลี่ยของ RMSE ของชุดข้อมูลทดสอบที่ได้จากการจำลองชุดข้อมูลอนุกรมเวลาด้วยตัวแบบ ARIMA(1,1,2)x(1,1,0)$_{12}$ โดยที่ $\Phi = -0.5, 0.5$, $\phi = -0.5, 0.5$, $\theta_1 = -0.5, 0.5$, $\theta_2 = -0.5$ ซึ่งมีทั้งหมด 8 กรณี เป็นดังนี้

MODEL	กรณีที่	Φ	Θ	ϕ	θ_1	θ_2	ค่าเฉลี่ยของ RMSE		
							SARIMA	SARIMA+ANN	SARIMA+SVM
ARIMA(1,1,2)x(1,1,0)$_{12}$	1	-0.5	-	-0.5	-0.5	-0.5	3.2517	1.4645	1.4691
	2	-0.5	-	-0.5	0.5	-0.5	2.7145	1.6489	1.6525
	3	-0.5	-	0.5	-0.5	-0.5	2.7067	1.6648	1.6718
	4	-0.5	-	0.5	0.5	-0.5	2.9017	1.9975	2.0020
	5	0.5	-	-0.5	-0.5	-0.5	3.1780	1.7431	1.7557
	6	0.5	-	-0.5	0.5	-0.5	2.5748	1.9458	1.9607
	7	0.5	-	0.5	-0.5	-0.5	2.5745	1.9535	1.9572
	8	0.5	-	0.5	0.5	-0.5	2.7040	2.3462	2.3631

**<u>หมายเหตุ</u> ช่องที่ระบายสีคือตัวแบบที่เหมาะสมที่สุด เมื่อใช้ค่าเฉลี่ยของ RMSE เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัว

สรุปได้ว่า การพยากรณ์สำหรับแบบจำลอง ARIMA(1,1,2)x(1,1,0)$_{12}$ ในชุดข้อมูลทดสอบเมื่อใช้รากของค่าคลาดเคลื่อนกำลังสองเฉลี่ย(Root Mean Square Error: RMSE) เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ ตัวแบบผสมระหว่าง ARIMA(1,1,2)x(1,1,0)$_{12}$ กับตัวแบบโครงข่ายประสาทเทียมให้ค่าพยากรณ์ที่แม่นยำที่สุด<u>สำหรับทุกกรณีของค่าพารามิเตอร์</u> เนื่องจากมีค่า RMSE ที่ต่ำที่สุด รองลงมาคือตัวแบบผสมระหว่าง ARIMA(1,1,2)x(1,1,0)$_{12}$ กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน และตัวแบบ ARIMA(1,1,2)x(1,1,0)$_{12}$ มีความแม่นยำในการพยากรณ์ต่ำที่สุด

<u>แบบจำลองที่ 7</u> ARIMA(2,1,1)x(1,1,0)$_{12}$

ตารางที่ 4.7 ตารางแสดงค่าเฉลี่ยของ RMSE ของชุดข้อมูลทดสอบที่ได้จากการจำลองชุดข้อมูลอนุกรมเวลาด้วยตัวแบบ ARIMA(2,1,1)x(1,1,0)$_{12}$ โดยที่ $\Phi = -0.5, 0.5$, $\phi_1 = -0.5, 0.5$, $\phi_2 = -0.5$, $\theta = -0.5, 0.5$ ซึ่งมีทั้งหมด 8 กรณี เป็นดังนี้

MODEL	กรณีที่	Φ	Θ	ϕ_1	ϕ_2	θ	ค่าเฉลี่ยของ RMSE		
							SARIMA	SARIMA+ANN	SARIMA+SVM
ARIMA(2,1,1)x(1,1,0)$_{12}$	1	-0.5	-	-0.5	-0.5	-0.5	3.4650	1.4526	1.4555
	2	-0.5	-	-0.5	-0.5	0.5	2.7651	1.6149	1.6203
	3	-0.5	-	0.5	-0.5	-0.5	2.7948	1.6092	1.6137
	4	-0.5	-	0.5	-0.5	0.5	3.1916	2.0376	2.0461
	5	0.5	-	-0.5	-0.5	-0.5	3.4251	1.7674	1.7785
	6	0.5	-	-0.5	-0.5	0.5	2.6697	1.9665	1.9797
	7	0.5	-	0.5	-0.5	-0.5	2.6596	1.9238	1.9258
	8	0.5	-	0.5	-0.5	0.5	2.9545	2.4267	2.4328

**<u>หมายเหตุ</u> ช่องที่ระบายสีคือตัวแบบที่เหมาะสมที่สุด เมื่อใช้ค่าเฉลี่ยของ RMSE เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ

สรุปได้ว่า การพยากรณ์สำหรับแบบจำลอง ARIMA(2,1,1)x(1,1,0)$_{12}$ ในชุดข้อมูลทดสอบเมื่อใช้รากของค่าคลาดเคลื่อนกำลังสองเฉลี่ย(Root Mean Square Error: RMSE) เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ ตัวแบบผสมระหว่าง ARIMA(2,1,1)x(1,1,0)$_{12}$ กับตัวแบบโครงข่ายประสาทเทียมให้ค่าพยากรณ์ที่แม่นยำที่สุด<u>สำหรับทุกกรณีของค่าพารามิเตอร์</u> เนื่องจากมีค่า RMSE ที่ต่ำที่สุด รองลงมาคือตัวแบบผสมระหว่าง ARIMA(2,1,1)x(1,1,0)$_{12}$ กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน และตัวแบบ ARIMA(2,1,1)x(1,1,0)$_{12}$ มีความแม่นยำในการพยากรณ์ต่ำที่สุด

<u>แบบจำลองที่ 8</u> ARIMA(2,1,2)x(1,1,0)$_{12}$

ตารางที่ 4.8 ตารางแสดงค่าเฉลี่ยของ RMSE ของชุดข้อมูลทดสอบที่ได้จากการจำลองชุดข้อมูลอนุกรมเวลาด้วยตัวแบบ ARIMA(2,1,2)x(1,1,0)$_{12}$ โดยที่ $\Phi = -0.5, 0.5$, $\phi_1 = -0.5, 0.5$, $\phi_2 = -0.5$, $\theta_1 = -0.5, 0.5$, $\theta_2 = -0.5$ ซึ่งมีทั้งหมด 8 กรณี เป็นดังนี้

MODEL	กรณีที่	Φ	Θ	ϕ_1	ϕ_2	θ_1	θ_2	ค่าเฉลี่ยของ RMSE		
								SARIMA	SARIMA+ANN	SARIMA+SVM
ARIMA(2,1,2)x(1,1,0)$_{12}$	1	-0.5	-	-0.5	-0.5	-0.5	-0.5	3.8623	1.4867	1.4980
	2	-0.5	-	-0.5	-0.5	0.5	-0.5	3.4724	1.7206	1.7254
	3	-0.5	-	0.5	-0.5	-0.5	-0.5	3.4646	1.7827	1.7882
	4	-0.5	-	0.5	-0.5	0.5	-0.5	3.5238	2.0650	2.0743
	5	0.5	-	-0.5	-0.5	-0.5	-0.5	3.7547	1.7795	1.7917
	6	0.5	-	-0.5	-0.5	0.5	-0.5	3.2993	2.0760	2.0673
	7	0.5	-	0.5	-0.5	-0.5	-0.5	3.3177	1.9891	1.9998
	8	0.5	-	0.5	-0.5	0.5	-0.5	3.3437	2.4433	2.4464

**<u>หมายเหตุ</u> ช่องที่ระบายสีคือตัวแบบที่เหมาะสมที่สุด เมื่อใช้ค่าเฉลี่ยของ RMSE เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ

สรุปได้ว่า การพยากรณ์สำหรับแบบจำลอง ARIMA(2,1,2)x(1,1,0)$_{12}$ ในชุดข้อมูลทดสอบเมื่อใช้รากของค่าคลาดเคลื่อนกำลังสองเฉลี่ย(Root Mean Square Error: RMSE) เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ ตัวแบบผสมระหว่าง ARIMA(2,1,2)x(1,1,0)$_{12}$ กับตัวแบบโครงข่ายประสาทเทียมให้ค่าพยากรณ์ที่แม่นยำที่สุด เนื่องจากมีค่า RMSE ที่ต่ำที่สุด รองลงมาคือตัวแบบผสมระหว่าง ARIMA(2,1,2)x(1,1,0)$_{12}$ กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน และตัวแบบ ARIMA(2,1,2)x(1,1,0)$_{12}$ มีความแม่นยำในการพยากรณ์ต่ำที่สุด **<u>ยกเว้นเพียง 1 กรณี</u>** คือ กรณีที่6 ซึ่งมีพารามิเตอร์ Φ = 0.5, ϕ_1 = -0.5, ϕ_2 = -0.5, θ_1 = 0.5, θ_2 = -0.5 ที่ตัวแบบผสมระหว่าง ARIMA(2,1,2)x(1,1,0)$_{12}$ กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีนให้ค่าพยากรณ์ที่แม่นยำที่สุด รองลงมาคือตัวแบบผสมระหว่าง ARIMA(2,1,2)x(1,1,0)$_{12}$ กับตัวแบบโครงข่ายประสาทเทียม และตัวแบบ ARIMA(2,1,2)x(1,1,0)$_{12}$ มีความแม่นยำในการพยากรณ์ต่ำที่สุด

<u>แบบจำลองที่ 9</u> ARIMA(1,1,1)x(1,1,1)$_{12}$

ตารางที่ 4.9 ตารางแสดงค่าเฉลี่ยของ RMSE ของชุดข้อมูลทดสอบที่ได้จากการจำลองชุดข้อมูลอนุกรมเวลาด้วยตัวแบบ ARIMA(1,1,1)x(1,1,1)$_{12}$ โดยที่ $\Phi = -0.5, 0.5$, $\Theta = -0.5, 0.5$, $\phi = -0.5, 0.5$, $\theta = -0.5, 0.5$ ซึ่งมีทั้งหมด 16 กรณี เป็นดังนี้

MODEL	กรณีที่	Φ	Θ	ϕ	θ	ค่าเฉลี่ยของ RMSE		
						SARIMA	SARIMA+ANN	SARIMA+SVM
ARIMA(1,1,1)x(1,1,1)$_{12}$	1	-0.5	-0.5	-0.5	-0.5	4.0277	1.4346	1.4415
	2	-0.5	-0.5	-0.5	0.5	3.1648	1.5479	1.5569
	3	-0.5	-0.5	0.5	-0.5	3.1874	1.5631	1.5727
	4	-0.5	-0.5	0.5	0.5	3.7602	1.9328	1.9381
	5	-0.5	0.5	-0.5	-0.5	3.2644	1.6163	1.6225
	6	-0.5	0.5	-0.5	0.5	2.2285	1.7287	1.7402
	7	-0.5	0.5	0.5	-0.5	2.2227	1.7302	1.7347
	8	-0.5	0.5	0.5	0.5	2.9135	2.1812	2.1957

**<u>หมายเหตุ</u> ช่องที่ระบายสีคือตัวแบบที่เหมาะสมที่สุด เมื่อใช้ค่าเฉลี่ยของ RMSE เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ

MODEL	กรณีที่	Φ	Θ	ϕ	θ	ค่าเฉลี่ยของ RMSE	MODEL	กรณีที่
ARIMA(1,1,1)x(1,1,1)$_{12}$	9	0.5	-0.5	-0.5	-0.5	3.7616	3.0806	3.0850
	10	0.5	-0.5	-0.5	0.5	2.2063	1.7351	1.7403
	11	0.5	-0.5	0.5	-0.5	2.2266	1.7324	1.7383
	12	0.5	-0.5	0.5	0.5	2.9198	2.1908	2.2000
	13	0.5	0.5	-0.5	-0.5	4.5796	1.9155	1.9183
	14	0.5	0.5	-0.5	0.5	2.8404	2.2750	2.2833
	15	0.5	0.5	0.5	-0.5	2.8611	2.1734	2.1800
	16	0.5	0.5	0.5	0.5	3.3097	2.7025	2.7146

