การเปรียบเทียบผลการทำนายราคาบิทคอยน์ ด้วยการเรียนรู้ของเครื่องแบบต่าง ๆ

เธียรศักดิ์ พลาดิศัยเลิศ และ ธนิศา นุ่มนนท์

คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกลาเจาคุณทหารลาดกระบัง กรุงเทพฯ

Emails: ph.thearasak@gmail.com, thanisa@it.kmitl.ac.th

บทคัดย่อ

ในช่วงหลายปีที่ผ่านมา บิทคอยน์เป็นสกุลเงินดิจิตอลที่มีมูลค่าในตลาดที่สูงที่สุด อย่างไรก็ตามราคาของบิทคอยน์มีความผันผวนมาก ทำให้ยากต่อการทำนาย ดังนั้นงานวิจัยนี้จึงมีวัตถุประสงค์ที่จะค้นหาโมเดลที่มีประสิทธิภาพและมีความแม่นยำในการทำนายราคาบิทคอยน์ โดยใช้อัลกอลิทึมของการเรียนรู้ด้วยเครื่องในแบบต่าง ๆ และใช้ข้อมูลของการแลกเปลี่ยนซื้อขายบิทคอยน์กับสกุลเงินดอลลาร์สหรัฐ (USD) จากเว็บไซต์ Bitstamp ตั้งแต่วันที่ 1 มกราคม 2012 ถึง 8 มกราคม 2018 เป็นรายนาที และนำมาทำการทดลองโดยสร้างโมเดลแบบถดถอยด้วยไลบรารี่ Scikit-learn และ Keras ซึ่งจากการประเมินผลโมเดล ได้ค่าที่ดีที่สุดของ Mean Squared Error (MSE) ต่ำสุดอยู่ที่ 0.00037 และค่าของ Coefficient of Determination (R^2) สูงสุดอยู่ที่ 99.1%

คำสำคัญ – บิทคอยน์; เงินดิจิตอล; การเรียนรู้ของเครื่อง

1. บทนำ

ในปี 2008 Satoshi Nakamoto ได้เขียนบทความ Bitcoin: peer-to-peer Electronic Cash System [1] ขึ้นด้วยแนวคิดการสร้างสกุลเงินอิเล็กทรอนิกส์ที่สามารถแลกเปลี่ยนกันได้โดยตรงผ่านเครือข่ายอินเตอร์เน็ต โดยไม่อาศัยคนกลางในการยืนยันการแลกเปลี่ยนและแชร์ข้อมูล Transaction ระหว่างกันเพื่อช่วยให้เกิดความโปร่งใสและความน่าเชื่อถือ โดยใช้หลักการที่เรียกว่า Proof of Work ในการรับรองความถูกต้องของ Transactions ซึ่งอาศัยความสามารถในการประมวลผลของคอมพิวเตอร์ในการสร้าง Hash ซึ่งผู้ที่สามารถสร้าง Hash ได้สำเร็จก่อนก็จะได้รับ บิทคอยน์ ซึ่งเรียกว่า Transaction fee เป็นสิ่งตอบแทน [1] บิทคอยน์จึงได้รับความสนใจจากผู้คนจำนวนมาก เป็นเหตุให้มูลค่าของบิทคอยน์เพิ่มสูงขึ้นตามไปด้วย จนปัจจุบันบิทคอยน์เป็นสกุลเงินดิจิตอลที่มีมูลค่าสูงที่สุดในตลาดของสกุลเงินดิจิตอล จึงดึงดูดให้กลุ่มธุรกิจต่าง ๆ รวมไปถึงนักลงทุนเข้ามาลงทุน ซื้อขายและแลกเปลี่ยน บิทคอยน์ เพื่อใช้ในการเก็งกำไรมากขึ้น ส่งผลให้ราคาของบิทคอยน์มีความผันผวนสูงตามมา โดยในปี 2017 ราคาบิทคอยน์มีการเปลี่ยนแปลงจาก 1,000 USD ในเดือน มกราคม ขึ้นไปสูงถึง 16,000 USD ในเดือนธันวาคม และมีแนวโน้มสูงขึ้นเรื่อย ๆ

งานวิจัยนี้จึงมุ่งเน้นไปที่การพยากรณ์ราคาของบิทคอยน์ด้วยอัลกอริทึมการเรียนรู้ของเครื่องในรูปแบบต่าง ๆ เพื่อค้นหาโมเดลที่มีความเหมาะสมในการทำนายราคาของบิทคอยน์ โดยใช้ข้อมูลจากการซื้อขายบิทคอยน์ ในเว็บไซต์ Bitstamp ซึ่งเป็นเว็บไซต์ซื้อขายแลกเปลี่ยนบิทคอยน์ที่มีปริมาณการซื้อขายมากเป็นอันดับต้น ๆ ของโลก [2] โดยข้อมูลที่ได้นำมาใช้ในงานวิจัยนี้ เป็นข้อมูลการซื้อขายบิทคอยน์กับเงินดอลลาร์สหรัฐ โดยมีข้อมูลการซื้อขายบิทคอยน์ ในทุก ๆ 1 นาที ตั้งแต่วันที่ 1 มกราคม 2012 ถึง 8 มกราคม 2018 ซึ่งมีผู้รวบรวมและได้เผยแพร่ผ่านเว็บไซต์ Kaggle จากข้อมูลดังกล่าวเราได้สร้างโมเดลแบบถดถอยด้วยอัลกอริทึมต่าง ๆ จากไลบรารี่สำหรับการสร้างโมเดลการเรียนรู้ของเครื่องแบบพื้นฐานด้วย Scikit-learnและสร้างโมเดลสำหรับการเรียนรู้เชิงลึกด้วย Keras

