การพยากรณ์-ความแม่นยำ เพื่อวิเคราะห์ประมาณการยอดขายสินค้าขายปลีก กรณีศึกษา Walmart

Prediction- Accuracy to Analyze Retail Seller Sales Data Walmart Case Study

ธัญญ์รวี ชวรัตน์ธนรังษี, ปุณยภัสร์ ชวรัตน์ธนรังษี, ศรัญธร มั่งมี, กันทิมา คงสถิตสุวรรณ, ณรงค์ฤทธิ์ สุคนธสิงห์

กลุ่มงา เทคโนโลยีสารสนเทศ, บริษัท กรุงไทยธรุกิจลีสซิ่ง จำกัด, thunravee.chawaratthanarungsri@ktbleasing.co.th

คณะเทคโนโลยีสารสนเทศ, มหาวิทยาลัยสยาม, punyapas.cha@siam.edu

คณะเทคโนโลยีสารสนเทศ, มหาวิทยาลัยสยาม, saranthon.mau@siam.edu

คณะเทคโนโลยีสารสนเทศ, มหาวิทยาลัยสยาม, kanthima.kon@siam.edu

คณะเทคโนโลยีสารสนเทศ, มหาวิทยาลัยสยาม, narongrit@siam.edu

บทคัดย่อ

ในการศึกษานี้นำเสนอการวิเคราะห์ "การพยากรณ์-ความแม่นยำ" ประมาณการยอดขายสินค้าขายปลีกของ Walmart โดยมีวัตถุประสงค์เพื่อพยากรณ์ยอดขายของร้านค้าล่วงหน้า 28 วัน ของ Supermarket “Walmart” เนื่องจากสินค้าส่วนใหญ่เป็นสินค้าอุปโภคบริโภคจึงไม่สามารถ stock ได้ระยะนาน การพยากรณ์นี้เพื่อช่วยในเรื่องของการปริหารจัดการคลังสินค้าในแต่ละสาขาให้เหมาะสมกับปริมาณการซื้อ การวิจัยนี้ใช้กระบวนการของ Deep Learning โดยใช้ภาษา Python ในการพัฒนา ซึ่งผลลัพธ์ที่ได้จะสามารถนำไปวิเคราะห์ยอดขาย เพื่อใช้งานการณ์พยากรณ์การ Stock สินค้าได้อย่างเหมาะสม

คำหลัก: การพยากรณ์, อนุกรมเวลาความแม่นยำ, การเรียนรู้อย่างลึก, การพยากรณ์ยอดค้าปลีก

Abstract

In this study, an analysis is presented. "Prediction-Accuracy" Walmart Retail Sales Estimates. The objective is to forecast the sales of stores 28 days in advance of the Supermarket "Walmart". Because most of the products are consumer products, they cannot be stocked for a long time. This forecast helps in managing the warehouse in each branch to suit the purchase volume. This research uses the process of Deep Learning using the Python language to develop, the results of which can be used to analyze sales. in order to use the forecasting of stock products appropriately

Keywords: Forecasting, Accuracy Time Series, Deep Learning, Retail Sales forecasting

ความเป็นมาและความสำคัญของปัญหา

The Makridakis Open Forecasting Center (MOFC) at the University of Nicosia [1] ได้เปรียบเทียบประสิทธิภาพของวิธีการพยากรณ์แบบต่างๆ ในการแก้ปัญหาให้กับบริษัทต่างๆ รวมถึงการ ประเมินระดับความไม่แน่นอนเพื่อหลีกเลี่ยงข้อผิดพลาดที่มีผลทำให้ค่าใช้จ่ายสูง

Walmar Supermarket พบปัญหาเรื่องการ Stock สินค้าในช่วงวันหยุด โดยเฉพาะวันหยุดในช่วงสิ้นปี ที่ไม่สามารถบริหารจัดการคลังสินค้าในแต่ละสาขาได้อย่างมีประสิทธิภาพ โดยที่ความต้องการซื้อในแต่ละสาขานั้นไม่เหมือนกัน รวมทั้งของส่วนใหญ่เป็นสินค้าอุปโภคบริโภคซึ่งไม่สามารถ Stock สินค้าได้นานมาก จึงใช้การพยากรณ์ล่วงหน้าเพียง 28 วันข้างหน้า

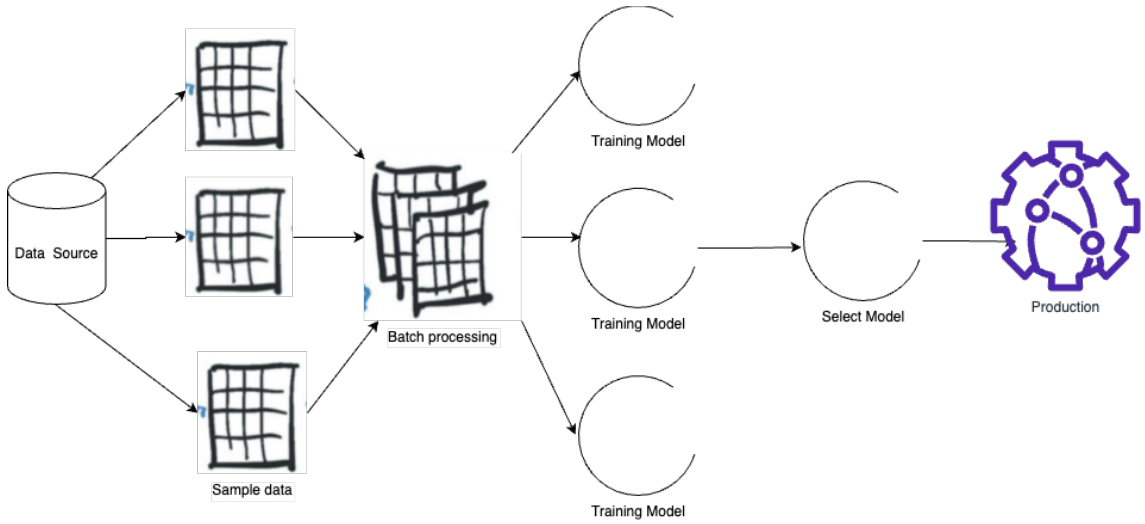
วัตถุประสงค์

การพยากรณ์ – ความแม่นยำยอดขายในอนาคตของผลิตภัณฑ์แต่ละรายการในแต่ละสาขาตามข้อมูลการขายแบบลำดับชั้นของบริษัท Walmart โดยคาดการณ์ยอดขายรายวันสำหรับ 28 วันข้างหน้า