**หมายเหตุ ช่องที่ระบายสีคือตัวแบบที่เหมาะสมที่สุด เมื่อใช้ค่าเฉลี่ยของ RMSE เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ

สรุปได้ว่า การพยากรณ์สำหรับแบบจำลอง ARIMA(1,1,1)x(1,1,1)$_{12}$ ในชุดข้อมูลทดสอบเมื่อใช้รากของค่าคลาดเคลื่อนกำลังสองเฉลี่ย(Root Mean Square Error: RMSE) เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ ตัวแบบผสมระหว่าง ARIMA(1,1,1)x(1,1,1)$_{12}$ กับตัวแบบโครงข่ายประสาทเทียมให้ค่าพยากรณ์ที่แม่นยำที่สุดสำหรับทุกกรณีของค่าพารามิเตอร์ เนื่องจากมีค่า RMSE ที่ต่ำที่สุด รองลงมาคือตัวแบบผสมระหว่าง ARIMA(1,1,1)x(1,1,1)$_{12}$ กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน และตัวแบบ ARIMA(1,1,1)x(1,1,1)$_{12}$ มีความแม่นยำในการพยากรณ์ต่ำที่สุด

<u>แบบจำลองที่ 10</u> ARIMA(1,1,2)x(1,1,1)$_{12}$

ตารางที่ 4.10 ตารางแสดงค่าเฉลี่ยของ RMSE ของชุดข้อมูลทดสอบที่ได้จากการจำลองชุดข้อมูลอนุกรมเวลาด้วยตัวแบบ ARIMA(1,1,2)x(1,1,1)$_{12}$ โดยที่ $\Phi=-0.5,0.5$,$\Theta=-0.5,0.5$,$\phi=-0.5,0.5$,$\theta_1=-0.5,0.5$,$\theta_2=-0.5$ ซึ่งมีทั้งหมด 16 กรณี เป็นดังนี้

MODEL	กรณีที่	Φ	Θ	ϕ	θ_1	θ_2	ค่าเฉลี่ยของ RMSE		
							SARIMA	SARIMA+ANN	SARIMA+SVM
ARIMA(1,1,2)x(1,1,1)$_{12}$	1	-0.5	-0.5	-0.5	-0.5	-0.5	3.7134	1.3974	1.4029
	2	-0.5	-0.5	-0.5	0.5	-0.5	3.2771	1.5400	1.5440
	3	-0.5	-0.5	0.5	-0.5	-0.5	3.2767	1.5547	1.5595
	4	-0.5	-0.5	0.5	0.5	-0.5	3.4161	1.8985	1.9101
	5	-0.5	0.5	-0.5	-0.5	-0.5	3.0854	1.5299	1.5380
	6	-0.5	0.5	-0.5	0.5	-0.5	2.4469	1.7518	1.7594
	7	-0.5	0.5	0.5	-0.5	-0.5	2.4195	1.7455	1.7521
	8	-0.5	0.5	0.5	0.5	-0.5	2.6337	2.1099	2.1191

**<u>หมายเหตุ</u> ช่องที่ระบายสีคือตัวแบบที่เหมาะสมที่สุด เมื่อใช้ค่าเฉลี่ยของ RMSE เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ

MODEL	กรณีที่	Φ	Θ	ϕ	θ_1	θ_2	ค่าเฉลี่ยของ RMSE		
							SARIMA	SARIMA+ANN	SARIMA+ANN
ARIMA(1,1,2)x(1,1,1)$_{12}$	9	0.5	-0.5	-0.5	-0.5	-0.5	3.0504	1.5534	1.5628
	10	0.5	-0.5	-0.5	0.5	-0.5	2.4129	1.7543	1.7565
	11	0.5	-0.5	0.5	-0.5	-0.5	2.4416	1.7398	1.7428
	12	0.5	-0.5	0.5	0.5	-0.5	2.6314	2.1510	2.1570
	13	0.5	0.5	-0.5	-0.5	-0.5	3.5450	1.8958	1.9011
	14	0.5	0.5	-0.5	0.5	-0.5	3.0621	2.1644	2.1737
	15	0.5	0.5	0.5	-0.5	-0.5	3.0420	2.1671	2.1686
	16	0.5	0.5	0.5	0.5	-0.5	3.0700	2.6280	2.6415

**หมายเหตุ ช่องที่ระบายสีคือตัวแบบที่เหมาะสมที่สุด เมื่อใช้ค่าเฉลี่ยของ RMSE เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ

สรุปได้ว่า การพยากรณ์สำหรับแบบจำลอง ARIMA(1,1,2)x(1,1,1)$_{12}$ ในชุดข้อมูลทดสอบเมื่อใช้รากของค่าคลาดเคลื่อนกำลังสองเฉลี่ย (Root Mean Square Error: RMSE) เป็น เกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ ตัวแบบผสมระหว่าง ARIMA(1,1,2)x(1,1,1)$_{12}$ กับตัวแบบโครงข่ายประสาทเทียม ให้ค่าพยากรณ์ที่แม่นยำที่สุดสำหรับทุกกรณีของค่าพารามิเตอร์ เนื่องจากมีค่า RMSE ที่ต่ำที่สุด รองลงมาคือตัวแบบผสมระหว่าง ARIMA(1,1,2)x(1,1,1)$_{12}$ กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน และตัวแบบ ARIMA(1,1,2)x(1,1,1)$_{12}$ มีความแม่นยำในการพยากรณ์ต่ำที่สุด

แบบจำลองที่ 11 ARIMA(2,1,1)x(1,1,1)$_{12}$

ตารางที่ 4.11 ตารางแสดงค่าเฉลี่ยของ RMSE ของชุดข้อมูลทดสอบที่ได้จากการจำลองชุดข้อมูลอนุกรมเวลาด้วยตัวแบบ ARIMA(2,1,1)x(1,1,1)$_{12}$ โดยที่ $\Phi = -0.5, 0.5$, $\Theta = -0.5, 0.5$, $\phi_1 = -0.5, 0.5$, $\phi_2 = -0.5$, $\theta = -0.5, 0.5$ ซึ่งมีทั้งหมด 16 กรณี เป็นดังนี้

MODEL	กรณีที่	Φ	Θ	ϕ_1	ϕ_2	θ	ค่าเฉลี่ยของ RMSE		
							SARIMA	SARIMA+ANN	SARIMA+SVM
ARIMA(2,1,1)x(1,1,1)$_{12}$	1	-0.5	-0.5	-0.5	-0.5	-0.5	3.8898	1.4127	1.4134
	2	-0.5	-0.5	-0.5	-0.5	0.5	3.3814	1.5536	1.5613
	3	-0.5	-0.5	0.5	-0.5	-0.5	3.3298	1.5261	1.5327
	4	-0.5	-0.5	0.5	-0.5	0.5	3.6360	1.9024	1.9079
	5	-0.5	0.5	-0.5	-0.5	-0.5	3.1492	1.5674	1.5728
	6	-0.5	0.5	-0.5	-0.5	0.5	2.5352	1.7634	1.7642
	7	-0.5	0.5	0.5	-0.5	-0.5	2.5092	1.6762	1.6824
	8	-0.5	0.5	0.5	-0.5	0.5	2.8888	2.1459	2.1563

**หมายเหตุ ช่องที่ระบายสีคือตัวแบบที่เหมาะสมที่สุด เมื่อใช้ค่าเฉลี่ยของ RMSE เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ

MODEL	กรณีที่	Φ	Θ	ϕ_1	ϕ_2	θ	ค่าเฉลี่ยของ RMSE		
							SARIMA	SARIMA+ANN	SARIMA+SVM
ARIMA(2,1,1)x$(1,1,1)_{12}$	9	0.5	-0.5	-0.5	-0.5	-0.5	3.1905	2.2040	2.2074
	10	0.5	-0.5	-0.5	-0.5	0.5	3.4353	1.7767	1.7789
	11	0.5	-0.5	0.5	-0.5	-0.5	2.4941	1.6681	1.6731
	12	0.5	-0.5	0.5	-0.5	0.5	2.8211	2.1130	2.1245
	13	0.5	0.5	-0.5	-0.5	-0.5	3.7699	1.9780	1.9918
	14	0.5	0.5	-0.5	-0.5	0.5	3.1084	2.2015	2.2045
	15	0.5	0.5	0.5	-0.5	-0.5	3.1234	2.1051	2.1119
	16	0.5	0.5	0.5	-0.5	0.5	3.3436	2.6745	2.6837

**หมายเหตุ ช่องที่ระบายสีคือตัวแบบที่เหมาะสมที่สุด เมื่อใช้ค่าเฉลี่ยของ RMSE เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ

สรุปได้ว่า การพยากรณ์สำหรับแบบจำลอง ARIMA(2,1,1)x$(1,1,1)_{12}$ ในชุดข้อมูลทดสอบเมื่อใช้รากของค่าคลาดเคลื่อนกำลังสองเฉลี่ย(Root Mean Square Error: RMSE) เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ ตัวแบบผสมระหว่าง ARIMA(2,1,1)x$(1,1,1)_{12}$ กับตัวแบบโครงข่ายประสาทเทียมให้ค่าพยากรณ์ที่แม่นยำที่สุดสำหรับทุกกรณีของค่าพารามิเตอร์ เนื่องจากมีค่า RMSE ที่ต่ำที่สุด รองลงมาคือตัวแบบผสมระหว่าง ARIMA(2,1,1)x$(1,1,1)_{12}$ กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน และตัวแบบ ARIMA(2,1,1)x$(1,1,1)_{12}$ มีความแม่นยำในการพยากรณ์ต่ำที่สุด

แบบจำลองที่ 12 ARIMA(2,1,2)x(1,1,1)$_{12}$

ตารางที่ 4.12 ตารางแสดงค่าเฉลี่ยของ RMSE ของชุดข้อมูลทดสอบที่ได้จากการจำลองชุดข้อมูลอนุกรมเวลาด้วยตัวแบบ ARIMA(2,1,2)x(1,1,1)$_{12}$ โดยที่ $\Phi = -0.5, 0.5$, $\Theta = -0.5, 0.5$, $\phi_1 = -0.5, 0.5$, $\phi_2 = -0.5$, $\theta_1 = -0.5, 0.5$, $\theta_2 = -0.5$ ซึ่งมีทั้งหมด 16 กรณี เป็นดังนี้

MODEL	กรณีที่	Φ	Θ	ϕ_1	ϕ_2	θ_1	θ_2	ค่าเฉลี่ยของ RMSE		
								SARIMA	SARIMA+ANN	SARIMA+SVM
ARIMA(2,1,2)x(1,1,1)$_{12}$	1	-0.5	-0.5	-0.5	-0.5	-0.5	-0.5	4.3573	1.4524	1.4586
	2	-0.5	-0.5	-0.5	-0.5	0.5	-0.5	3.9172	1.6232	1.6247
	3	-0.5	-0.5	0.5	-0.5	-0.5	-0.5	3.9972	1.5770	1.5784
	4	-0.5	-0.5	0.5	-0.5	0.5	-0.5	4.0567	1.9470	1.9576
	5	-0.5	0.5	-0.5	-0.5	-0.5	-0.5	3.5994	1.5789	1.5875
	6	-0.5	0.5	-0.5	-0.5	0.5	-0.5	3.1798	1.8259	1.8378
	7	-0.5	0.5	0.5	-0.5	-0.5	-0.5	3.1924	1.7765	1.7833
	8	-0.5	0.5	0.5	-0.5	0.5	-0.5	3.2711	2.2025	2.2081

**หมายเหตุ ช่องที่ระบายสีคือตัวแบบที่เหมาะสมที่สุด เมื่อใช้ค่าเฉลี่ยของ RMSE เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ

MODEL	กรณีที่	Φ	Θ	ϕ_1	ϕ_2	θ_1	θ_2	ค่าเฉลี่ยของ RMSE		
								SARIMA	SARIMA+ANN	SARIMA+SVM
ARIMA(2,1,2)x(1,1,1)$_{12}$	9	0.5	-0.5	-0.5	-0.5	-0.5	-0.5	3.5804	1.5746	1.5796
	10	0.5	-0.5	-0.5	-0.5	0.5	-0.5	3.1500	1.8413	1.8509
	11	0.5	-0.5	0.5	-0.5	-0.5	-0.5	3.2690	1.7692	1.7803
	12	0.5	-0.5	0.5	-0.5	0.5	-0.5	3.2563	2.1987	2.2014
	13	0.5	0.5	-0.5	-0.5	-0.5	-0.5	4.0915	1.9848	1.9881
	14	0.5	0.5	-0.5	-0.5	0.5	-0.5	3.6685	2.2869	2.2998
	15	0.5	0.5	0.5	-0.5	-0.5	-0.5	3.7722	2.1903	2.1999
	16	0.5	0.5	0.5	-0.5	0.5	-0.5	3.6146	3.3570	3.3676

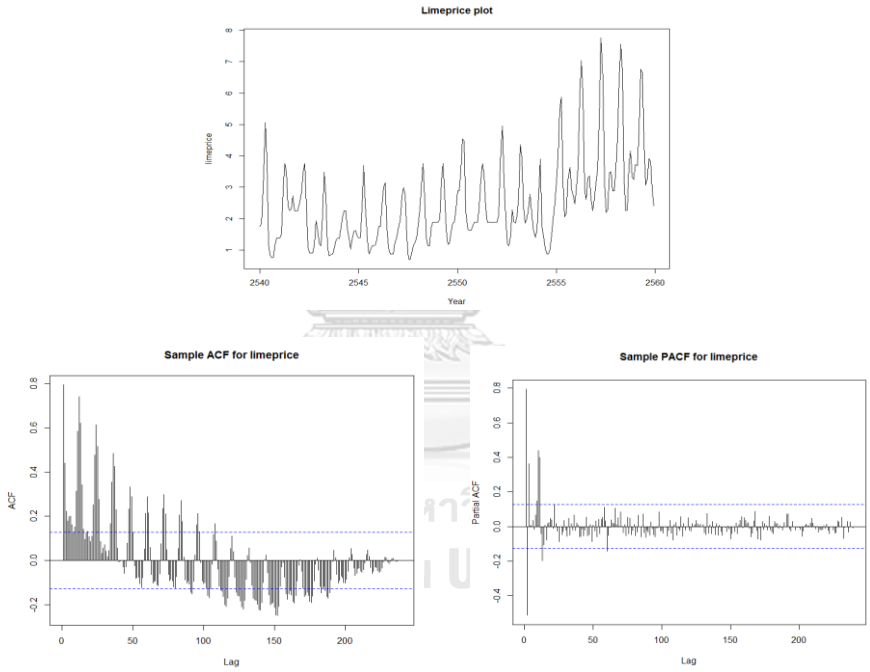
**หมายเหตุ ช่องที่ระบายสีคือตัวแบบที่เหมาะสมที่สุด เมื่อใช้ค่าเฉลี่ยของ RMSE เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ

สรุปได้ว่า การพยากรณ์สำหรับแบบจำลอง ARIMA(2,1,2)x(1,1,1)$_{12}$ ในชุดข้อมูลทดสอบเมื่อใช้รากของค่าคลาดเคลื่อนกำลังสองเฉลี่ย(Root Mean Square Error: RMSE) เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ ตัวแบบผสมระหว่าง ARIMA(2,1,2)x(1,1,1)$_{12}$ กับตัวแบบโครงข่ายประสาทเทียมให้ค่าพยากรณ์ที่แม่นยำที่สุดสำหรับทุกกรณีของค่าพารามิเตอร์ เนื่องจากมีค่า RMSE ที่ต่ำที่สุด รองลงมาคือตัวแบบผสมระหว่าง ARIMA(2,1,2)x(1,1,1)$_{12}$ กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน และตัวแบบ ARIMA(2,1,2)x(1,1,1)$_{12}$ มีความแม่นยำในการพยากรณ์ต่ำที่สุด

4.2 ผลการเปรียบเทียบความแม่นยำของค่าพยากรณ์ที่ได้จากตัวแบบ SARIMA, ตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบโครงข่ายประสาทเทียม(SARIMA+ANN) และตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน(SARIMA+SVM) โดยใช้ชุดข้อมูลจริง

ในขั้นต้นจะทำการวิเคราะห์ลักษณะเบื้องต้นของข้อมูลจริงราคาขายปลีกมะนาวก่อน จากนั้นจะนำเสนอผลการเปรียบเทียบความแม่นยำของค่าพยากรณ์ของตัวแบบสำหรับข้อมูลจริงในลำดับถัดไป

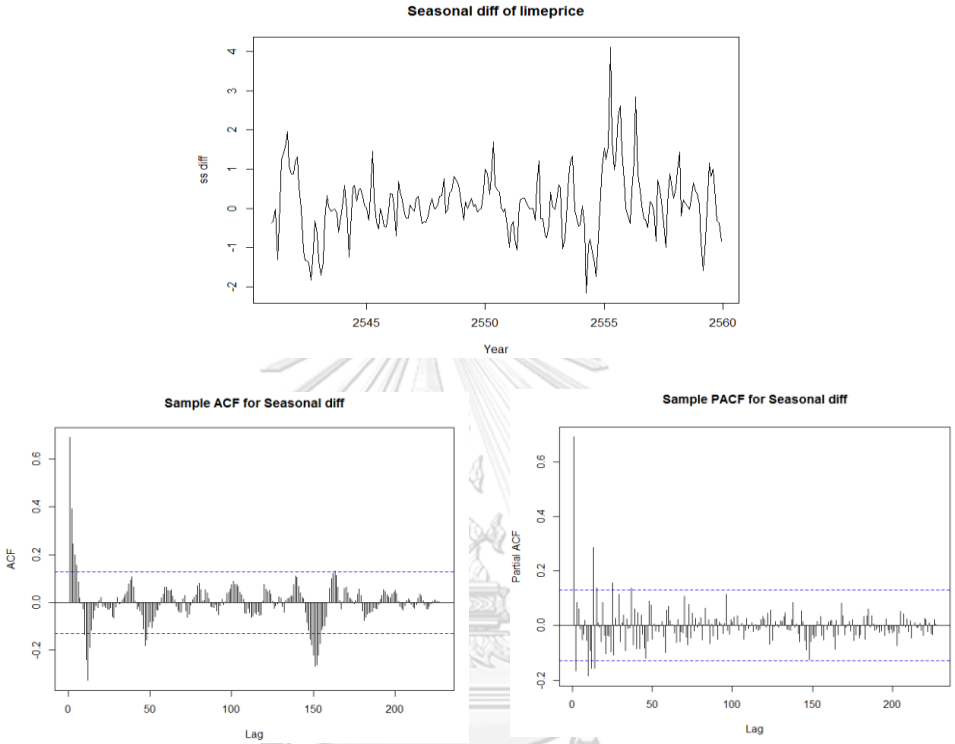
พิจารณาข้อมูลจริงราคาขายปลีกมะนาวโดยการสร้างกราฟ Time Series เพื่อดูลักษณะเบื้องต้นของข้อมูล



ภาพที่ 4.1 Time plot , ACF และ PACF plots สำหรับข้อมูลราคาขายปลีกมะนาว

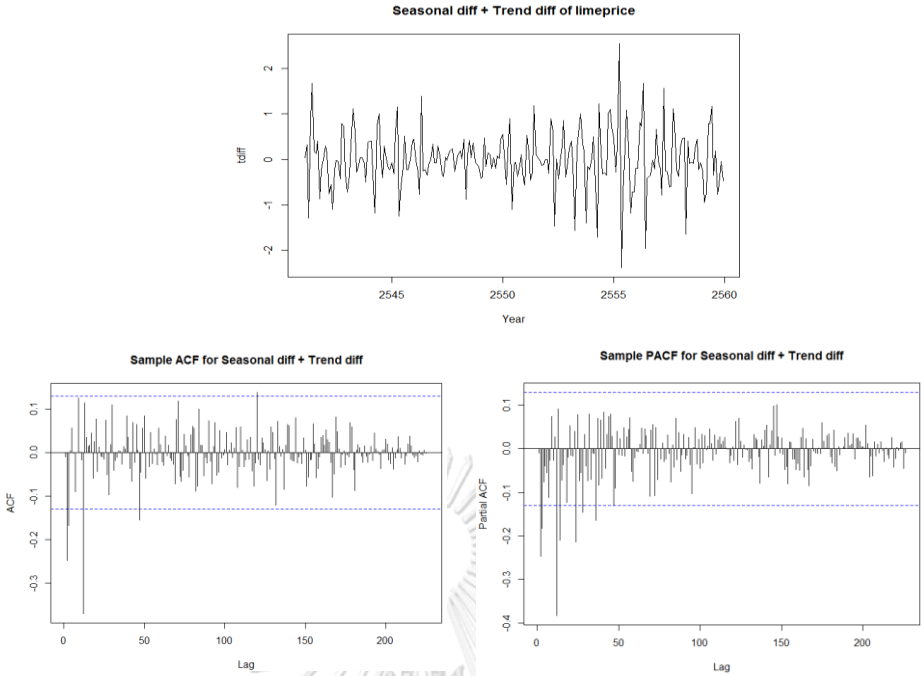
จากภาพที่ 4.1 เมื่อพิจารณาลักษณะเบื้องต้นของกราฟพล็อตของข้อมูลราคาขายปลีกมะนาว จะพบว่าข้อมูลมีลักษณะของฤดูกาล(seasonal) และมีส่วนประกอบของแนวโน้มที่เพิ่มขึ้น (Increasing trend) โดยราคาขายปลีกมะนาวจะสูงขึ้นในช่วงประมาณเดือนมีนาคม-เมษายนของทุกๆปี เนื่องจากในช่วงหน้าแล้งสภาพอากาศร้อนจัดส่งผลให้มะนาวออกลูกน้อย ทำให้มีผลผลิตออกสู่ตลาดน้อยลง และเมื่อพิจารณากราฟฟังก์ชันสหสัมพันธ์ในตัวเอง(Sample ACF)ของข้อมูลราคาขายปลีกมะนาวก็พบว่ามีความสัมพันธ์ที่เป็นฤดูกาล(period =12 เดือน)ค่อนข้างชัดเจน จึงทำการ

กำจัดความผันแปรทางฤดูกาลด้วยวิธีการหาผลต่างของฤดูกาล(Seasonal Differencing) 1 ครั้ง โดยใช้คาบ(s)เท่ากับ 12 โดยการคำนวณค่า $\nabla_{12}X_t = X_t - X_{t-12}$ จากนั้นนำข้อมูลที่ผ่านการกำจัดความผันแปรทางฤดูกาลแล้วมาสร้างกราฟ Time Series อีกครั้งหนึ่งดังภาพที่ 4.2



ภาพที่ 4.2 Time plot , ACF และ PACF plots ของข้อมูลราคาขายปลีกมะนาวภายหลังจากการปรับข้อมูลอนุกรมเวลาให้คงที่ด้วยการผลต่างของฤดูกาล(Seasonal differencing) อันดับที่ 1(D=1)

พบว่ากราฟพล็อตที่ได้ยังมีลักษณะไม่คงที่(stationary)นัก จึงทำการปรับข้อมูลให้คงที่มากขึ้นโดยทำการหาผลต่าง(Differencing) 1 ครั้ง เพื่อกำจัด trend โดยคำนวณค่า $\nabla X_t = X_t - X_{t-1}$ ซึ่งหลังจากที่ผ่านการปรับข้อมูลด้วยการหาผลต่าง(Differencing)แล้วพบว่ากราฟพล็อตที่ได้มีลักษณะคงที่(Stationary)มากขึ้นดังภาพที่ 4.3



ภาพที่ 4.3 Time plot , ACF และ PACF plots ของข้อมูลราคาขายปลีกมะนาว ภายหลังจากการปรับข้อมูลอนุกรมเวลาให้คงที่ด้วยการผลต่างของฤดูกาล(Seasonal differencing) อันดับที่ 1(D=1) และการหาผลต่าง(Differencing)อันดับที่ 1(d=1)