2. วรรณกรรมที่เกี่ยวข้อง

2.1. วรรณกรรมที่เกี่ยวข้อง

Shah และคณะ [3] ได้ใช้วิธีการ Bayesian Regression สำหรับการสร้างโมเดลแบบ Latent Source โดยมีการนำข้อมูลการซื้อขายบิทคอยน์จาก Okcoin ในประเทศจีนมาใช้ในการทำนายราคาบิทคอยน์ ซึ่งทำการเก็บข้อมูลทุก ๆ 10 วินาทีตั้งแต่วันที่ 6 พฤษภาคม 2014 ถึง 21 มิถุนายน 2014 และได้นำไปใช้ในการสร้างโมเดลสำหรับกำหนดกลยุทธ์ในการซื้อขายแลกเปลี่ยนบิทคอยน์ โดยคำนวณจากค่า Average Price Movement และ ค่า Threshold ในการตัดสินใจซื้อ ขายหรือการถือบิทคอยน์ไว้ จากการใช้โมเดลดังกล่าวในการซื้อขายบิทคอยน์ ทำให้ได้รับ ROI 89% ในระยะเวลา 50 วัน ด้วยระดับ Sharp Ratio ที่ 4.10

Madan และคณะ [4] ได้ใช้ข้อมูลจาก Okcoin, Coinbase API และข้อมูลภายใน Blockchain ของ Bitcoin โดยเก็บข้อมูลทุก ๆ 10 วินาที จากนั้นได้ทำการแบ่งข้อมูลเป็นข้อมูลอนุกรมเวลาเป็นช่วงละ 10 นาที และใช้วิธี Binomial Logistic Regression, Support Vector Machine (SVM) และ Random Forest ในการทำนายแนวโน้มของราคาบิทคอยน์ โดยได้ระดับความแม่นยำอยู่ที่ 97% และ 55% ในการทำนายราคา 10 นาทีถัดไป ซึ่งข้อจำกัดของงานวิจัยนี้คือไม่มีการทำ Cross-Validate ทำให้โมเดลที่ได้อาจเกิด Overfitting ได้

Greaves และคณะ [5] ได้ใช้ข้อมูล Transaction Graph มาใช้ในการทำนายราคาบิทคอยน์ โดยใช้ข้อมูลธุรกรรมภายใน Transaction ของบิทคอยน์จากเว็บไซต์ CS224W ซึ่งมีข้อมูลของ Transaction Bitcoin ตั้งแต่ 1 กุมภาพันธ์ 2012 จนถึงวันที่ 7 เมษายน 2013 และใช้ข้อมูลราคา Bitcoin จากเว็บไซต์ api.bitcoincharts.com โดยดึงข้อมูลดังกล่าวเป็นรายชั่วโมง จากนั้นทดลองนำโมเดล Linear Regression, และ SVM มาทำนายราคาเป็นรายชั่วโมง โดยได้ผลลัพธ์ดีที่สุดที่ MSE เท่ากับ 1.94 และได้ใช้ Logistic Regression, SVM และ Neural Network มาใช้ในการทำนายการขึ้นลงของราคาบิทคอยน์ ซึ่งได้ระดับความแม่นยำอยู่ที่ประมาณ 55% แต่เนื่องจากมูลค่าของบิทคอยน์ขึ้นอยู่กับพฤติกรรมการแลกเปลี่ยน ซึ่งไม่ได้เป็นข้อมูลที่อยู่ภายใน Transaction จึงส่งผลให้ระดับความแม่นยำของโมเดลนี้ยังไม่สูงเพียงพอ ผู้เขียนจึงแนะนำให้ใช้ข้อมูลการแลกเปลี่ยนซื้อขายร่วมด้วยเพื่อให้ได้ความแม่นยำที่สูงขึ้น

Almeida และคณะ [6] ได้สร้างโมเดลโครงข่ายประสาทเทียมเพื่อทำนายแนวโน้มของราคาปริมาณของบิทคอยน์ในวันถัดไป โดยใช้ข้อมูลย้อนหลังจาก Quandl ซึ่งรวบรวม Open Source Dataset ต่าง ๆ ไว้ และใช้ไลบรารี่ Theano ผ่าน MATLAB ในการสร้างโมเดลโครงข่ายประสาทเทียมโดยใช้ Levenberg Marquardt (LM) Backpropagation ในการทำนายแนวโน้มของราคาบิทคอยน์ในวันถัดไป โดยแบ่งแนวโน้มของราคาเป็น Up, Down และ Margin เพื่อช่วยในการตัดสินใจว่าควรซื้อ ขาย หรือ ถือบิทคอยน์ไว้ ซึ่งโมเดลดังกล่าวสร้างผลกำไร 8,000 USD ในการจำลองการแลกเปลี่ยนสองปี

McNally และคณะ [7] ได้ทำการทดลองสร้างโมเดลแบบ LSTM, RNN และ ARIMA โดยใช้ข้อมูล Open, High, Low และ Close จาก CoinDesk ตั้งแต่วันที่ 19 สิงหาคม 2013 ถึง กรกฎาคม 2016 และข้อมูลภายใน Blockchain จาก Blockchain.info จากนั้นสร้างโมเดลแบบถดถอย เพื่อทำนายราคาและสร้างโมเดลจำแนกข้อมูลเพื่อจำแนกการขึ้นลงของราคา โดยพบว่า LSTM ให้ความแม่นยำมากที่สุดในการจำแนกข้อมูล ส่วน RNN ให้ค่า RMSE ต่ำสุดในการทำนายราคา โดยได้ความแม่นยำสูงสุดอยู่ที่ 52.78% และค่า Root Mean Square Error (RMSE) อยู่ที่ 5.45%