ประโยชน์ที่คาดว่าจะได้รับ

1. เพื่อพยากรณ์ยอดขายในช่วง 28 วันข้างหน้าของบริษัท Walmart ได้อย่างแม่นยำมากขึ้น
2. เปรียบเทียบวิธีการพยากรณ์ใน Model ต่างๆ เพื่อให้ได้ Model ที่เหมาะสมที่สุด
3. ช่วยในการวางแผนการ Stock สินค้าในหมวดต่างๆ รวมทั้งต่างสาขากันอาจจะมีความต้องการซื้อที่แตกต่างกันด้วย

กรอบแนวคิด



ภาพประกอบ 1 : กรอบแนวความคิด

จากภาพประกอบที่ 1 แสดงถึงกรอบแนวความคิดของงานวิจัยนี้ โดยเริ่มจากนำข้อมูลย้อนหลังของ Walmart ในปี 2011-2016 มาโดยเลือกสุ่มตัวอย่างมากจาก 3 รัฐ และ 3 กลุ่มสินค้า

การจัดเก็บข้อมูลยอดการขายของ Walmart ถูกจัดเก็บไว้ใน database ในรูปแบบของ Batch File หลังจากนั้นดึง Sample ตัวอย่างของข้อมูลการขายจาก สาขาในรัฐที่ถูกสุ่ม และเลือกกลุ่มสินค้าที่เหมือนกันเพื่อนำมาแปลงให้อยู่ใน File .csv ใช้ Python ในการ develop แต่ละ Deep learning model โดยใช้ข้อมูลไฟล์ csv. เป็นข้อมูล Input ของแต่ละ model ในแต่ละ model จะต้องหาค่าประสิทธิภาพว่า model ใดที่ให้ค่าประสิทธิภาพที่ดีที่สุด ซึ่งในที่นี้เราจะพิจารณาถึงความเร็วในการประมวลผลของแต่ละ Model ด้วย เมื่อได้ Model ที่เหมาะสมที่สุด จะใช้ Model นั้นในการพยากรณ์ยอดขายของแต่ละสาขาต่อไป

วิธีดำเนินการวิจัย

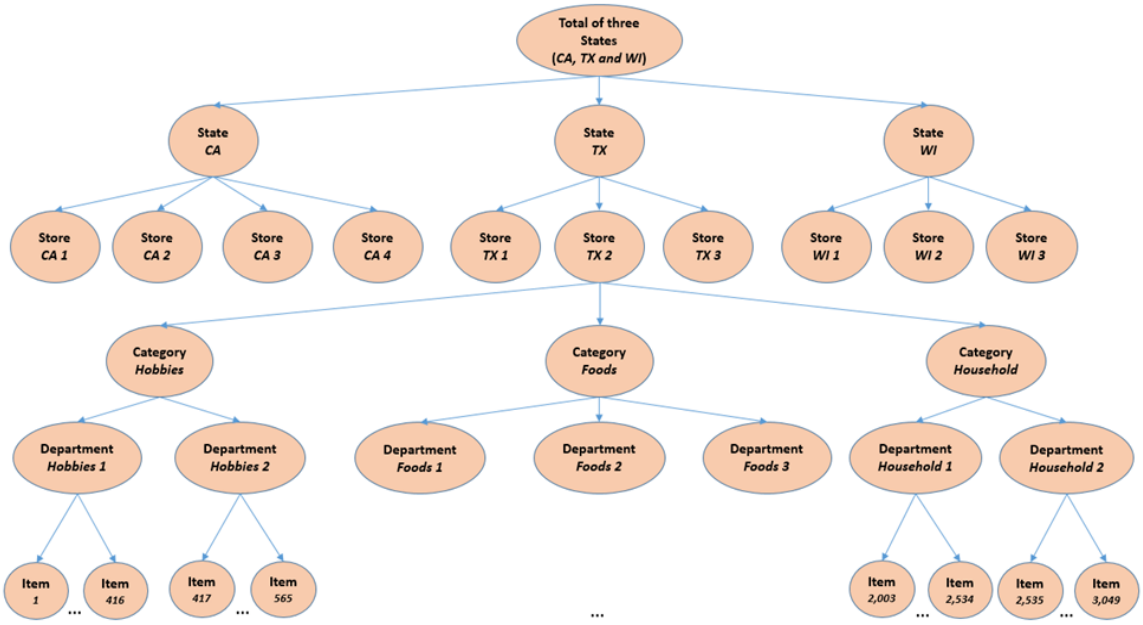
Walmart เป็นบริษัทค้าปลีกที่มีชื่อเสียงแห่งหนึ่ง โดยมีสาขาทั้งสิ้น 45 สาขา การจัดเก็บข้อมูลยอดขายจะถูกแบ่งออกเป็นยอดขายรายวันและรายสัปดาห์ Walmart พยายามค้นหาผลกระทบของวันหยุดที่มีผลต่อการบริหารจัดการคลังสินค้า ซึ่งส่งผลกระทบของยอดขายในแต่ละสาขาที่แตกต่างกัน รวมถึงต้องการที่จะพยากรณ์ล่วงหน้าเพื่อเตรียมสำหรับวันหยุดเทศกาล เช่น คริสต์มาส วันขอบคุณพระเจ้า ซูเปอร์โบวล์ วันแรงงาน เป้าหมายของงานนี้คือการพยากรณ์ข้อมูลการขายจากบริษัท Walmart ด้านการค้าปลีกในอีก 28 วันข้างหน้า โดยข้อมูลประกอบด้วยตัวอย่างสินค้า 42,840 sku จาก 3 หมวดหมู่และ 7 แผนก และมีสาขาทั้งหมด 10 สาขา ใน 3 รัฐของสหรัฐอเมริกา