จากนั้นจะทำการศึกษาและเปรียบเทียบตัวแบบอนุกรมเวลาที่เหมาะสมสำหรับข้อมูลราคาขายปลีกมะนาวจริง ภายใต้ขอบเขตของการวิจัย คือ ศึกษาตัวแบบ ARIMA(p,1,q)x $(P,1,Q)_s$ ภายใต้ค่า P=0-1 ,Q=0-1 ,p=1-2 และ q=1-2 กล่าวคือจะทำการเปรียบเทียบตัวแบบทั้งหมดจำนวน 12 ตัวแบบ แล้วจึงพิจารณาคัดเลือกตัวแบบที่เหมาะสมที่สุดสำหรับข้อมูลจริงราคาขายปลีกมะนาว พบว่าตัวแบบที่เหมาะสมที่สุดคือ ตัวแบบ ARIMA(1,1,2)x$(0,1,1)_{12}$ เนื่องจากมีค่า AIC ต่ำที่สุด ดังแสดงในตารางที่ 4.13

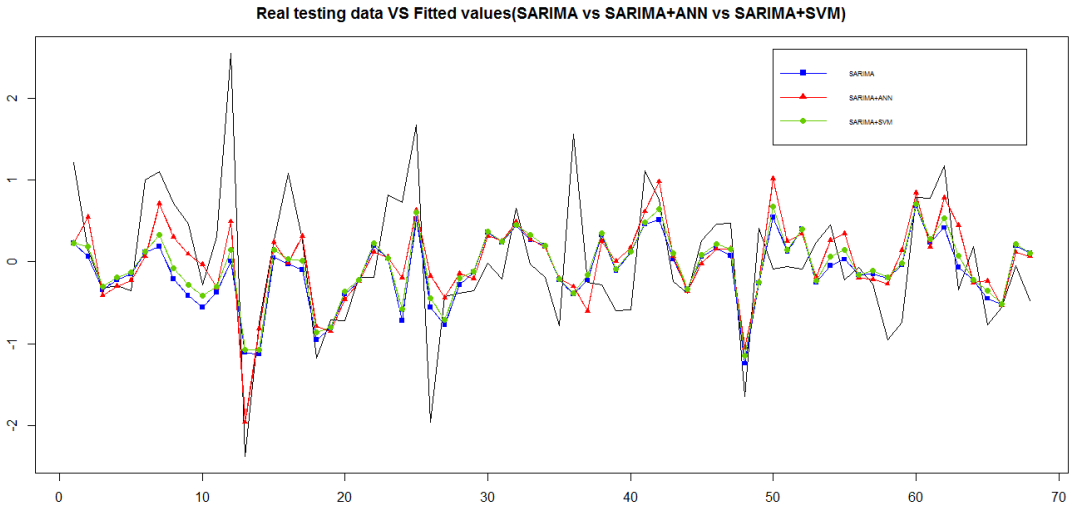
MODEL	AIC
ARIMA(1,1,1)x$(0,1,1)_{12}$	218.1080
ARIMA(1,1,2)x$(0,1,1)_{12}$	195.1267
ARIMA(2,1,1)x$(0,1,1)_{12}$	196.6899
ARIMA(2,1,2)x$(0,1,1)_{12}$	196.9419
ARIMA(1,1,1)x$(1,1,0)_{12}$	234.1493

MODEL	AIC
ARIMA(1,1,2)x$(1,1,0)_{12}$	213.2431
ARIMA(2,1,1)x$(1,1,0)_{12}$	217.0009
ARIMA(2,1,2)x$(1,1,0)_{12}$	215.0224
ARIMA(1,1,1)x$(1,1,1)_{12}$	220.0609
ARIMA(1,1,2)x$(1,1,1)_{12}$	197.0611
ARIMA(2,1,1)x$(1,1,1)_{12}$	198.5064
ARIMA(2,1,2)x$(1,1,1)_{12}$	198.8272

ตารางที่ 4.13 ตารางแสดงค่า AIC ของตัวแบบ ARIMA(p,1,q)x $(P,1,Q)_s$ ภายใต้ค่า P=0-1 ,Q=0-1 ,p=1-2 และ q=1-2

MODEL	ค่า RMSE		
	SARIMA	SARIMA+ANN	SARIMA+SVM
ARIMA(1,1,2)x$(0,1,1)_{12}$	0.6828	0.6242	0.6639

ตารางที่ 4.14 ตารางแสดงค่า RMSE ของชุดข้อมูลทดสอบที่ได้จากชุดข้อมูลจริง ราคาขายปลีกมะนาว



ภาพที่ 4.4 ค่าพยากรณ์ที่ได้จากตัวแบบ SARIMA ,ตัวแบบผสมระหว่าง SARIMA กับ ANN และตัวแบบผสมระหว่าง SARIMA กับ SVM ในชุดข้อมูลทดสอบของข้อมูลจริง

จากตารางที่ 4.13 การศึกษาตัวแบบ ARIMA ที่มีฤดูกาล(SARIMA) ภายใต้ค่า P=0-1, Q=0-1, p=1-2 และ q=1-2 พบว่าตัวแบบที่เหมาะสมที่สุดสำหรับใช้พยากรณ์ราคาขายปลีกมะนาว คือ ตัวแบบ ARIMA(1,1,2)x(0,1,1)$_{12}$ เนื่องจากมีค่า AIC ที่ต่ำที่สุด ซึ่งมีรูปแบบของสมการพยากรณ์คือ $X_t = -0.0003 + 1.3211X_{t-1} - 0.3211X_{t-2} + X_{t-12} - 1.3211X_{t-13} + 0.3211X_{t-14} + W_t - 0.3892W_{t-1} - 0.4646W_{t-2} - 0.7365W_{t-12} + 0.2866W_{t-13} + 0.3421W_{t-14}$ โดยให้ค่า RMSE สำหรับชุดข้อมูลทดสอบเท่ากับ 0.6828

การศึกษาตัวแบบผสมระหว่างตัวแบบ ARIMA กับตัวแบบโครงข่ายประสาทเทียม (SARIMA+ANN) โดยนำค่าส่วนเหลือที่ได้จากการพยากรณ์ด้วยตัวแบบ ARIMA(1,1,2)x(0,1,1)$_{12}$ มาสร้างตัวแบบโครงข่ายประสาทเทียม โดยใช้เทคนิคการฝึกสอนโครงข่ายแบบ Resilient back propagation(Rprop) พบว่าตัวแบบโครงข่ายประสาทเทียม 1-5-1 เป็นตัวแบบที่เหมาะสมที่สุด กล่าวคือเป็นตัวแบบโครงข่ายประสาทเทียมที่ชั้นข้อมูลนำเข้ามีจำนวนโหนด 1 โหนด, ชั้นซ่อนมีจำนวนโหนด 5 โหนด และชั้นผลลัพธ์มีจำนวนโหนด 1 โหนด เนื่องจากเป็นตัวแบบที่ให้ค่า RMSE ต่ำที่สุด จากนั้นทำการคำนวณหาค่าพยากรณ์รวม(Total forecasting) ซึ่งเป็นการรวมข้อมูลส่วนที่เป็นฟังก์ชันเชิงเส้นตรงที่ได้จากการพยากรณ์ราคาขายปลีกมะนาวด้วยตัวแบบ ARIMA(1,1,2)x(0,1,1)$_{12}$ และข้อมูลส่วนที่ไม่เป็นฟังก์ชันเชิงเส้นตรงที่ได้จากการพยากรณ์ด้วยตัวแบบโครงข่ายประสาทเทียม 1-5-1 เข้าด้วยกัน ซึ่งให้ค่า RMSE สำหรับชุดข้อมูลทดสอบเท่ากับ 0.6242

การศึกษาตัวแบบผสมระหว่างตัวแบบ ARIMA กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน (SARIMA+SVM) โดยนำค่าส่วนเหลือที่ได้จากการพยากรณ์ด้วยตัวแบบ ARIMA(1,1,2)x(0,1,1)$_{12}$ มาสร้างตัวแบบซัพพอร์ทเวกเตอร์แมชชีน โดยได้ชุดของค่าพารามิเตอร์ที่ดีที่สุดจากการทำการปรับจูนค่า คือ cost(C)=0.03125, epsilon(ε)=1 และค่าส่วนเบี่ยงเบนมาตรฐาน: sigma(σ) = 2.7572 จากนั้นทำการคำนวณหาค่าพยากรณ์รวม(Total forecasting) ซึ่งเป็นการรวมข้อมูลส่วนที่เป็นฟังก์ชันเชิงเส้นตรงที่ได้จากการพยากรณ์ราคาขายปลีกมะนาวด้วยตัวแบบ ARIMA(1,1,2)x(0,1,1)$_{12}$ และข้อมูลส่วนที่ไม่เป็นฟังก์ชันเชิงเส้นตรงที่ได้จากการพยากรณ์ด้วยตัวแบบซัพพอร์ทเวกเตอร์แมชชีนเข้าด้วยกัน ซึ่งให้ค่า RMSE สำหรับชุดข้อมูลทดสอบเท่ากับ 0.6639

กล่าวคือการพยากรณ์ราคาขายปลีกมะนาวสำหรับชุดข้อมูลจริงในชุดข้อมูลทดสอบเมื่อใช้รากของค่าคลาดเคลื่อนกำลังสองเฉลี่ย(Root Mean Square Error: RMSE) เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ ตัวแบบผสมระหว่าง ARIMA(1,1,2)x(0,1,1)$_{12}$ กับตัวแบบโครงข่ายประสาทเทียมให้ค่าพยากรณ์ที่แม่นยำที่สุด รองลงมาคือตัวแบบผสมระหว่าง

ARIMA(1,1,2)x$(0,1,1)_{12}$ กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน และตัวแบบ ARIMA(1,1,2)x$(0,1,1)_{12}$ มีความแม่นยำในการพยากรณ์ต่ำที่สุด **ซึ่งให้ผลสอดคล้องกับผลการพยากรณ์ด้วยชุดข้อมูลจำลอง**

บทที่ 5
สรุปผลวิจัยและข้อเสนอแนะ

การศึกษาเปรียบเทียบความแม่นยำของค่าพยากรณ์ที่ได้จาก 3 ตัวแบบ คือ ตัวแบบ ARIMA ที่มีฤดูกาล(SARIMA), ตัวแบบผสมระหว่างตัวแบบ ARIMA ที่มีฤดูกาลกับตัวแบบโครงข่ายประสาทเทียม(SARIMA+ANN) และตัวแบบผสมระหว่างตัวแบบ ARIMA ที่มีฤดูกาลกับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน(SARIMA+SVM) โดยทำการศึกษาเปรียบเทียบทั้งในส่วนของข้อมูลจริงและข้อมูลจำลอง ในส่วนของข้อมูลจริงนั้นได้มีการนำราคาขายปลีกมะนาวเบอร์ 1-2 (หน่วยเป็นบาท/ผล) จากกรมการค้าภายใน กระทรวงพาณิชย์ ซึ่งเป็นราคาผลผลิตทางการเกษตรซึ่งอยู่ในรูปแบบอนุกรมเวลาที่มีปัจจัยเชิงฤดูกาลมาทำการเปรียบเทียบ จากนั้นทำการเปรียบเทียบตัวแบบที่ให้ค่าพยากรณ์ที่แม่นยำที่สุดโดยพิจารณาจากรากของค่าคลาดเคลื่อนกำลังสองเฉลี่ย(Root Mean Square Error: RMSE) โดยตัวแบบที่มีความแม่นยำในการพยากรณ์มากที่สุด คือ ตัวแบบที่มีค่า RMSE ต่ำที่สุด ซึ่งสรุปผลการวิจัยได้ดังนี้

5.1 สรุปผลการวิจัย

ผลการวิจัยจะแบ่งออกเป็น 2 ส่วนคือ

ส่วนที่ 1 ผลการเปรียบเทียบความแม่นยำของค่าพยากรณ์ที่ได้จากตัวแบบ SARIMA, ตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบโครงข่ายประสาทเทียม(SARIMA+ANN) และตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน(SARIMA+SVM) โดยใช้ชุดข้อมูลจำลอง

ส่วนที่ 2 ผลการเปรียบเทียบความแม่นยำของค่าพยากรณ์ที่ได้จากตัวแบบ SARIMA, ตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบโครงข่ายประสาทเทียม(SARIMA+ANN) และตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน(SARIMA+SVM) โดยใช้ชุดข้อมูลจริง