2.2. เทคโนโลยีที่ใช้

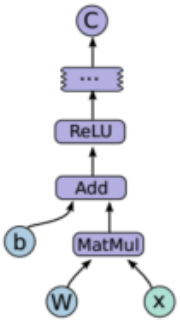
2.2.1. Scikit-learn

Scikit-learn คือ Open Source Library สำหรับทำเหมืองข้อมูลและการวิเคราะห์ข้อมูลด้วยภาษาไพธอน มีอัลกอริทึมในการสร้างโมเดลการเรียนรู้ของเครื่องเช่น โมเดลจำแนกข้อมูล (Classification), โมเดลแบบถดถอย (Regression) และ โมเดลจัดกลุ่มข้อมูล (Clustering) ตลอดจนการเตรียมข้อมูลต่าง ๆ เช่น การทำ Normalization หรือ Standardization ตลอดจนการจัดการกับข้อมูลที่ผิดปกติเช่นข้อมูลที่ไม่ได้กำหนดค่า (Missing Value) เป็นต้น [8]

2.2.2. Tensorflow

TensorFlow คือ Open Source Library สำหรับการสร้างโมเดลการเรียนรู้เชิงลึกที่พัฒนาโดย Google รองรับการทำงานแบบประมวลผลร่วมกันหลาย ๆ เครื่องและสามารถใช้ GPU ในการประมวลผลได้ และมีอัลกอริทึมสำหรับการสร้างโครงข่ายประสาทเทียมและการเรียนรู้เชิงลึกที่หลากหลาย และมีการนำไปประยุกต์ใช้กับงานวิจัยในหลายสาขาวิชาเช่น การรู้จำเสียง,การประมวลผลภาพ, สร้างหุ่นยนต์ ฯลฯ โดยการทำงานของ TensorFlow จะมีการสร้าง

กราฟสำหรับการประมวลผล ซึ่งกราฟจะประกอบไปด้วยกลุ่มของโหนด โดยกราฟจะแสดงการไหลของการประมวลผลข้อมูล ดังแสดงตัวอย่างในรูปที่ 1 [9]



รูปที่ 1. ตัวอย่างกราฟสำหรับกำหนดขั้นตอนการประมวลผลของ Tensorflow

2.2.3. Keras

Keras คือ API สำหรับ การพัฒนาโครงข่ายประสาทเทียม ซึ่งเขียนด้วยภาษาไพธอน สามารถทำงานโดยอาศัย Library อย่าง TensorFlow, CNTK หรือ Theano [10] โดย Keras สามารถรองรับการสร้างโมเดลการเรียนรู้ของเครื่องในรูปแบบโครงข่ายประสาทเทียม ตลอดจนการสร้างโมเดลการเรียนรู้เชิงลึก โดยมีข้อดีคือ ง่ายต่อการเขียนและทำความเข้าใจ มีการทำงานแยกเป็นส่วน ๆ สามารถแยกส่วนประกอบต่าง ๆ ในการสร้างโมเดลอย่างเช่น Neural Layers, Cost Functions, Optimizers, Activation Functions และประกอบกันเป็นโมเดลใหม่ได้, สามารถพัฒนาฟังก์ชันหรือคลาสต่าง ๆ เพิ่มได้ง่าย โดยการพัฒนาทั้งหมดต้องใช้ภาษาไพธอนในการพัฒนา

3. วิธีดำเนินการวิจัย

3.1. ข้อมูลที่นำมาใช้

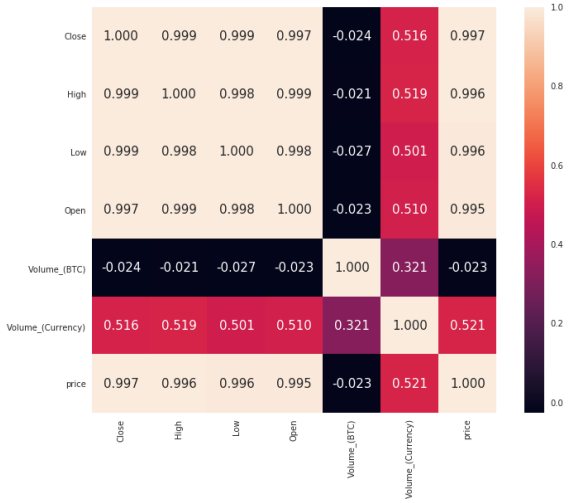
การทดลองสร้างโมเดลการเรียนรู้ของเครื่องในงานวิจัยนี้ ได้อาศัยข้อมูลการซื้อขายแลกเปลี่ยนของเว็บไซต์แลกเปลี่ยนบิทคอยน์ Bitstamp ซึ่งมีผู้รวบรวมและเผยแพร่ข้อมูลดังกล่าวไว้ในเว็บไซต์ Kaggle โดยในงานวิจัยนี้มุ่งเน้นไปที่การแลกเปลี่ยนบิทคอยน์กับสกุลเงิน USD ซึ่งข้อมูลที่ได้เป็นข้อมูลการซื้อขายตั้งแต่วันที่ 1 มกราคม 2012 จนถึงวันที่ 8 มกราคม 2018 โดยเก็บข้อมูลการซื้อขายและราคาเป็นรายนาที ในรูปแบบของไฟล์ CSV