Data Exploration: การสำรวจข้อมูล

Data Overview : ชุดข้อมูลนี้เป็นข้อมูลการขายต่อหน่วยตามลำดับชั้นจากบริษัท Walmart ข้อมูลประกอบด้วยราคาต่อผลิตภัณฑ์ ระดับสินค้า แผนก หมวดหมู่สินค้า และรายละเอียดสาขา รวมถึงวันที่มีโปรโมชั่นในแต่ละสัปดาห์ และกิจกรรมพิเศษ

ข้อมูลเกี่ยวข้องกับการขายต่อหน่วยของตัวอย่างสินค้า 42,840 sku โดยจำแนก 3 หมวดหมู่ผลิต(งานอดิเรก อาหาร และของใช้ในครัวเรือน) และ 7 แผนกผลิตภัณฑ์ ที่มีจำหน่ายในแต่ละสาขาที่ตั้งอยู่ในสามรัฐ โดยจำแนกเป็น 3 รัฐในสหรัฐอเมริกา ได้แก่ แคลิฟอร์เนีย (CA) เท็กซัส (TX) และวิสคอนซิน (WI) โดยข้อมูลจะถูกจัดเก็บในรูปแบบของไฟล์ .esv ดังนั้น เมื่อใช้ข้อมูลนี้เราจะคาดการณ์ยอดขายรายวันสำหรับ 28 วันถัดไปได้ โดยเราจะเรียกข้อมูลที่ใช้ในการทดสอบว่า M5

การสุ่มตัวอย่างโดย 1. สุ่มรัฐ ได้แก่ แคลิฟอร์เนีย (CA) เท็กซัส (TX) และวิสคอนซิน (WI) จากนั้นในแต่ละรัฐทำการสุ่มแบบหลายขั้นตอน (Stratified Random Sampling) คือหมวดหมู่ผลิตภัณฑ์สินค้าตามจริงและสุ่มสินค้าโดยวิธีการสุ่มอย่างง่าย (Simple Random Sampling) 42,849 sku



ภาพประกอบ 2 : แผนการสุ่มตัวอย่าง

ตัวแปรที่ใช้ในการวิจัยประกอบด้วย

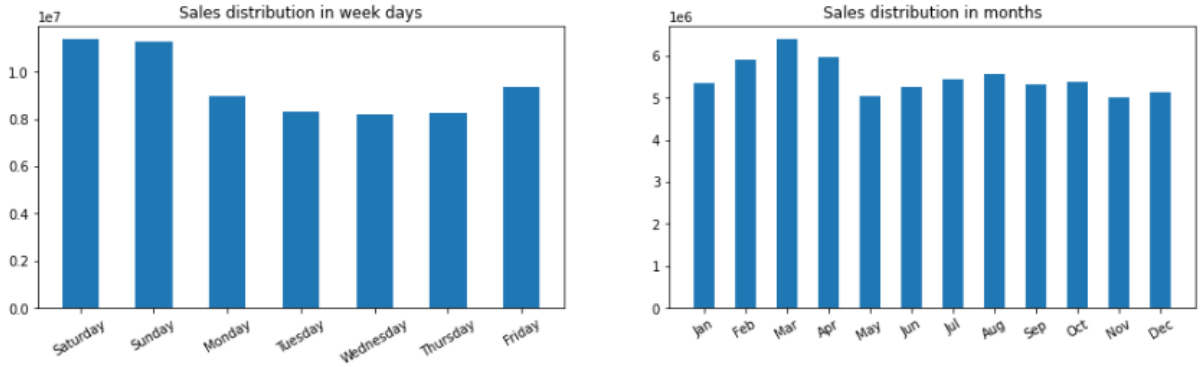
- date: The date in a “y-m-d” format.
- wm_yr_wk: The id of the week the date belongs to.
- weekday: The type of the day (Saturday, Sunday, ..., Friday).
- wday: The id of the weekday, starting from Saturday.
- month: The month of the date.
- year: The year of the date.
- event_name_1: If the date includes an event, the name of this event.
- event_type_1: If the date includes an event, the type of this event.
- event_name_2: If the date includes a second event, the name of this event.
- event_type_2: If the date includes a second event, the type of this event.
- snap_CA, snap_TX, and snap_WI: A binary variable (0 or 1) indicating whether the stores of CA, TX or WI allow SNAP purchases on the examined date. 1 indicates that SNAP purchases are allowed.

ข้อมูลเกี่ยวกับราคาของผลิตภัณฑ์ที่ขายตามร้านค้าและวันที่

- store_id: The id of the store where the product is sold.
- item_id: The id of the product.
- wm_yr_wk: The id of the week.
- sell_price: The price of the product for the given week/store. The price is provided per week (average across seven days). If not available, this means that the product was not sold during the examined week. Note that although prices are constant at weekly basis, they may change through time (both training and test set).