5.1.1 ผลการเปรียบเทียบความแม่นยำของค่าพยากรณ์ที่ได้จากตัวแบบ SARIMA, ตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบโครงข่ายประสาทเทียม(SARIMA+ANN) และตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน(SARIMA+SVM) โดยใช้ชุดข้อมูลจำลอง

ตารางที่ 5.1 ตารางแสดงผลการเปรียบเทียบความแม่นยำของค่าพยากรณ์ที่ได้จากตัวแบบ SARIMA, SARIM-ANN และ SARIMA-SVM

แบบจำลอง	อันดับความแม่นยำของตัวแบบ
1. ARIMA(1,1,1)x$(0,1,1)_{12}$ (มีทั้งสิ้น 8 กรณีย่อย)	อันดับที่1 SARIMA+ANN อันดับที่2 SARIMA+SVM อันดับที่3 SARIMA ทุกกรณีของค่าพารามิเตอร์
2. ARIMA(1,1,2)x$(0,1,1)_{12}$ (มีทั้งสิ้น 8 กรณีย่อย)	อันดับที่1 SARIMA+ANN อันดับที่2 SARIMA+SVM อันดับที่3 SARIMA ทุกกรณีของค่าพารามิเตอร์
3. ARIMA(2,1,1)x$(0,1,1)_{12}$ (มีทั้งสิ้น 8 กรณีย่อย)	อันดับที่1 SARIMA+ANN อันดับที่2 SARIMA+SVM อันดับที่3 SARIMA ยกเว้น 2 กรณีของค่าพารามิเตอร์ คือ กรณีพารามิเตอร์ Θ= -0.5, ϕ_1= -0.5, ϕ_2= -0.5, θ= 0.5 และกรณีพารามิเตอร์ Θ= 0.5, ϕ_1= -0.5, ϕ_2= -0.5, θ= -0.5 ที่ตัวแบบผสมระหว่าง SARIMA+SVM มีความแม่นยำมากที่สุด รองลงมาคือตัวแบบผสมระหว่าง SARIMA+ANN และตัวแบบ SARIMA ตามลำดับ
4. ARIMA(2,1,2)x$(0,1,1)_{12}$ (มีทั้งสิ้น 8 กรณีย่อย)	อันดับที่1 SARIMA+ANN อันดับที่2 SARIMA+SVM อันดับที่3 SARIMA ยกเว้น 1 กรณีของค่าพารามิเตอร์ คือ กรณีพารามิเตอร์ Θ= -0.5, ϕ_1= -0.5, ϕ_2= -0.5, θ_1= -0.5, θ_2 = -0.5 ที่ตัวแบบผสมระหว่าง SARIMA+SVM มีความแม่นยำมากที่สุด รองลงมาคือตัวแบบผสมระหว่าง SARIMA+ANN และ

แบบจำลอง	อันดับความแม่นยำของตัวแบบ
	ตัวแบบ SARIMA ตามลำดับ
5. ARIMA(1,1,1)x(1,1,0)$_{12}$ (มีทั้งสิ้น 8 กรณีย่อย)	อันดับที่1 SARIMA+ANN อันดับที่2 SARIMA+SVM อันดับที่3 SARIMA ยกเว้น 1 กรณีของค่าพารามิเตอร์ คือ กรณีพารามิเตอร์ Φ = -0.5, ϕ = -0.5, θ =0.5 ที่ตัวแบบผสมระหว่าง SARIMA+SVM มีความแม่นยำมากที่สุด รองลงมาคือตัวแบบผสมระหว่าง SARIMA+ANN และตัวแบบ SARIMA ตามลำดับ
6. ARIMA(1,1,2)x(1,1,0)$_{12}$ (มีทั้งสิ้น 8 กรณีย่อย)	อันดับที่1 SARIMA+ANN อันดับที่2 SARIMA+SVM อันดับที่3 SARIMA ทุกกรณีของค่าพารามิเตอร์
7. ARIMA(2,1,1)x(1,1,0)$_{12}$ (มีทั้งสิ้น 8 กรณีย่อย)	อันดับที่1 SARIMA+ANN อันดับที่2 SARIMA+SVM อันดับที่3 SARIMA ทุกกรณีของค่าพารามิเตอร์
8. ARIMA(2,1,2)x(1,1,0)$_{12}$ (มีทั้งสิ้น 8 กรณีย่อย)	อันดับที่1 SARIMA+ANN อันดับที่2 SARIMA+SVM อันดับที่3 SARIMA ยกเว้น 1 กรณีของค่าพารามิเตอร์ คือ กรณีพารามิเตอร์ Φ = 0.5, ϕ_1 = -0.5, ϕ_2 = -0.5 , θ_1= 0.5 และ θ_2= -0.5 ที่ตัวแบบผสมระหว่าง SARIMA+SVM มีความแม่นยำมากที่สุด รองลงมาคือตัวแบบผสมระหว่าง SARIMA+ANN และตัวแบบ SARIMA ตามลำดับ
9. ARIMA(1,1,1)x(1,1,1)$_{12}$ (มีทั้งสิ้น 16 กรณีย่อย)	อันดับที่1 SARIMA+ANN อันดับที่2 SARIMA+SVM อันดับที่3 SARIMA ทุกกรณีของค่าพารามิเตอร์

แบบจำลอง	อันดับความแม่นยำของตัวแบบ
10. ARIMA(1,1,2)x(1,1,1)$_{12}$ (มีทั้งสิ้น 16 กรณีย่อย)	อันดับที่1 SARIMA+ANN อันดับที่2 SARIMA+SVM อันดับที่3 SARIMA ทุกกรณีของค่าพารามิเตอร์
11. ARIMA(2,1,1)x(1,1,1)$_{12}$ (มีทั้งสิ้น 16 กรณีย่อย)	อันดับที่1 SARIMA+ANN อันดับที่2 SARIMA+SVM อันดับที่3 SARIMA ทุกกรณีของค่าพารามิเตอร์
12. ARIMA(2,1,2)x(1,1,1)$_{12}$ (มีทั้งสิ้น 16 กรณีย่อย)	อันดับที่1 SARIMA+ANN อันดับที่2 SARIMA+SVM อันดับที่3 SARIMA ทุกกรณีของค่าพารามิเตอร์

5.1.2 ผลการเปรียบเทียบความแม่นยำของค่าพยากรณ์ที่ได้จากตัวแบบ SARIMA, ตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบโครงข่ายประสาทเทียม(SARIMA+ANN) และตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน(SARIMA+SVM) โดยใช้ชุดข้อมูลจริง

จากการศึกษาตัวแบบ ARIMA ที่มีฤดูกาล(SARIMA) ภายใต้ค่า P=0-1, Q=0-1, p=1-2 และ q=1-2 พบว่าตัวแบบที่เหมาะสมที่สุดสำหรับใช้พยากรณ์ราคาขายปลีกมะนาว คือ ตัวแบบ ARIMA(1,1,2)x(0,1,1)$_{12}$ เนื่องจากมีค่า AIC ที่ต่ำที่สุด ซึ่งมีรูปแบบของสมการพยากรณ์คือ $X_t = -0.0003 + 1.3211X_{t-1} - 0.3211X_{t-2} + X_{t-12} - 1.3211X_{t-13} + 0.3211X_{t-14} + W_t$ $-0.3892W_{t-1} - 0.4646W_{t-2} - 0.7365W_{t-12} + 0.2866W_{t-13} + 0.3421W_{t-14}$ โดยให้ค่าของ RMSE สำหรับชุดข้อมูลทดสอบเท่ากับ 0.6828

จากการศึกษาตัวแบบผสมระหว่างตัวแบบ ARIMA กับตัวแบบโครงข่ายประสาทเทียม (SARIMA+ANN) โดยนำค่าส่วนเหลือที่ได้จากการพยากรณ์ด้วยตัวแบบ ARIMA(1,1,2)x(0,1,1)$_{12}$ มาสร้างตัวแบบโครงข่ายประสาทเทียม โดยใช้เทคนิคการฝึกสอนโครงข่ายแบบ Resilient back propagation(Rprop) พบว่าตัวแบบโครงข่ายประสาทเทียม 1-5-1 เป็นตัวแบบที่เหมาะสมที่สุด กล่าวคือเป็นตัวแบบโครงข่ายประสาทเทียมที่ชั้นข้อมูลนำเข้ามีจำนวนโหนด 1 โหนด, ชั้นซ่อนมีจำนวนโหนด 5 โหนด และชั้นผลลัพธ์มีจำนวนโหนด 1 โหนด เนื่องจากเป็นตัวแบบที่ให้ค่า RMSE

ต่ำที่สุด จากนั้นทำการคำนวณหาค่าพยากรณ์รวม(Total forecasting) ซึ่งเป็นการรวมข้อมูลส่วนที่เป็นฟังก์ชันเชิงเส้นตรงที่ได้จากการพยากรณ์ราคาขายปลีกมะนาวด้วยตัวแบบ ARIMA(1,1,2)x$(0,1,1)_{12}$ และข้อมูลส่วนที่ไม่เป็นฟังก์ชันเชิงเส้นตรงที่ได้จากการพยากรณ์ด้วยตัวแบบโครงข่ายประสาทเทียม 1-5-1 เข้าด้วยกัน ซึ่งให้ค่า RMSE สำหรับชุดข้อมูลทดสอบเท่ากับ 0.6242

จากการศึกษาตัวแบบผสมระหว่างตัวแบบ ARIMA กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน (SARIMA+SVM) โดยนำค่าส่วนเหลือที่ได้จากการพยากรณ์ด้วยตัวแบบ ARIMA(1,1,2)x$(0,1,1)_{12}$ มาสร้างตัวแบบซัพพอร์ทเวกเตอร์แมชชีน โดยได้ชุดของค่าพารามิเตอร์ที่ดีที่สุดจากการทำการปรับจูนค่า คือ cost(C)=0.03125, epsilon(ε)=1 และค่าส่วนเบี่ยงเบนมาตรฐาน: sigma(σ) = 2.7572 จากนั้นทำการคำนวณหาค่าพยากรณ์รวม(Total forecasting) ซึ่งเป็นการรวมข้อมูลส่วนที่เป็นฟังก์ชันเชิงเส้นตรงที่ได้จากการพยากรณ์ราคาขายปลีกมะนาวด้วยตัวแบบ ARIMA(1,1,2)x$(0,1,1)_{12}$ และข้อมูลส่วนที่ไม่เป็นฟังก์ชันเชิงเส้นตรงที่ได้จากการพยากรณ์ด้วยตัวแบบซัพพอร์ทเวกเตอร์แมชชีนเข้าด้วยกัน ซึ่งให้ค่า RMSE สำหรับชุดข้อมูลทดสอบเท่ากับ 0.6639

กล่าวคือการพยากรณ์ราคาขายปลีกมะนาวสำหรับชุดข้อมูลจริงในชุดข้อมูลทดสอบเมื่อใช้รากของค่าคลาดเคลื่อนกำลังสองเฉลี่ย(Root Mean Square Error: RMSE) เป็นเกณฑ์ในการเปรียบเทียบความแม่นยำของตัวแบบ ตัวแบบผสมระหว่าง ARIMA(1,1,2)x$(0,1,1)_{12}$ กับตัวแบบโครงข่ายประสาทเทียมให้ค่าพยากรณ์ที่แม่นยำที่สุด รองลงมาคือตัวแบบผสมระหว่าง ARIMA(1,1,2)x$(0,1,1)_{12}$ กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน และตัวแบบ ARIMA(1,1,2)x$(0,1,1)_{12}$ มีความแม่นยำในการพยากรณ์ต่ำที่สุด **ซึ่งให้ผลสอดคล้องกับผลการพยากรณ์ด้วยชุดข้อมูลจำลอง**