3.2. การเลือก Features

จากข้อมูลในข้อ 3.1 มีโครงสร้างต่าง ๆ ดังนี้

- Close ราคาปิด
- Open ราคาเปิด
- High ราคาสูงสุด
- Low ราคาต่ำสุด
- Weighted price ราคาเฉลี่ย
- Volume_(BTC) ปริมาณการซื้อขาย (หน่วย BTC)
- Volume_(Currency) ปริมาณการซื้อขาย (หน่วย USD)
- Timestamp เวลาที่บันทึกข้อมูล

จากข้อมูลข้างต้นเราได้นำสร้างโมเดล ด้วย Scikit-learn และ Keras โดยนำข้อมูลดังกล่าวมาสร้างแผนภาพ Heatmap เพื่อให้เห็นค่าสัมประสิทธิ์สหสัมพันธ์ (Coefficient Correlation) ของแต่ละคุณลักษณะในข้อมูล ดังแสดงในรูปที่ 2 จะเห็นว่าคุณลักษณะที่สัมพันธ์กับมูลค่าของ Bitcoin (Weighted price) ได้แก่ Close, Open, High และ Low จึงได้เลือกคุณลักษณะดังกล่าวในการสร้างโมเดลเพื่อทำนายมูลค่าของ Bitcoin



รูปที่ 2 . แผนภาพ Heatmap แสดงค่าสัมประสิทธิ์สหสัมพันธ์ของคุณลักษณะต่าง ๆ ในชุดข้อมูล

เนื่องจากทรัพยากรของคอมพิวเตอร์ที่ใช้ในการประมวลผลไม่เพียงพอสำหรับข้อมูลรายนาที และ การสรุปข้อมูลการซื้อขายบิท

คอยน์ ในเว็บไซต์ซื้อขายบิทคอยน์จำนวนมากมีการสรุปข้อมูลการซื้อขายในรูปแบบข้อมูลรายวัน ดังนั้นการสรุปข้อมูลให้อยู่ในรูปแบบรายวันจึงเหมาะสำหรับการนำไปประยุกต์ใช้ในการใช้งานจริงมากกว่าการใช้ข้อมูลรายนาทีหรือรายชั่วโมง จากข้อมูลรายนาทีกว่า 3,161,057 แถว จากนั้นจึงปรับข้อมูลให้เป็นรายวันเหลือข้อมูลทั้งหมด 2,195 แถว จากนั้นทำการแบ่งข้อมูลออกเป็นชุดข้อมูลสำหรับการฝึกฝนโมเดล และ ชุดข้อมูลสำหรับการทดสอบประเมินผลโมเดลด้วยอัตราส่วน 70:30 ตามลำดับของชุดข้อมูล จึงเหลือชุดข้อมูลในการฝึกฝนโมเดล 1,756 ชุด หรือข้อมูลในช่วง 1,756 วันแรก และข้อมูลสำหรับทดสอบ 439 ชุด หรือข้อมูลในช่วง 439 วันล่าสุดที่เก็บข้อมูลได้

3.3. การเตรียมข้อมูล

ในการเตรียมข้อมูลเพื่อสร้างโมเดลการเรียนรู้ของเครื่องนั้น เราได้นำวิธีการแปลงข้อมูลโดยใช้ Min-Max Scaler มาใช้ในการปรับข้อมูลให้อยู่ในช่วง 0-1 ตามค่าต่ำสุดถึงค่าสูงสุดของข้อมูล [11] โดยมีวิธีการแปลงข้อมูลตามสมการที่ 1 ซึ่งสามารถใช้ Min-Max Scaler ใน Scikit-learn ในการแปลงข้อมูลตาม Features ต่าง ๆ ที่ได้เลือกไว้ในข้อ 3.2

$$x_{sc} = \frac{x - \min(x)}{\max(x) - \min(x)} \qquad (1)$$

จากนั้นทำการแปลงข้อมูลให้อยู่ในรูปที่สามารถนำไปใช้ในการสร้างโมเดล Long Short Term Memory (LSTM) และ Gated Recurrent Unit (GRU) ได้ โดยเปลี่ยนจากข้อมูลรูปแบบ 2 มิติ ให้อยู่ในรูปแบบ 3 มิติ ซึ่งมีมิติด้านเวลาเพิ่มขึ้นมา

3.4. การสร้างโมเดล

ในการทดลองสร้างโมเดลเราได้เลือกสร้างโมเดลแบบ Regression เนื่องจากค่าที่เราต้องการทำนายคือมูลค่าของ Bitcoin เมื่อเทียบกับสกุลเงิน USD เนื่องจากสกุลเงิน USD เป็นสกุลเงินที่เป็นมาตรฐานและสามารถหาข้อมูลได้ง่ายและมีตลาดซื้อขายแลกเปลี่ยน Bitcoin และเงินสกุลดังกล่าวจำนวนมาก โดยมูลค่าของ Bitcoin นั้นเป็นค่าที่มีความต่อเนื่องกัน (Continuous-values) เราจึงต้องเลือกโมเดลประเภทถดถอย (Regression Model) โดยการสร้างโมเดลประเภทดังกล่าวนั้นผู้เขียนได้นำไลบรารี่ Keras มาใช้ในการสร้างโมเดลการเรียนรู้เชิงลึกด้วยวิธี LSTM และ GRU นอกจากนี้ยังมีการใช้ Scikit-learn โมเดล Regression ในรูปแบบต่าง ๆ และได้เลือกโมเดลที่ให้ผลที่ดีที่สุดมาเปรียบเทียบกับวิธีที่กล่าวถึงในหัวข้อวรรณกรรมที่เกี่ยวข้อง