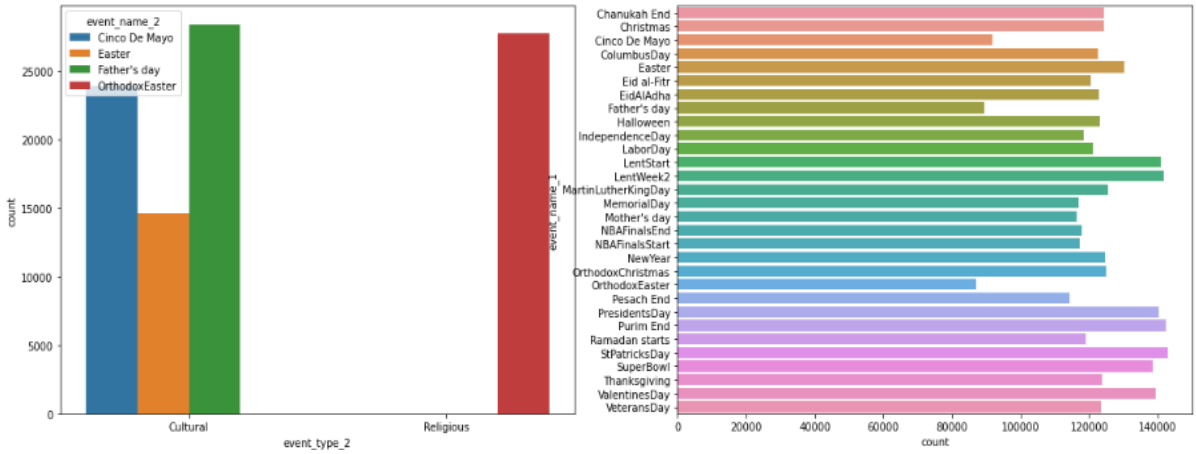
รวมยอดขาย [d_1 ถึง d_1941]

- item_id: The id of the product.
- dept_id: The id of the department the product belongs to.
- cat_id: The id of the category the product belongs to.
- store_id: The id of the store where the product is sold.
- state_id: The State where the store is located.
- d_1, d_2, ..., d_i, ... d_1941: The number of units sold at day i, starting from 2011-01-29.



ภาพประกอบ 3 :ข้อมูลการขายต่อหน่วยรายวันในอดีตต่อผลิตภัณฑ์และร้านค้า (2001-2016)

จะเห็นว่าวันเสาร์และวันอาทิตย์จะมียอดขายสูงกว่าวันอื่นๆ และการแจกแจงรายเดือนก็แตกต่างกันไปในแต่ละเดือน โดยมีการระบุวันในแต่ละสัปดาห์และคุณลักษณะหมวดหมู่ที่สำคัญ เพื่อช่วยในการคาดการณ์



ภาพประกอบ 4 : ข้อมูลวันหยุดสำหรับปฏิทินและประเภทวันหยุด (2017)

จากภาพประกอบที่ 4 แสดงให้เห็นว่าคอลัมน์เหตุการณ์จะแสดงข้อมูลวันหยุดตามปฏิทิน และประเภทวันหยุดที่แตกต่างกันไป ดังนั้นสำหรับฟีตเจอร์หมวดหมู่จึงไม่จำเป็นต้องการประเภทของกิจกรรม แต่ต้องการเพียงแค่สถานะเป็นวันหยุดเท่านั้นที่จะแสดงถึงยอดขายในช่วงวันหยุด [3]

Performance Metric : การวัดวัดประสิทธิภาพ

$$\boldsymbol{RMSSE} = \sqrt{\frac{1}{h}\frac{\sum_{t=n+1}^{n+h}(Y_t - \hat{Y}_t)^2}{\frac{1}{n-1}\sum_{t=2}^{n}(Y_t - Y_{t-1})^2}}, \qquad \boldsymbol{WRMSSE} = \sum_{i=1}^{42{,}840} w_i * \boldsymbol{RMSSE}$$

ความแม่นยำของการคาดการณ์จะได้รับการประเมินโดยใช้ Root Mean Squared Scaled Error (RMSSE) ซึ่งเป็นตัวแปรของ Mean Absolute Scaled Error (MASE) โดย Hyndman และ Koehler (2017) [2] มีการคำนวณการวัดสำหรับแต่ละอนุกรมดังนี้:

โดยที่ Y_t คือค่าพยากรณ์ ณ เวลา t และ h คือขอบเขตการคาดการณ์ ตัวหารของ RMSSE จะถูกคำนวณเฉพาะสำหรับช่วงเวลาที่ผลิตภัณฑ์ที่ตรวจสอบมีการขายอยู่เท่านั้นเช่นระยะเวลาถัดจากความต้องการแรกที่ไม่ใช่ศูนย์ที่สังเกตได้สำหรับซีรีส์ที่อยู่ระหว่างการประเมิน

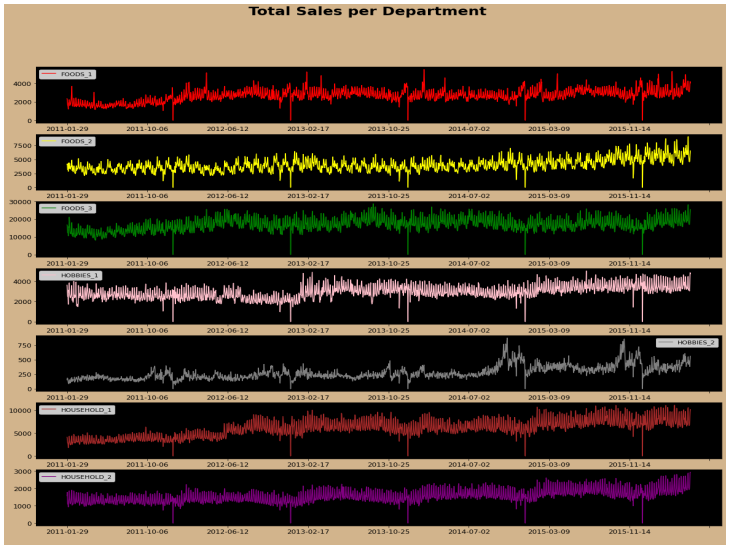
หลังจากการประมาณค่า RMSSE สำหรับอนุกรมเวลาทั้งหมด 42,840 รายการ วิธีการที่เข้าร่วมจะได้รับการจัดอันดับโดยใช้ Weighted RMSSE (WRMSSE) โดยที่ W_i คือน้ำหนักของซีรีส์ i_{th} คะแนน WRMSSE ที่ต่ำกว่าจะดีกว่า น้ำหนักของแต่ละซีรีส์จะคำนวณจากการสังเกต 28 ครั้งล่าสุดของชุดข้อมูล เช่น