สรุปได้ว่า ภายใต้ขอบเขตการวิจัยในงานวิจัยนี้ตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบโครงข่ายประสาทเทียม(SARIMA+ANN) และตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน(SARIMA+SVM) ให้ผลการพยากรณ์ที่แม่นยำกว่าตัวแบบ SARIMA ทั้งในชุดข้อมูลจริง และชุดข้อมูลจำลอง ดังนั้นการพิจารณาเลือกใช้ตัวแบบแบบผสมสำหรับการพยากรณ์ข้อมูลอนุกรมเวลาที่มีปัจจัยเชิงฤดูกาลก็ถือเป็นอีกทางเลือกหนึ่งที่น่าสนใจ เนื่องจากเป็นการรวมเอาลักษณะเด่นของตัวแบบเดี่ยวเข้าไว้ด้วยกัน ทำให้สามารถจับกับข้อมูลทั้งในส่วนที่เป็นเชิงเส้นตรงและไม่เป็นเชิงเส้นตรงได้มีประสิทธิภาพมากยิ่งขึ้น ช่วยลดความผิดพลาดในการพยากรณ์ค่าของข้อมูลเมื่อเทียบกับการใช้ตัวแบบเดี่ยวเพียงตัวแบบเดียว แต่ก็มีข้อจำกัดคือเรื่องของระยะเวลาที่ใช้ในการประมวลผลที่ค่อนข้างนานกว่ามากเมื่อเทียบกับระยะเวลาในการประมวลผลของตัวแบบเดี่ยว ทั้งนี้ขึ้นอยู่กับวัตถุประสงค์ของงานนั้นๆ ความซับซ้อนของข้อมูล และระยะเวลาที่มีของผู้ที่นำไปใช้งาน

5.2 ข้อเสนอแนะ

1. ในงานวิจัยนี้ทำการเปรียบเทียบความแม่นยำของค่าพยากรณ์เพียง 3 ตัวแบบ คือ ตัวแบบ ARIMA ที่มีฤดูกาล(SARIMA), ตัวแบบผสมระหว่างตัวแบบตัวแบบ ARIMA ที่มีฤดูกาลกับตัวแบบโครงข่ายประสาทเทียม(SARIMA+ANN) และตัวแบบผสมระหว่างตัวแบบ ARIMA ที่มีฤดูกาลกับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน(SARIMA+SVM) ผู้วิจัยอื่นอาจทดลองทำการเปรียบเทียบตัวแบบผสมอื่นๆที่น่าสนใจ เช่น ตัวแบบผสม SARIMA+ANN+SVM ,ตัวแบบผสม Pegels-ARIMA ,ตัวแบบผสม ARIMA กับ Adaptive neuro-fuzzy inference system(ANFIS) หรือตัวแบบผสมระหว่างตัวแบบ ARIMA และการถดถอยโพลิโนเมียล เป็นต้น

2. การสร้างตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบโครงข่ายประสาทเทียม ในงานวิจัยนี้เลือกใช้เทคนิคการฝึกสอนโครงข่ายแบบ Resilient back propagation(Rprop) ซึ่งผู้วิจัยอื่นๆที่สนใจศึกษาอาจะเลือกใช้เทคนิคการฝึกสอนโครงข่ายแบบอื่น เช่น Levenberg-Marquardt, BFGS Quasi-Newton หรือ Gradient Descent with Momentum เป็นต้น

3. การสร้างตัวแบบผสมระหว่างตัวแบบ SARIMA กับตัวแบบซัพพอร์ทเวกเตอร์แมชชีน ในงานวิจัยนี้เลือกใช้ kernel function คือ Gaussian radial basis function(RBF) ซึ่งผู้วิจัยอื่นๆอาจเลือกใช้ kernel function ประเภทอื่นได้ เช่น Polynomial kernel หรือ Sigmoid kernel เป็นต้น ทั้งนี้ขึ้นอยู่กับลักษณะของข้อมูลด้วย หรืออาจทดลองเพิ่มจำนวนโหนดที่ใช้ในชั้นซ่อนเพื่อหาจำนวนโหนดที่เหมาะสมมากขึ้น นอกจากนี้การทดลองปรับจูนหาชุดของค่าพารามิเตอร์ของตัวแบบซัพพอร์ทเวกเตอร์แมชชีน คือ C และ ε ชุดอื่นๆที่ต่างออกไป ก็อาจจะทำให้ผลลัพธ์ในการพยากรณ์ค่ามีความแม่นยำมากยิ่งขึ้น

4. การจำลองชุดข้อมูลราคาขายปลีกมะนาวในงานวิจัยนี้ เลือกใช้ค่าพารามิเตอร์ในการสร้างตัวแบบเพียง 2 ค่า คือ -0.5 และ 0.5 ผู้วิจัยอื่นสามารถเปลี่ยนค่าพารามิเตอร์ในการสร้างตัวแบบเป็นค่าอื่นๆ ซึ่งอาจทำให้เกิดผลลัพธ์ในการพยากรณ์ที่ต่างออกไป

รายการอ้างอิง

Chen, K.-Y. and C.-H. Wang (2007). "A hybrid SARIMA and support vector machines in forecasting the production values of the machinery industry in Taiwan." Expert Systems with Applications **32**: 254-264.

Hsu, C.-W., C.-C. Chang and C.-J. Lin (2016). A Practical Guide to Support Vector Classification, National Taiwan University.

Karlik, B. and A. V. Olgac "Performance Analysis of Various Activation Functions in Generalized MLP Architectures of Neural Networks." International Journal of Artificial Intelligence And Expert Systems(IJAE) **1**(4).

Riedmiller, M. and H. Braun (1993). "A Direct Adaptive Method for Faster Backpropagation Learning :The RPROP Algorithm." Proceedings of the IEEE International Conference on Neural Networks(ICNN): 586-591

Shumway., R. H. and D. S. Stoffer. (2010). Time series analysis and its applications with R examples. USA, Springer.

Thissena, U., R. v. Brakela, A. P. d. Weijerb, W. J. Melssena and L. M. C. Buydensa (2003). "Using support vector machines for time series prediction." Chemometrics and Intelligent Laboratory Systems **69**: 35-49.

Wang, L. L. B. S. X. (2014). "Research on Kernel Function of Support Vector Machine." Journal of Computers **25**(1): 13-14

Zhang, G., B. E. Patuwo and M. Y. Hu. (1988). "Forecasting with artificial neural networks." International Journal of Forecasting **14**: 35-62.

ZHANG, G. P. (2003). "Time series forecasting using a hybrid ARIMA and neural network model." Neurocomputing **50**: 159-175.

ชญานิน บุญมานะ และ นัท กุลวานิช (2560). "การเปรียบเทียบความแม่นยำของการพยากรณ์ด้วยตัวแบบอนุกรมเวลาแบบผสม." วารสารวิทยาศาสตร์และเทคโนโลยี: 177-190.

ทรงศิริ แต้สมบัติ (2553). การพยากรณ์เชิงปริมาณ. กรุงเทพมหานคร, สำนักพิมพ์มหาวิทยาลัยเกษตรศาสตร์.

ภัทร วรภู (2556). การเปรียบเทียบความแม่นยำของการพยากรณ์อนุกรมเวลาระหว่างตัวแบบผสมและตัวแบบเดี่ยว. กรุงเทพมหานคร, จุฬาลงกรณ์มหาวิทยาลัย.

ภูมิฐาน รังคกูลนุวัฒน์ (2556). การวิเคราะห์อนุกรมเวลาสำหรับเศรษฐศาสตร์และธุรกิจ. กรุงเทพมหานคร. กรุงเทพมหานคร, สำนักพิมพ์แห่งจุฬาลงกรณ์มหาวิทยาลัย.

มุกดา แม้นมินทร์ (2549). อนุกรมเวลาและการพยากรณ์. กรุงเทพมหานคร, สำนักพิมพ์ประกายพรึก.

ศิริลักษณ์ สุวรรณวงศ์ (2535). การวิเคราะห์อนุกรมเวลา. กรุงเทพมหานคร, สำนักพิมพ์สุวีริยาสาสน์.

ภาคผนวก

<u>คำสั่งการวิเคราะห์ข้อมูลด้วยโปรแกรม R เวอร์ชัน 3.4.3</u>

```
### Library used in analysis ##
library(TSA)
library(DMwR)
library(forecast)
library(neuralnet)
library(Metrics)
library(kernlab)
library(CombMSC)
library(e1071)

##########################################################
########################## REAL DATA #####################
##########################################################
limeprice<-read.table("limeprice.txt")
limeprice<-ts(limeprice,start=2540,deltat = 1/12)
plot(limeprice,main="Limeprice plot",ylab="limeprice",xlab="Year",type="l")
acf(as.vector(limeprice),main="Sample ACF for limeprice",lag.max = 240)
pacf(as.vector(limeprice),main="Sample PACF for limeprice",lag.max = 240)

### seasonal diff
ssdiff<-diff(limeprice,12)
length(ssdiff)
plot(ssdiff,main="Seasonal diff of limeprice",ylab="ss diff",xlab="Year",type="l",deltat =
1/12)
acf(as.vector(ssdiff),main="Sample ACF for Seasonal diff",lag.max = 240)
pacf(as.vector(ssdiff),main="Sample PACF for Seasonal diff",lag.max = 240)
```

```
### trend diff
tdiff<-diff(ssdiff,1)
plot(tdiff,main="Seasonal diff + Trend diff of
limeprice",ylab="tdiff",xlab="Year",type="l",deltat = 1/12)
acf(as.vector(tdiff),main="Sample ACF for Seasonal diff + Trend diff",lag.max = 240)
pacf(as.vector(tdiff),main="Sample PACF for Seasonal diff + Trend diff",lag.max = 240)

### Data set splitting (training 70% and testing 30%) ###
diff_lime<-tdiff
diff_lime_train<-diff_lime[1:ceiling(length(diff_lime)*0.7)]
diff_lime_test<-diff_lime[(ceiling(length(diff_lime)*0.7)+1):length(diff_lime)]
plot(as.vector(diff_lime),type='l',xlab="Time",ylab="diff_lime",main="Training data VS
Testing data")
abline(v=159,col="red",lty=5,lwd=2)

############################################################
#################### 1) SARIMA models  ######################
############################################################

##########################
##### 1.1 TRAINING DATA #####
##########################
## Fit SARIMA(P=0-1,Q=0-1,p=1-2,q=1-2,d=1,D=1) models
P<-1
Q<-1
p<-2
q<-2
aic<-c()
I<-c()
J<-c()
K<-c()
```

```
L<-c()
for(i in 0:P){
  for(j in 0:Q){
    for(k in 1:p){
      for(l in 1:q){
        fit_train<-
Arima(diff_lime_train,seasonal=list(order=c(i,0,j),period=12),order=c(k,0,l),method='ML'
)
        aic<-c(aic,fit_train$aic)
        I<-c(I,i)
        J<-c(J,j)
        K<-c(K,k)
        L<-c(L,l)
      }
    }
  }
}
I<-data.frame(P=I,Q=J,p=K,q=L,AIC=aic)
all_arima_model<-I[5:16,]
```



```
### Choosing the best ARIMA(p,d,q) models using "AIC"
num.mod<-dim(all_arima_model)[1]
for(i in 1:num.mod){
  min_aic<-min(all_arima_model[,5])
  if(min_aic==min(all_arima_model[i,5]))
  {
    P_best<-all_arima_model[i,1]
    Q_best<-all_arima_model[i,2]
    p_best<-all_arima_model[i,3]
    q_best<-all_arima_model[i,4]
  }
```

```
}
P_best
Q_best
p_best
q_best

### Fit the best arima model
fit_train<-
arima(diff_lime_train,seasonal=list(order=c(P_best,0,Q_best),period=12),order=c(p_bes
t,0,q_best),method='ML')

### Compute residual&fitted values
res_arima_train<-resid(fit_train)
fitted_arima_train<-diff_lime_train-res_arima_train
fitted_arima_train
### Compare real training data VS fitted values from the best ARIMA models in
training data set
plot(diff_lime_train,col="blue",type="l",ylab= "diff(lime_train)",main="Real training data
VS Fitted values")
lines(fitted_arima_train,col="red")



##########################
#### 1.2 TESTING DATA ######
##########################
fit_test<-
arima(diff_lime_test,seasonal=list(order=c(P_best,0,Q_best),period=12),order=c(p_best
,0,q_best),method='ML')

### Compute residual&fitted values
res_arima_test<-resid(fit_test)
fitted_arima_test<-diff_lime_test-res_arima_test
```

```
fitted_arima_test

### Compare real testing data VS fitted values from the best ARIMA models in testing data set
par(mfrow=c(1,1))
plot(diff_lime_test,col=" blue",type="l",ylab="diff(lime_test)",main="Real testing data VS Fitted values")
lines(fitted_arima_test,col="red")

### Compute RMSE from the best ARIMA models in testing data set
crite.arima.test<-
regr.eval(diff_lime_test[2:68],fitted_arima_test[2:68],stat=c("mae","mse","rmse"))
crite.arima.test



############################################################
##################### 2) Hybrid ARIMA+ANN models ##############
############################################################

#########################
#### 2.1 TRAINING DATA #####
#########################

#### Using residuals in training data set from the best ARIMA model to bulid ANN model
#### Transformation by Min-Max Normalization
norm_res_train <-(res_arima_train-min(res_arima_train))/(max(res_arima_train)-min(res_arima_train))
norm_res_train_dat <-data.frame(norm_res_train)

### Find the best hidden neuron + best weights ANN model
no.iteration<-1000
```

```
input_neuron<-1
output_neuron<-1
hidden_neuron<-5
J<-c()
I<-c()
Weight<-c()
RMSE<-c()
for(i in 1:no.iteration){
  for(j in 1:hidden_neuron){
    ann_train<-neuralnet(norm_res_train[2:159]~norm_res_train[1:158],data=
    norm_res_train_dat,hidden=j, algorithm = "rprop+",learningrate
    =0.1,learningrate.factor=list(minus=0.5,plus=1.2),act.fct="logistic",
    linear.output=TRUE)
    weight<-ann_train$startweights
    Weight<-c(Weight,weight)
    predict_ann_train<-ann_train$net.result[[1]]
    rmse <-rmse(norm_res_train[2:159],predict_ann_train)
    J<-c(J,j)
    I<-c(I,i)
    RMSE<-c(RMSE,rmse)
  }
}
data.frame(hidden_neuron=J,iteration=I,RMSE=RMSE)
all_ann_train<-data.frame(hidden_neuron=J,iteration=I,RMSE=RMSE)

### Choose the best hidden neuron + best weights ANN model using "RMSE"
b<-which.min(all_ann_train$RMSE)
best_ann<-all_ann_train[b,]
best_weight<-Weight[b]
best_hidden_neuron<-all_ann_train[b,1]
```

```
##############################
######## 2.2 TESTING DATA ######
##############################

### Using residuals in testing data set to checking validation from Hybrid
ARIMA+ANN model
### Min-Max Normalization the residuals from the best ARIMA models in
testing data set
norm_res_test<-(res_arima_test-min(res_arima_test))/(max(res_arima_test)-
min(res_arima_test))
norm_res_test_dat<-data.frame(norm_res_test)

### Forecast from the best ANN model
best_ann_test<-
neuralnet(norm_res_test[2:68]~norm_res_test[1:67],data=norm_res_test_dat,startweig
hts = best_weight,hidden=best_hidden_neuron,algorithm = "rprop+",learningrate =
0.1,learningrate.factor=list(minus=0.5,plus=1.2),act.fct="logistic",linear.output=TRUE)
norm_res_test_forec<-best_ann_test$net.result[[1]]

### Tranform forecasting value to the original scale (Denormalization)
res_test_forec<-(norm_res_test_forec*(max(res_arima_test)-
min(res_arima_test)))+min(res_arima_test)

### Calculate "Total forecasting"(ARIMA+ANN) in testing data set
real_test_data    <-diff_lime_test
forec_arima_test  <-fitted_arima_test
forec_ann_test    <-c(rep(0,1),res_test_forec)
total_forec_arima_ann_test<-forec_arima_test+forec_ann_test
total_arima_ann_test<-
cbind(real_test_data,forec_arima_test,forec_ann_test,total_forec_arima_ann_test)
```