3.4.1. Theil-sen Regression

Theil-Sen Regression เป็นวิธีที่ใช้การประมาณค่าโดยใช้ค่ามัธยฐานของความชันของทุกเส้นที่ผ่านคู่อันดับของข้อมูล จึงทำให้มีความทนทานต่อข้อมูลที่เป็น outliers ถึง 29.3% สำหรับข้อมูลที่อยู่ในรูป 2 มิติ หรือคู่อันดับ อย่างไรก็ตามหากข้อมูลมีมิติที่มากขึ้นก็จะส่งผลต่อความคงทนต่อค่าผิดปกติ (Outliers) ให้ลดลงได้ [12]

โดยในการสร้างโมเดล Regression ด้วย Theil-Sen Regression โดยใช้ Scikit-learn นั้น เราได้กำหนดค่า Parameter ต่าง ๆ เป็นค่าเริ่มต้นของ Scikit-learn ดังแสดงในตารางที่ 1

ตารางที่ 1. ค่า Parameter ต่าง ๆ ที่ใช้ในการสร้าง Model Theil-sen Regression

Parameter	Value
max_subpopulation	10000
max_iter	300
tol	0.001

3.4.4. Huber Regression

Huber Regression มีการใช้ Linear Loss ในการจำแนกกลุ่มตัวอย่างของข้อมูลที่มีค่าผิดปกติและข้อมูลที่เป็นค่าปกติ (Inlier) โดยจะให้ Weight ของข้อมูลที่อยู่ในค่าผิดปกติเป็นค่าที่น้อยกว่าข้อมูลที่เป็นค่าปกติ [13]

โดยในการสร้างโมเดล Huber Regression ด้วย Scikit-learn นั้นเราได้กำหนดค่าพารามิเตอร์ต่าง ๆ เป็นค่าเริ่มต้น ดังนี้

- Epsilon คือค่าที่ใช้กำหนดกลุ่มตัวอย่างที่ควรจะถูกจัดเป็น Outlier โดยค่าน้อยจะทำให้โมเดลมีความคงทนต่อ Outlier ซึ่งค่าเริ่มต้นเป็น 1.35 โดยค่าต้องมากกว่า 1
- Alpha คือ Regularization Parameter มีค่าเป็น 0.0001

ดังแสดงในตารางที่ 2

ตารางที่ 2. ค่า Parameter ต่าง ๆ ที่ใช้ในการสร้าง Huber Regression

Parameter	Value
epsilon	1.35
max_iter	100
alpha	0.0001
tol	0.00001

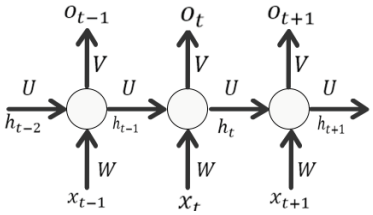
3.4.5. Recurrent Neural Network (RNN)

ในอัลกอริทึมแบบโครงข่ายประสาทเทียมมีอัลกอริทึมสำหรับการสร้างโมเดล ที่เหมาะกับข้อมูลที่เป็นอนุกรมเวลา เรียกว่า Recurrent neural network (RNN) [14] ซึ่งจะมีการเป็นข้อมูลสถานะไว้ใน Hidden State โดยมีการนำ Hidden State ก่อนหน้ามาใช้ในการคำนวณ Hidden State ปัจจุบัน และใช้ Hidden State ปัจจุบันในการคำนวณของข้อมูลในช่วงเวลาถัดไป โดยมีการคำนวณตามสมการที่ 3.2 และ 3.3

$$h_t = \sigma(x_t W + h_{t-1} U) \quad (2)$$

$$o_t = \sigma(h_t V) \quad (3)$$

โดยที่ t คือช่วงเวลา, h คือ hidden state, x หมายถึงข้อมูล input, o คือค่า output, σ คือ activation function และ W, U, V คือ weight matrix สำหรับคำนวณ input, hidden state ก่อนหน้า และ hidden state ปัจจุบัน ตามลำดับ และมีโครงสร้างตามรูปที่ 3



รูปที่ 3. โครงสร้างของ Recurrent Neural Network

RNN จึงเหมาะกับข้อมูลที่เป็นลำดับหรือข้อมูลที่เป็นอนุกรมเวลา โดยการใช้ RNN นั้นสามารถใช้ได้กับข้อมูลที่มีลำดับเวลาไม่ห่างกันมากนัก แต่จะประสบปัญหากับข้อมูลที่มีลำดับเวลาที่ห่างกัน (long-term dependencies) จะก่อให้เกิดปัญหา vanishing gradient ตามมา