ยอดขายสะสมเป็นดอลลาร์จริงที่แต่ละซีรี่ส์แสดงในช่วงเวลานั้น (ผลรวมของหน่วยที่ขายคูณด้วยราคาตามลำดับ)

Data Preprocessing : การประมวลผลข้อมูล

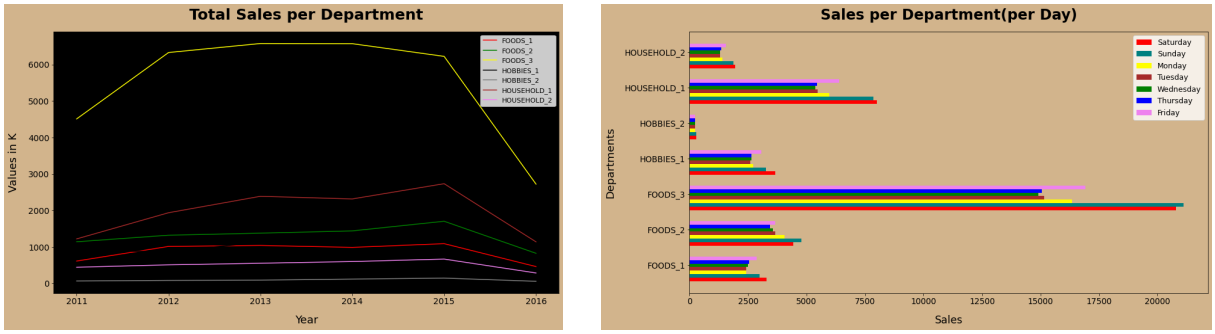
พิจารณาจากขนาดข้อมูลที่มีขนาดใหญ่ หากแบ่งชุดข้อมูลเพื่อฝึกโมเดลและคาดการณ์ข้อมูลเป็นชุด จะทำให้ลดการใช้หน่วยความจำและเพิ่มความเร็วเวลาการประมวลผล Model

นอกจากคุณลักษณะหมวดหมู่เหล่านั้นแล้ว คุณลักษณะตัวเลขในแบบจำลองการทำนาย ควรกำหนด "sell_price" ซึ่งจะส่งผลต่อการขายสินค้า sku นั้นๆ และข้อมูลการขายของสัปดาห์ที่แล้วหรือเดือนที่แล้วในวันเดียวกันในสัปดาห์ก็มีความสำคัญต่อการทำนายเช่นกัน



ภาพประกอบ 5 : ยอดขายรวมต่อแผนก (2011-2015)

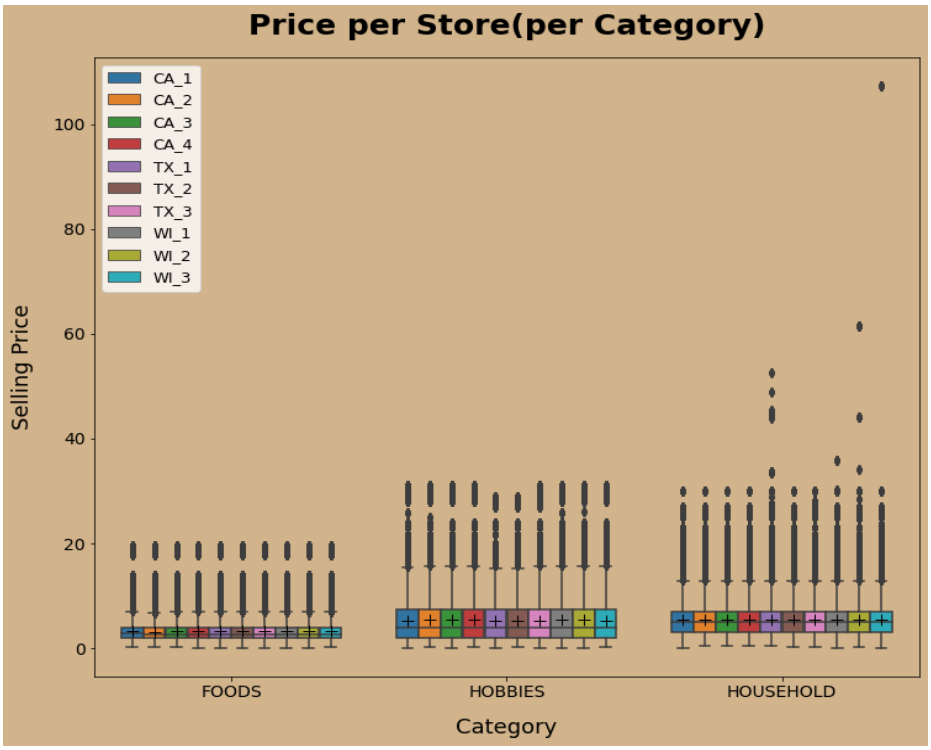
จากแผนกย่อยทั้งหมดข้างต้นเราสามารถสังเกตได้ว่าวันสุดท้ายของทุกปีมียอดขายเป็นศูนย์ ซึ่งอาจเป็นผลมาจากวันคริสต์มาสที่ร้านค้าไม่เปิดทำการ



ภาพประกอบ 6 : ยอดขายแยกตามแผนกแยกตามรายปีและรายวัน (2011-2015)