Compare real testing data VS fitted values from the best Hybrid ARIMA+ANN models in testing data

```
plot(diff_lime_test,col=" blue",type="l",ylab="",main="Real testing data VS Fitted values(ARIMA vs ARIMA+ANN)")
lines(fitted_arima_test,col="red")
lines(total_forec_arima_ann_test,col="green")
```

Compute MSE,RMSE and MAE from Hybrid ARIMA+ANN models in testing data

```
crite.arima.ann.test<-regr.eval(real_test_data[2:68],total_forec_arima_ann_test[2:68],stat=c("mae","mse","rmse"))
crite.arima.ann.test
```

3) Hybrid ARIMA+SVM models ###############

########################
3.1 TRAINING DATA
########################

Using residuals in training data set from ARIMA model to buliding the SVM model

```
svm_train_dat<-data.frame(res_arima_train)
```

tune parameter C and epsilon in SVM

```
X<-data.frame(X=res_arima_train[1:158])
Y<-data.frame(Y=res_arima_train[2:159])
svm_train_dat1<-data.frame(X,Y)
set.seed(2)
```

```
obj<- tune.svm(Y~X,data=svm_train_dat1,cost=c(2^-5,2^-3,2^-
1,2,2^3,2^5),epsilon=c(0.001,0.01,0.1,1),kernel='radial')
best_c<-obj$best.parameters[1]
best_epsilon<-obj$best.parameters[2]

no.iterations<-1000
Forec_svm_train<-c()
J<-c()
RMSE<-c()
SIGMA<-c()
for(j in 1:no.iterations){
  ## estimate good sigma values for kernel RBF
  srange.train<-
sigest(res_arima_train[2:159]~res_arima_train[1:158],data=svm_train_dat)
  svm_train<-
ksvm(res_arima_train[2:159]~res_arima_train[1:158],data=svm_train_dat,kernel="rbfdot
",C=best_c, epsilon = best_epsilon,kpar=list(sigma=srange.train[2]))
  rmse<-rmse(res_arima_train[2:159],predict(svm_train))
  mae<-mae(res_arima_train[2:159],predict(svm_train))
  J<-c(J,j)
  RMSE<-c(RMSE,rmse)
  SIGMA<-c(SIGMA,srange.train[2])
}
all_svm_train<-data.frame(sigma.value=SIGMA,iteration=J,RMSE=RMSE)

### Choose the best sigma model using "RMSE"
i<-which.min(all_svm_train[,3])
#best_svm<-all_svm_train[min,]
best_sigma_train<-SIGMA[i]
```

```
############################
####### 3.2 TESTING DATA ######
############################
### Using residuals in testing data set from ARIMA model to buliding the SVM model
svm_test_dat<-data.frame(res_arima_test)

### Forecast from the best SVM model
best_svm_test<-
ksvm(res_arima_test[2:68]~res_arima_test[1:67],data=svm_test_dat,kernel="rbfdot",C=best_c,epsilon=best_epsilon,kpar=list(sigma=best_sigma_train))

### Calculate "Total forecasting"(ARIMA + SVM) in testing data set
real_test_data   <-diff_lime_test
forec_arima_test <-fitted_arima_test
forec_svm_test   <-c(rep(0,1),predict(best_svm_test))
total_forec_arima_svm_test<-forec_arima_test+forec_svm_test
total_arima_svm_test<-
cbind(real_test_data,forec_arima_test,forec_svm_test,total_forec_arima_svm_test)

### Compare real testing data VS fitted values from the best Hybrid ARIMA+SVM models in testing data set
plot(diff_lime_test,col=" black",type="l",ylab="",main="Real testing data VS Fitted values(SARIMA vs SARIMA+ANN vs SARIMA+SVM)")
lines(fitted_arima_test,col="blue",type="l",pch=15,lwd=1)
lines(fitted_arima_test,col="blue",type="b",pch=15,lwd=1,cex=0.8)
lines(total_forec_arima_ann_test,col="red",type="l",pch=17,lwd=1)
lines(total_forec_arima_ann_test,col="red",type="b",pch=17,lwd=1,cex=0.8)
lines(total_forec_arima_svm_test,col="chartreuse3",type="l",pch=16,lwd=1)
lines(total_forec_arima_svm_test,col="chartreuse3",type="b",pch=16,lwd=1)
```

```
legend(x=50,y=2.6,c("SARIMA","SARIMA+ANN","SARIMA+SVM"),col=c("blue","red","chartreuse3"),lty=rep(1,2,3),pch=c(15,17,16),cex=0.5,bty="o",pt.cex=1)
```

Compute MSE, RMSE and MAE from Hybrid ARIMA+SVM models in testing data set

```
crite.arima.svm.test<-
regr.eval(real_test_data[2:68],total_forec_arima_svm_test[2:68],stat=c("mae","mse","rmse"))
crite.arima.svm.test
```

SUMMARY : show RMSE from SARIMA/ANN/SVM for REAL DATA

```
crite.arima.test
crite.arima.ann.test
crite.arima.svm.test
```

##
####################### SIMULATED DATA ####################
##

##input no.of simmulation and iteration

```
no.simmu<-100
no_iteration_ann<-100  #find best hidden neuron+ best weight
no.iteration_svm<-100  #find best sigma
```