3.4.6. Long Short-Term Memory (LSTM)

จากปัญหา vanishing gradient ใน RNN จึงได้มีการพัฒนา LSTM [15] ขึ้นโดยจะใช้ข้อมูล cell state และ hidden state ในการเก็บข้อมูลและส่งไปประมวลผลยังช่วงเวลาถัดไป ซึ่งอาศัย gate ต่าง ๆ ในการคำนวณว่าควรจะเก็บรักษาข้อมูลภายใน cell state และ hidden state มากน้อยเพียงใด โดยประกอบไปด้วย input gate, output gate, และ forget gate แต่ละ gate จะมีหน้าที่ในการตัดสินใจว่าจะให้ข้อมูลผ่านไปหรือไม่ ตามค่าความสำคัญของข้อมูล หากมีค่าน้อยก็จะไม่สามารถผ่าน gate ไปได้จึงช่วยป้องกันการเกิด vanishing gradient ได้ โดยมีการคำนวณค่าต่าง ๆ ดังสมการที่ 4- 8

$$i = \sigma(x_t W_i + h_{t-1} U_i) \quad (4)$$

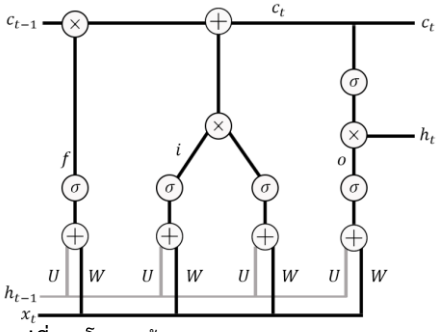
$$f = \sigma(x_t W_f + h_{t-1} U_f) \quad (5)$$

$$o = \sigma(x_t W_o + h_{t-1} U_o) \quad (6)$$

$$c_t = (c_{t-1} \times f) + (i \times \sigma(x_t W_c + h_{t-1} U_c)) \quad (7)$$

$$h_t = \sigma(c_t) \times o \quad (8)$$

โดยที่ i คือ input gate, f คือ forget gate, o คือ output gate, c คือ cell state, h คือ hidden state, σ คือ Activation function, W และ U คือ weight matrix และ t แทนช่วงเวลา โดยสามารถแสดงโครงสร้างได้ดังรูปที่ 4



รูปที่ 4. โครงสร้างของ Long Short Term Memory

โดยในการสร้างโมเดล LSTM นั้นเราได้กำหนด parameter ต่าง ๆ ดังแสดงในตารางที่ 3

ตารางที่ 3. ค่า Parameter ต่าง ๆ ที่ใช้ในการสร้าง Long Short Term Memory

Parameter	Value
epoch	450
Batch size	160
Hidden node	259
Hidden layer	1

3.4.7. Gated Recurrent Unit (GRU)

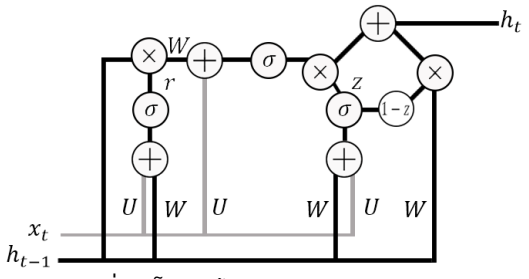
ในปี 2014 Kyunghyun Cho และคณะได้เสนอวิธี Gate Recurrent Unit (GRU) [16] โดยพัฒนาต่อยอดจาก LSTM ให้มีโครงสร้างที่ไม่ซับซ้อนโดยการปรับ gate ใน LSTM เป็น Reset Gate และ Update Gate ซึ่ง Update Gate ทำหน้าที่พิจารณาว่าควรจะเก็บข้อมูล State ก่อนหน้าไว้มากน้อยเพียงใด และ Reset Gate ทำหน้าที่คำนวณว่าจะนำข้อมูลจากจาก State ก่อนหน้ามาพิจารณาร่วมกับข้อมูล Input ปัจจุบันมากน้อยเพียงใด โดยมีการคำนวณค่าต่าง ๆ ดังสมการที่ 9- 11

$$r = \sigma(x_t W_r + h_{t-1} U_r) \quad (9)$$

$$z = \sigma(x_t W_z + h_{t-1} U_z) \quad (10)$$

$$h_t = \left(z \times \sigma\left((h_{t-1} \times r) W_h + (x_t U_h)\right)\right) + \left((1 - z) \times h_{t-1}\right) \quad (11)$$

โดยที่ r คือ Reset Gate, z คือ Update Gate, h คือ Hidden State, σ คือ Activation Function, W และ U คือ Weight Matrix และ t แทนช่วงเวลา โดยสามารถแสดงโครงสร้างได้ดังรูปที่ 5



รูปที่ 5. โครงสร้างของ Gated Recurrent Unit

โดยในการสร้างโมเดล GRU นั้นเราได้กำหนด Parameter ต่าง ๆ ดังแสดงในตารางที่ 4

ตารางที่ 4. ค่า Parameter ต่าง ๆ ที่ใช้ในการสร้าง Gated Recurrent Unit

Parameter	Value
epoch	480
Batch size	160
Hidden node	231
Hidden layer	1

4. ผลการดำเนินงาน

4.1. ผลการทดลอง

ในการสร้างโมเดลแบบต่าง ๆ ตามที่กล่าวมานั้นนำมาประกอบกับโมเดลต่าง ๆ ที่ได้กล่าวถึงในวรรณกรรมที่เกี่ยวข้อง ซึ่งงานวิจัยนี้ได้ทำการวัดผลของโมเดลโดยใช้ค่า MSE และ R^2 และนำมาเปรียบเทียบผลดังแสดงในตารางที่ 5 จะเห็นว่าโมเดลถดถอยแบบ Theil-Sen Regression และ Huber Regression ให้ผลที่ใกล้เคียงกันที่ค่า MSE ที่ 0.00037 และมีค่า R^2 อยู่ที่ 0.991 หรือ 99.1% ดีกว่าโมเดลแบบ การเรียนรู้เชิงลึกที่เราสร้างขึ้นคือ Long Short Term Memory ซึ่งให้ค่า MSE ที่ 0.00043 และค่า R^2 เท่ากับ 0.990 หรือ 90% และ Gated Recurrent Unit ซึ่งให้ค่าที่ MSE เป็น 0.00046 และ R^2 ที่ 0.989 หรือ 98.9%