จากภาพประกอบที่ 6 สามารถสรุปได้ว่า

- ยอดขายของ FOODS_3 สูงที่สุดในบรรดาแผนกต่างๆ เราสามารถสังเกตเห็นความแตกต่างอย่างมากในยอดขายของ FOODS_3 และยอดขายแผนกอื่นๆ
- ยอดขายของ HOUSEHOLD_1, FOODS_2 ค่อยๆ เพิ่มขึ้นตั้งแต่ปี 2011 ถึง 2015 ในขณะที่ยอดขายของ HOBBIES_1 และ FOODS_1 ทับซ้อนกันและเกือบจะคงที่ตั้งแต่ปี 2011 ถึง 2015
- ยอดขายของ HOBBIES_2 ต่ำที่สุดในบรรดาแผนกต่างๆ และเกือบคงที่ตั้งแต่ปี 2011 ถึง 2016
- ยอดขายของแผนก FOODS_1, FOODS_2, FOODS_3, HOBBIES_1, HOUSEHOLD_1, HOUSEHOLD_2 หลังจากปี 2015 ตกลงทันทีเพราะเราไม่มีข้อมูลทั้งปี 2016
- ยอดขายวันเสาร์อาทิตย์สูงกว่าวันธรรมดา
- HOOBIES_2 แผนกมียอดขายน้อยที่สุด
- แผนก FOOD_3 มียอดขายสูงสุด

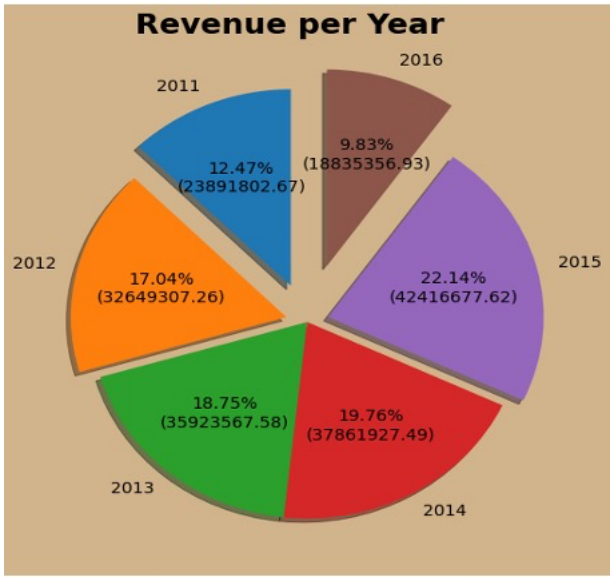


ภาพประกอบ 7 : ราคาต่อร้าน (ต่อประเภท)

จากภาพประกอบที่ 7 สามารถสรุปในส่วนของราคาขายได้ว่า

- อาหาร : ราคาขายเกือบจะเท่ากันสำหรับทุกร้านและต่ำสุดในบรรดาหมวดหมู่ทั้งหมด
- งานอดิเรก : ราคาขายเท่ากันทุกร้านและสูงกว่าหมวดอาหาร

- ครัวเรือน : ราคาขายสูงที่สุดในทุกประเภท และบางครั้งราคาขายก็สูงมาก (มากกว่า 100)



ภาพประกอบ 8 : รายได้ต่อปี

กิจกรรมเทศกาลหรือกีฬาสามารถมีอิทธิพลเชิงบวกอย่างมากต่อการขาย ตัวอย่างเช่น ผู้ซื้อมีแนวโน้มที่จะซื้อขนมหรืออาหารมากกว่าหนึ่งวันก่อนวันขอบคุณพระเจ้าหรือซูเปอร์โบว์ล ในทำนองเดียวกันสามารถคาดได้ว่าจะมียอดขายเพิ่มขึ้นในระหว่างวัน ด้วยโปรแกรม SNAP จะใส่ค่าคุณสมบัติเพิ่มเติม 5 รายการได้แก่ เหตุการณ์ 1, เหตุการณ์ 2, SNAP_CA, SNAP_WI, SNAP_TX วันที่ของโปรแกรม SNAP นั้นแตกต่างกันไปตามที่ตั้งของสาขาแต่ละแห่ง ดังนั้นจึงต้องการคุณสมบัติ 3 อย่างแยกกันสำหรับ SNAP และสามารถสรุปได้ว่า

รายได้ต่อปี

- รายได้มีทิศทางเพิ่มขึ้นดูได้จากรายได้ในระหว่างปี 2011 ถึง 2015
- ในปี 2015 ถึง 2016 พบว่ามีรายได้ลดลงแต่ไม่ใช่เพราะยอดขายจริงๆที่ลดลง แต่เนื่องจากว่าข้อมูลในปี 2016 บางส่วนไม่ครบถ้วน (ข้อมูลมถึงแค่เดือน พฤษภาคม 2016 เท่านั้น)

Methodology : วิธีการ / Approach : แนวทาง

Deep Learning : วิธีเลือกคุณสมบัติสำหรับ Model

- รวม item_id และ store_id เข้าด้วยกันเพื่อรับ ID เฉพาะสำหรับ SKU หลังจากลบ dept_id, cat_id, state_id แล้ว RMSE ปรับปรุงเล็กน้อย
- ใส่เฉพาะชื่อเหตุการณ์เป็นข้อมูลวันหยุด ลบประเภทเหตุการณ์ นอกจากนี้ยังปรับปรุงบางอย่างใน RMSE
- มีโปรโมชั่นหรือไม่ (snap_CA, snap_TX, snap_WI)
- ข้อมูลวันที่ (วัน สัปดาห์ เดือน ปี)
- ราคาขาย