##
########### 1) Simulations ARIMA(1,1,2)x(0,1,1)_12 models ########
##

```
Real_train_sim <-c()
Real_test_sim  <-c()
RES_train_sim  <-c()
RES_test_sim   <-c()
```

```
FITTED_arima_train_sim<-c()
FITTED_arima_test_sim<-c()
for(i in 1:no.simmu){
  P<-0
  Q<-1
  p<-1
  q<-2
  sim3<-sarima.Sim(n=20,period=12,seasonal= list(order=c(P,1,Q),ma=c(0.5))

,model=list(order=c(p,1,q),ar=c(0.5),ma=c(0.5,0.5)),rand.Gen.Fun=rnorm,rand.Gen.Seas
=rnorm)
  sim2<-diff(sim3,12)
  sim<-diff(sim2,1)

  ### Split data 70%(training) and 30%(testing)
  real_train_sim <-sim[1:159]
  real_test_sim  <-sim[160:227]
  Real_train_sim <-cbind(Real_train_sim,real_train_sim)
  Real_test_sim  <-cbind(Real_test_sim,real_test_sim)
### Fit SARIMA(P=0-1,Q=0-1,p=1-2,q=1-2,d=1,D=1) models
  P<-1
  Q<-1
  p<-2
  q<-2
  AIC<-c()
  T<-c()
  J<-c()
  K<-c()
  L<-c()
  for(t in 0:P){
    for(j in 0:Q){
```

```
      for(k in 1:p){
        for(l in 1:q){
          fit_arima_train_sim<-

Arima(Real_train_sim[,i],seasonal=list(order=c(t,0,j),period=12),order=c(k,0,l),method='
ML')
          AIC<-c(AIC,fit_arima_train_sim$aic)
          T<-c(T,t)
          J<-c(J,j)
          K<-c(K,k)
          L<-c(L,l)
        }
      }
    }
  }
  I2<-data.frame(P=T,Q=J,p=K,q=L,AIC=AIC)[5:16,]
  best<-which.min(I2$AIC)

  ### Choosing the best ARIMA(p,d,q) models using "AIC"
  P_best<-I2[best,1]
  Q_best<-I2[best,2]
  p_best<-I2[best,3]
  q_best<-I2[best,4]

 ### Fitting SARIMA model
  fit_train_sim <-
arima(Real_train_sim[,i],order=c(p_best,0,q_best),method='ML',seasonal=list(order=c(P
_best,0,Q_best),period=12))
  fit_test_sim  <-
arima(Real_test_sim[,i],order=c(p_best,0,q_best),method='ML',seasonal=list(order=c(P
_best,0,Q_best),period=12))
```

```
  res_train_sim            <-resid(fit_train_sim)
  res_test_sim             <-resid(fit_test_sim)
  fitted_arima_train_sim  <-real_train_sim-res_train_sim
  fitted_arima_test_sim   <-real_test_sim-res_test_sim

  RES_train_sim            <-cbind(RES_train_sim,res_train_sim)
  RES_test_sim             <-cbind(RES_test_sim,res_test_sim)
  FITTED_arima_train_sim  <-cbind(FITTED_arima_train_sim,fitted_arima_train_sim)
  FITTED_arima_test_sim   <-cbind(FITTED_arima_test_sim,fitted_arima_test_sim)
}

### Computing RMSE from ARIMA models
Crite.arima.train.sim<-c()
Crite.arima.test.sim<-c()
for(i in 1:no.simmu){
crite.arima.train.sim <-
regr.eval(Real_train_sim[2:159,i],FITTED_arima_train_sim[2:159,i],stat=c("mae","mse","rm
se"))
crite.arima.test.sim  <-
regr.eval(Real_train_sim[2:68,i],FITTED_arima_test_sim[2:68,i],stat=c("mae","mse","rmse"
))
Crite.arima.train.sim <-rbind(Crite.arima.train.sim,crite.arima.train.sim)
Crite.arima.test.sim  <-rbind(Crite.arima.test.sim,crite.arima.test.sim)
}

### Show RMSE for each simulation
Crite.arima.train.sim
Crite.arima.test.sim
```

```
### Calculate E[RMSE] from all simulations
mean_error_sim_train_arima <-
data.frame(Mean_MAE_train=mean(Crite.arima.train.sim[,1]),Mean_MSE_train=mean(Cri
te.arima.train.sim[,2]),Mean_RMSE_train=mean(Crite.arima.train.sim[,3]))
mean_error_sim_test_arima  <-
data.frame(Mean_MAE_test=mean(Crite.arima.test.sim[,1]),Mean_MSE_test=mean(Crite
.arima.test.sim[,2]),Mean_RMSE_test=mean(Crite.arima.test.sim[,3]))
mean_error_sim_train_arima
mean_error_sim_test_arima



############################################################
############### 2) Simulation Hybrid ARIMA+ANN models ############
############################################################

#########################
#### 2.1 TRAINING DATA #####
#########################

### Transformation by Min-Max Normalization ###
Norm_res_train<-c()
for(i in 1:no.simmu){
  norm_res_train<-(RES_train_sim[,i]-min(RES_train_sim[,i]))/(max(RES_train_sim[,i])-
min(RES_train_sim[,i]))
  Norm_res_train<-cbind(Norm_res_train,norm_res_train)
}

input_neuron<-1
output_neuron<-1
hidden_neuron<-5
Norm_res_train_forec<-c()
```

```
BEST_weight<-c()
BEST_hidden_neuron<-c()
for(i in 1:no.simmu){
  J<-c()
  K<-c()
  RMSE<-c()
  Weight<-c()
  for(j in 1:no_iteration_ann){
    for(k in 1:hidden_neuron){
      ann_train<-
neuralnet(Norm_res_train[2:159,i]~Norm_res_train[1:158,i],data=Norm_res_train[,i],hidd
en=k,algorithm = "rprop+",learningrate =
0.1,learningrate.factor=list(minus=0.5,plus=1.2),act.fct="logistic",linear.output=TRUE)
      weight<-ann_train$startweights
      Weight<-c(Weight,weight)
      predict_ann_train<-ann_train$net.result[[1]]
      rmse <-rmse(Norm_res_train[2:159,i],predict_ann_train)
      J<-c(J,j)
      K<-c(K,k)
      RMSE<-c(RMSE,rmse)
    }
  }
  all_ann_train<-data.frame(hidden_neuron=K,iteration=J,RMSE=RMSE)

  ### Choose the best hidden neuron+ best weights ANN model using "RMSE"
  b<-which.min(all_ann_train$RMSE)
  #best_ann<-all_ann_train[b,]
  best_weight<-Weight[b]
  best_hidden_neuron<-all_ann_train[b,1]
  BEST_hidden_neuron<-c(BEST_hidden_neuron,best_hidden_neuron)
  BEST_weight<-c(BEST_weight,best_weight)
```

```
}

##############################
####### 2.2 TESTING DATA #######
##############################
### Using residuals in testing data set to checking validation from Hybrid
ARIMA+ANN model
### Min-Max Normalization the residuals from the best ARIMA models in
testing data set
### Forecast from the best ANN model each simulation
Norm_res_test<-c()
for(i in 1:no.simmu){
  norm_res_test<-(RES_test_sim[,i]-min(RES_test_sim[,i]))/(max(RES_test_sim[,i])-
min(RES_test_sim[,i]))
  Norm_res_test<-cbind(Norm_res_test,norm_res_test)
}

Norm_res_test_forec<-c()
for(i in 1:no.simmu){
best_ann_test<-
neuralnet(Norm_res_test[2:68,i]~Norm_res_test[1:67,i],data=Norm_res_test[,i],startweig
hts = BEST_weight[i],hidden=BEST_hidden_neuron[i],algorithm =
"rprop+",learningrate.factor=list(minus=0.5,plus=1.2),act.fct="logistic",linear.output=TR
UE)
norm_res_test_forec<-best_ann_test$net.result[[1]]
Norm_res_test_forec<-cbind(Norm_res_test_forec,norm_res_test_forec)
}
### Tranform forecasting value to the original scale (Denormalization)
Res_test_forec_sim_ann<-c()
for(i in 1:no.simmu){
```

```
  res_test_forec_sim_ann<-(Norm_res_test_forec[,i]*(max(RES_test_sim[,i])-
min(RES_test_sim[,i])))+min(RES_test_sim[,i])
  Res_test_forec_sim_ann<-cbind(Res_test_forec_sim_ann,res_test_forec_sim_ann)
}
```

Calculate "Total forecasting"(ARIMA + ANN) in training data set

```
Total_forec_arima_ann_test<-c()
for(i in 1:no.simmu){
  #real_test_sim     <-Real_test_sim[,i]
  forec_arima_test <-FITTED_arima_test_sim[,i]
  forec_ann_test    <-c(rep(0,1),Res_test_forec_sim_ann[,i])

  total_forec_arima_ann_test<-forec_arima_test+forec_ann_test
  Total_forec_arima_ann_test<-
cbind(Total_forec_arima_ann_test,total_forec_arima_ann_test)
}
```

Computing RMSE from Hybrid ARIMA+ANN models in testing data set

```
Crite.arima.ann.test_sim<-c()
for(i in 1:no.simmu){
  crite.arima.ann.test_sim<-
regr.eval(Real_test_sim[2:68,i],Total_forec_arima_ann_test[2:68,i],stat=c("mae","mse","r
mse"))
  Crite.arima.ann.test_sim<-rbind(Crite.arima.ann.test_sim,crite.arima.ann.test_sim)
}
```


Calculate E[RMSE] from all simulations in testing data set

```
mean_error_sim_test_arima_ann <-
data.frame(Mean_MAE_test_ann=mean(Crite.arima.ann.test_sim[,1]),Mean_MSE_test_a
nn=mean(Crite.arima.ann.test_sim[,2]),Mean_RMSE_test_ann=mean(Crite.arima.ann.tes
t_sim[,3]))
```

```
mean_error_sim_test_arima_ann

##############################################################
############ 3) Simulation Hybrid ARIMA+SVM models ###############
##############################################################

##############################
####### 3.1 TRAINING DATA #######
##############################
### Using residuals in training data set from ARIMA model to bulid the SVM
model

### tune parameter C and epsilon in SVM
BEST_c<-c()
BEST_epsilon<-c()
BEST_sigma_train<-c()
for(i in 1:no.simmu){
  X<-data.frame(X=RES_train_sim[1:158,i])
  Y<-data.frame(Y=RES_train_sim[2:159,i])
  svm_train_dat1<-data.frame(X,Y)
  set.seed(2)
  obj<- tune.svm(Y~X,data=svm_train_dat1,cost=c(2^-5,2^-3,2^-
1,2,2^3,2^5),epsilon=c(0.001,0.01,0.1,1),kernel='radial')
  best_c<-obj$best.parameters[1]
  best_epsilon<-obj$best.parameters[2]
  BEST_c<-c(BEST_c,best_c)
  BEST_epsilon<-c(BEST_epsilon,best_epsilon)

  ## estimate good sigma values for res_arima_train
  J<-c()
  RMSE<-c()
```

```
  SIGMA<-c()
  for(j in 1:no.iteration_svm){
    srange.train<-
sigest(RES_train_sim[2:159,i]~RES_train_sim[1:158,i],data=RES_train_sim[,i])
    ## train a support vector machine
    svm_train<-
ksvm(RES_train_sim[2:159,i]~RES_train_sim[1:158,i],data=RES_train_sim[,i],kernel="rbfd
ot",C=best_c,epsilon=best_epsilon,kpar=list(sigma=srange.train[2]))
    rmse<-rmse(RES_train_sim[2:159,i],predict(svm_train))
    J<-c(J,j)
    RMSE<-c(RMSE,rmse)
    SIGMA<-c(SIGMA,srange.train[2])
  }
  all_svm_train<-data.frame(sigma.value=SIGMA,iteration=J,RMSE=RMSE)

  ### Choose the best Sigma using "RMSE"
  i<-which.min(all_svm_train$RMSE)
  #best_svm<-all_svm_train[min,]
  best_sigma_train<-SIGMA[i]
  BEST_sigma_train<-c(BEST_sigma_train,best_sigma_train)
  rm(J,RMSE,SIGMA)
}
all_best_svm_train<-
cbind(BEST_sigma_train=BEST_sigma_train,BEST_c=BEST_c,BEST_epsilon=BEST_epsilo
n)

##############################
#######  3.2 TESTING DATA #######
##############################
### Forecast from the best SVM model
Forec_svm_test <-c()
```

```
for(i in 1:no.simmu){
  best_svm_test<-
ksvm(RES_test_sim[2:68,i]~RES_test_sim[1:67,i],data=RES_test_sim[,i],kernel="rbfdot",C
=BEST_c[i],epsilon=BEST_epsilon[i],kpar=list(sigma=BEST_sigma_train[i]))
  forec_svm_test <-predict(best_svm_test)
  Forec_svm_test <-cbind(Forec_svm_test,forec_svm_test)
}
```

Calculate "Total forecasting"(ARIMA + SVM) in testing data set

```
Total_forec_arima_svm_test<-c()
for(i in 1:no.simmu){
  #real_test_sim     <-Real_test_sim[,i]
  forec_arima_test  <-FITTED_arima_test_sim[,i]
  forec_svm_test     <-c(rep(0,1),Forec_svm_test[,i])

  total_forec_arima_svm_test<-forec_arima_test+forec_svm_test
  Total_forec_arima_svm_test<-
cbind(Total_forec_arima_svm_test,total_forec_arima_svm_test)
}
```


Computing RMSE from Hybrid ARIMA+SVM models in testing data set

```
Crite.arima.svm.test_sim<-c()
for(i in 1:no.simmu){
  crite.arima.svm.test_sim<-
regr.eval(Real_test_sim[2:68,i],Total_forec_arima_svm_test[2:68,i],stat=c("mae","mse","r
mse"))
  Crite.arima.svm.test_sim<-rbind(Crite.arima.svm.test_sim,crite.arima.svm.test_sim)
}
```

Calculate E[RMSE] from all simulations in testing data set

```
mean_error_sim_test_arima_svm <-
data.frame(Mean_MAE_test_svm=mean(Crite.arima.svm.test_sim[,1]),Mean_MSE_test_
svm=mean(Crite.arima.svm.test_sim[,2]),Mean_RMSE_test_svm=mean(Crite.arima.svm.
test_sim[,3]))
mean_error_sim_test_arima_svm
```

SUMMARY : show RMSE from SARIMA/ANN/SVM for SIMULATED DATA

```
mean_error_sim_test_arima
mean_error_sim_test_arima_ann
mean_error_sim_test_arima_svm
```

ประวัติผู้เขียนวิทยานิพนธ์

นางสาวอนุธิดา อนันต์ทรัพย์สุข เกิดเมื่อวันที่ 14 พฤษภาคม 2537 สำเร็จการศึกษาปริญญาวิทยาศาสตรบัณฑิต(คณิตศาสตร์) เกียรตินิยมอันดับ 2 คณะวิทยาศาสตร์ มหาวิทยาลัยมหิดล ปีการศึกษา 2558 และเข้าศึกษาต่อในหลักสูตรวิทยาศาสตรมหาบัณฑิต (วท.ม.) สาขาวิชาสถิติ ภาควิชาสถิติ คณะพาณิชยศาสตร์และการบัญชี จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2559