และได้แสดงผลการเปรียบเทียบระยะเวลาในการสร้างโมเดลแต่ละแบบไว้ในตารางที่ 6 จะเห็นว่าโมเดลถดถอยแบบ Theil-Sen Regression และ Huber Regression ใช้เวลาในการสร้างโมเดลน้อยกว่าโมเดลแบบการเรียนรู้เชิงลึก โดยโมเดลที่ใช้เวลาในการสร้างน้อยที่สุดได้แก่ Bayesian Regression โดยใช้เวลาเพียง 0.00003 วินาที และโมเดลที่ใช้เวลานานที่สุดคือโมเดลแบบ LSTM โดยใช้เวลาสร้างโมเดลที่ 111.0601 วินาที ทั้งนี้ระยะเวลาในการฝึกสอนโมเดลขึ้นอยู่กับประสิทธิภาพหรือทรัพยากรของเครื่องคอมพิวเตอร์ที่ใช้ฝึกสอนโมเดล

ตารางที่ 5. ค่า MSE และ R^2 ของโมเดลต่าง ๆ ที่ได้จากการทดลอง

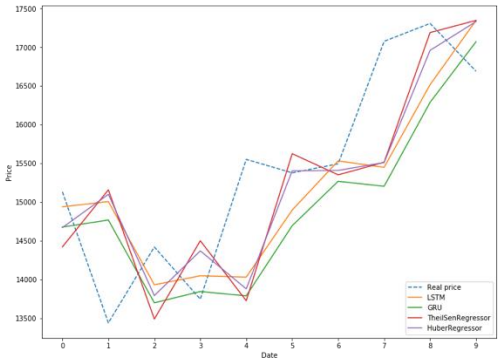
Model	MSE	R^2
Linear Regression	0.00038	0.991
Theil-Sen Regression	0.00037	0.991
Huber Regression	0.00037	0.991
Bayesian Regression	0.00038	0.991
LSTM	0.00043	0.990
GRU	0.00046	0.989

ตารางที่ 6. เปรียบเทียบเวลาที่ใช้ในการสร้างโมเดลแต่ละแบบ

Model	Time (sec)
Linear Regression	0.0041
Theil-Sen Regression	0.9018
Huber Regressor	0.0002
Bayesian Regression	0.00003
LSTM	111.0601
GRU	85.2718

4.2. กราฟเปรียบเทียบผลการทดลอง

จากโมเดลรูปแบบต่าง ๆ เราได้สร้างตารางเปรียบเทียบผลการทำนายราคาบิทคอยน์ และราคาจริงในชุดข้อมูลสำหรับทดสอบในช่วง 10 วันหลังสุดในแต่ละโมเดล โดยแสดงในรูปที่ 6 และในตารางที่ 7 และมีค่าความคลาดเคลื่อนของผลการทำนายในแต่ละโมเดล (Error) ดังแสดงในตารางที่ 8



รูปที่ 6. แผนภาพแสดงการเปรียบเทียบผลการทำนายของโมเดลแบบต่าง ๆ กับราคาบิทคอยน์

ตารางที่ 7. เปรียบเทียบผลการทำนายและราคาบิทคอยน์ในโมเดลแต่ละแบบ

Bitcoin Price (USD)	Theil-Sen Regression Predicted Price (USD)	Huber Regression Predicted Price (USD)	LSTM Predicted Price (USD)	GRU Predicted Price (USD)
14431.58	15509.51	15733.71	15974.69	15702.57
15133.45	14416.55	14672.45	14940.96	14679.41
13443.46	15185.55	15102.18	15008.14	14770.15
14421.79	13439.63	13794.82	13934.65	13701.48
13748.86	14511.93	14372.27	14050.10	13847.21
15553.45	13719.58	13881.75	14033.01	13791.67
15379.56	15649.79	15405.97	14898.36	14697.20
15496.05	15367.06	15410.66	15533.38	15270.50
17077.18	15516.43	15512.61	15450.08	15205.52
17308.17	17211.64	16960.66	16517.99	16289.66

ตารางที่ 8. เปรียบเทียบผลค่าความคลาดเคลื่อนของการทำนายราคาบิทคอยน์ของโมเดลแต่ละแบบ

Theil-Sen Regression Predicted Price (USD)	Huber Regression Predicted Price (USD)	LSTM Predicted Price (USD)	GRU Predicted Price (USD)
1077.93	1302.13	1543.12	1270.99
716.90	461.00	192.49	454.04
1742.09	1658.72	1564.68	1326.69
982.15	626.96	487.13	720.30
763.07	623.41	301.24	98.35
1833.87	1671.70	1520.45	1761.78
270.23	26.41	481.20	682.36
128.99	85.38	37.33	225.55
1560.75	1564.57	1627.10	1871.66
96.53	347.51	790.18	1018.52