KNN Regression

อันดับแรกใช้ KNN Regression และให้พารามิเตอร์ต่างๆ พยายามหาพารามิเตอร์ที่ดีที่สุด ก่อนที่จะนำไปใช้ในการฝึกโมเดล คุณลักษณะประเภทของข้อมูลจำเป็นต้องทำการประมวลผลล่วงหน้าเพื่อสร้างค่าจำลอง ใช้ไลบรารี OneHotEncoder ของ sk-learn

```
%%time
# create the training dataset and validation dataset
train_data, validate_data, evaluate_data = create_dataset(df_sales, df_calendar, df_price, id_list, batch_size, batch_start, start_day)

X = train_data[cat_features + num_features]
y = train_data['sales']
#X_validate = validate_data[cat_features + num_features]
#X_evaluate = evaluate_data[cat_features + num_features]

# feed the training data (X, y) and train the model
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)

# create dummy value for the category features via OneHotEncoder function
cat_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='constant', fill_value=0)),
    ('onehot', preprocessing.OneHotEncoder(handle_unknown='ignore'))])

preprocessor = ColumnTransformer(
    transformers=[('cat', cat_transformer, cat_features)])

clf = Pipeline(steps=[('preprocessor', preprocessor),
                      ('KNN', KNeighborsRegressor())])

k_range = [5, 50, 100, 200]
weight_options = ['uniform', 'distance']
metric_options = ['euclidean', 'minkowski', 'mahalanobis']

param_grid = {'KNN__n_neighbors':k_range, 'KNN__weights':weight_options, 'KNN__metric':metric_options}
grid = GridSearchCV(clf, param_grid, n_jobs=-1)

grid.fit(X_train, y_train)

print("Best Parameter: ", grid.best_params_)
print("KNN Model with best parameter training RMSE: %.4f" %(sqrt(mean_squared_error(y_train, grid.best_estimator_.predict(X_train)))))
print("KNN Model with best parameter testing RMSE: %.4f" %(sqrt(mean_squared_error(y_test, grid.best_estimator_.predict(X_test)))))

Best Parameter:  {'KNN__metric': 'euclidean', 'KNN__n_neighbors': 100, 'KNN__weights': 'uniform'}
KNN Model with best parameter training RMSE: 3.6884
KNN Model with best parameter testing RMSE: 3.8787
CPU times: user 52.3 s, sys: 1.71 s, total: 54.1 s
Wall time: 2min 1s
```

ภาพประกอบ 9 การ run KNN Regression

จากภาพประกอบที่ 9 ใช้พารามิเตอร์ที่ดีที่สุดที่ประเมินโดย GridSearchCV ผลการทดสอบ RMSE = 3.8787

Random Forest

```
%%time
# dataframe to store the training and test RMSE for each model
train_RMSE = pd.DataFrame()
test_RMSE = pd.DataFrame()

# Fit estimators
ESTIMATORS = {
    "Random Forest": RandomForestRegressor(n_estimators=100, max_features=32, random_state=0),
    "Linear regression": LinearRegression(),
    "Logistics regression": LogisticRegression(solver='saga'),
    "SVM": SVR(),}

for name, estimator in ESTIMATORS.items():

    df, tr_RMSE, te_RMSE = create_model(name, estimator, X, y)
    train_RMSE = train_RMSE.append(tr_RMSE, ignore_index=True)
    test_RMSE = test_RMSE.append(te_RMSE, ignore_index=True)

test_RMSE
```

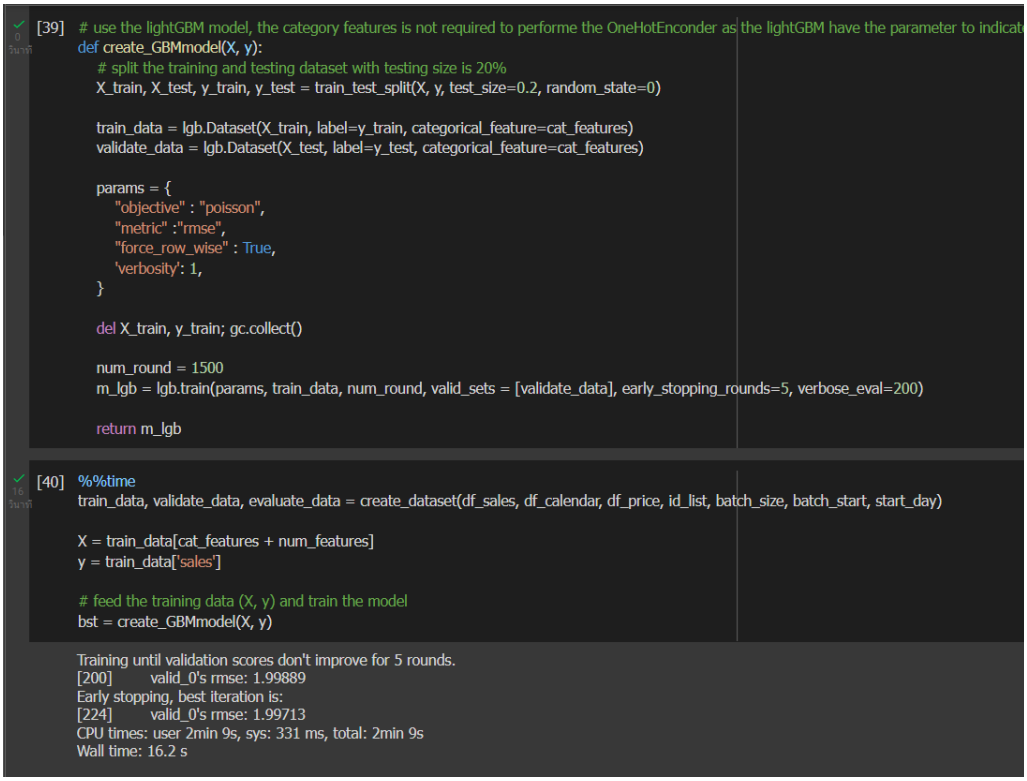
```
model: Random Forest, training RMSE: 1.6331
model: Random Forest, test RMSE: 3.4751

model: Linear regression, training RMSE: 3.1245
model: Linear regression, test RMSE: 3.2572

model: Logistics regression, training RMSE: 3.9617
model: Logistics regression, test RMSE: 4.0736

model: SVM, training RMSE: 3.2699
model: SVM, test RMSE: 3.4506

CPU times: user 5min 32s, sys: 2.62 s, total: 5min 35s
Wall time: 5min 31s
```

	model	RMSE
0	Random Forest	3.475070
1	Linear regression	3.257237
2	Logistics regression	4.073648
3	SVM	3.450590