5. สรุป

จากผลการทดลองสร้าง Regression ด้วยอัลกอริทึมในรูปแบบต่าง ๆ และสร้างกราฟแสดงการเปรียบเทียบราคาจริงและราคาที่ได้จากการทำนายของโมเดลแต่ละชนิดจะเห็นว่ามีประสิทธิภาพใกล้เคียงกัน แต่โมเดล Huber Regressor และ Theil-Sen Regression ให้ผลที่ดีที่สุดที่ในด้านความแม่นยำโดยให้ค่า MSE เป็น 0.00037 และค่า R^2 ที่ 0.991 ส่วนในด้านระยะเวลาที่ใช้ในการฝึกฝนโมเดลนั้น Bayesian Regression เป็นโมเดลที่ใช้เวลาในการฝึกฝนโมเดลน้อยที่สุดที่ระยะเวลา 0.00003 วินาที แต่หากพิจารณาในแง่ของระยะเวลาที่ใช้ในการฝึกฝนโมเดลและค่า MSE และ R^2 ของโมเดลจะพบว่า Huber Regressor นั้นให้ผลที่ดีที่สุดโดยค่า MSE ที่ 0.00037 และ R^2 เท่ากับ 0.991 โดยใช้เวลาในการฝึกฝนโมเดลเพียง 0.0002 วินาทีเท่านั้น ทั้งนี้ค่าความแม่นยำของโมเดลยังขึ้นอยู่กับการตั้งค่าต่าง ๆ ของโมเดล และจำนวนชุดข้อมูลที่มีด้วย จากข้อมูลที่นำมาใช้จะเห็นว่าจำนวน Features ที่นำมาใช้มีเพียง 4 ตัวคือ Open, Close, High, Low เท่านั้น ซึ่ง Features ดังกล่าวอาจไม่เพียงพอที่จะใช้ในการพยากรณ์ราคาของ Bitcoin ได้ เนื่องจากราคาของบิทคอยน์ นั้นขึ้นอยู่กับปัจจัยต่าง ๆ มากมาย เช่น สื่อต่าง ๆ นโยบายและกฎหมายที่แต่ละประเทศประกาศขึ้นเพื่อรับมือกับสกุลเงินดิจิตอล ตลอดจนปฏิกริยาจากสื่อสังคมออนไลน์ ต่าง ๆ ก็ล้วนส่งผลต่อการขึ้นลงและความผันผวนของราคาบิทคอยน์ทั้งสิ้น

ดังนั้นหากต้องการโมเดลที่มีประสิทธิภาพมากยิ่งขึ้นควรเพิ่มข้อมูลที่ใช้ในการฝึกฝนโมเดล โดยเพิ่มข้อมูลที่เป็นข้อมูลที่ไม่มีโครงสร้าง เช่น ข้อความจาก สื่อต่าง ๆ ตลอดจน สังคมออนไลน์ ที่มีการพูดถึง บิทคอยน์ เพื่อให้ได้ข้อมูลต่าง ๆ ที่เป็นปัจจัยที่ส่งผลต่อราคาของบิทคอยน์มากยิ่งขึ้น

เอกสารอ้างอิง

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[2] J. Chu, S. Nadarajah, and S. Chan, "Statistical analysis of the exchange rate of bitcoin," *PloS one,* vol.10, no. 7, pp. e0133678, July. 2015.

[3] D. Shah, and K. Zhang, "Bayesian regression and Bitcoin," In *Conf. Annual Allerton Conference on Communication, Control, and Computing (Allerton),* Illinois, USA, Oct. 2014, pp. 409-414.

[4] I. Madan, S. Saluja, and A. Zhao. "Automated Bitcoin Trading via Machine Learning Algorithms," Dept. Computer Science, Stanford University, Stanford, CA, 2015.

[5] A. Greaves, and B. Au "Using the Bitcoin Transaction Graph to Predict the Price of Bitcoin," Stanford University, Stanford, CA, 2015.

[6] J Almeida, S Tata, A Moser, and V Smit, "Bitcoin prediciton using ANN," *Neural Network,* vol.7, pp. 1-12, June. 2015.

[7] S. McNally, J. Roche, and S. Caton, "Predicting the price of Bitcoin using Machine Learning," In *Parallel, Distributed and Network-based Processing (PDP), 2018 26th Euromicro International Conference,* Cambridge, UK, March. 2018, pp. 339-343.

[8] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, and R. Layton, “API design for machine learning software: experiences from the scikit-learn project,” In *European Conference on Machine Learning and Principles and Practices of Knowledge Discovery,* Sep. 2013.

[9] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, and M. Kudlur, “Tensorflow: a system for large-scale machine learning,” OSDI vol. 16, pp. 265-283, Nov. 2016.

[10] F. Chollet. (2018) .Keras: Deep learning library for theano and tensorflow. [Online]. Available: *https://keras. io/k*

[11] S. Patro, and K.K. Sahu, "Normalization: A preprocessing stage," Dept. CSE & IT, VSSUT, Burla, Odisha, India, 2015.

[12] X. Dang, H. Peng, X. Wang, and H Zhang, "Theil-Sen Estimators in a Multiple Linear Regression Model." Dept. Mathematics, University of Mississippi, Mississippi, MS, and Dept. Epidemiology and Public Health, Yale University, New Haven, CT, 2008.

[13] P.J. Huber, "Robust statistics," In *International Encyclopedia of Statistical Science* Springer, Heidelberg, Berlin, 2011, pp. 1248-1251.

[14] A. Bernal, S. Fok, and R. Pidaparthi, "Financial Market Time Series Prediction with Recurrent Neural Networks." Dec. 2012.

[15] S. Hochreiter, and J. Schmidhuber, "Long short-term memory," *Neural computation,* vol. 9, no. 8 pp. 1735-1780, Nov. 1997.

[16] K. Cho, B.V. Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, Oct. 2014, pp. 1724-1734