ภาพประกอบ 10 : การทดสอบด้วย Random Forest model

จากภาพประกอบที่ 10 Random Forest มีผลการ Train ที่ดีที่สุดแต่มีปัญหาเรื่องเวลาในการประมวลผลที่นานมากเมื่อเทียบกับ 4 โมเดล และ Linear Regression มีค่า RMS ที่ดีกว่า

LightGBM Model (ด้านล่าง) เป็นอีก Model ที่จะใช้นำมาเปรียบเทียบ [4] LightGBM เป็นเฟรมเวิร์คส่งเสริมการไล่ระดับสีที่ใช้อัลกอริทึมการเรียนรู้ตามต้นไม้ LightGBM สามารถจัดการข้อมูลขนาดใหญ่และใช้หน่วยความจำน้อยกว่าในการเรียกใช้ และวิธีการนี้เน้นความแม่นยำของผลลัพธ์ แต่ LightGBM นั้นไวต่อการใช้ข้อมูลมาก และสามารถใช้งานข้อมูลขนาดเล็กได้อย่างดี ซึ่งจะเหมาะกับข้อมูลที่มีมากกว่า 10,000 แถว (rows) เท่านั้น เนื่องจากชุดข้อมูลการขายมีขนาดใหญ่มาก การใช้ LightGBM เพื่อคาดการณ์ผลลัพธ์จึงเหมาะสม



```
[39]  # use the lightGBM model, the category features is not required to performe the OneHotEnconder as the lightGBM have the parameter to indicat
      def create_GBMmodel(X, y):
          # split the training and testing dataset with testing size is 20%
          X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)

          train_data = lgb.Dataset(X_train, label=y_train, categorical_feature=cat_features)
          validate_data = lgb.Dataset(X_test, label=y_test, categorical_feature=cat_features)

          params = {
              "objective" : "poisson",
              "metric" :"rmse",
              "force_row_wise" : True,
              'verbosity': 1,
          }

          del X_train, y_train; gc.collect()

          num_round = 1500
          m_lgb = lgb.train(params, train_data, num_round, valid_sets = [validate_data], early_stopping_rounds=5, verbose_eval=200)

          return m_lgb

[40]  %%time
      train_data, validate_data, evaluate_data = create_dataset(df_sales, df_calendar, df_price, id_list, batch_size, batch_start, start_day)

      X = train_data[cat_features + num_features]
      y = train_data['sales']

      # feed the training data (X, y) and train the model
      bst = create_GBMmodel(X, y)

      Training until validation scores don't improve for 5 rounds.
      [200]   valid_0's rmse: 1.99889
      Early stopping, best iteration is:
      [224]   valid_0's rmse: 1.99713
      CPU times: user 2min 9s, sys: 331 ms, total: 2min 9s
      Wall time: 16.2 s
```

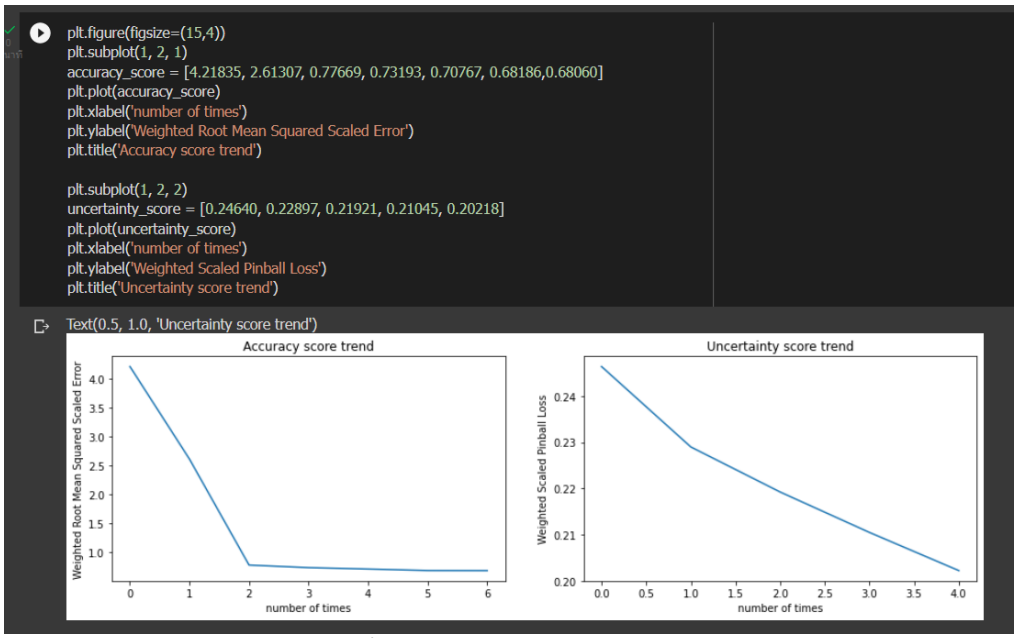
ภาพประกอบ 11 : การทดสอบด้วย LightGBM

จากผลลัพธ์ตาม ภาพประกอบ 11 จะเห็นว่าโมเดลได้รับผลลัพธ์ RMSE ที่ดีที่ 1.997 และใช้เวลาสั้นมากในการฝึก (train) ข้อมูล 1,000 รายการ ดังนั้นจึงนำ LightGBM มาใช้เพื่อนำเสนอผลงานสุดท้ายสำหรับทุกSKU

สรุปผลการวิจัย

จากการทดสอบด้วย Model ต่างๆ เพื่อพยากรณ์ยอดการขายสินค้าของ Walmart ใน 28 วันข้างหน้าด้วย Model ต่างๆที่แตกต่างกันพบว่าแต่ละ Model ให้ผลดังต่อไปนี้ KNN Regression RMSE = 3.8787, Random Forest RMSE = 3.4750.70, Linear Regression RMSE = 3.257237, Logistics Regression RMSE =4.073648, SVM RMSE =3.450590 และ LightGBM RMSE = 1.997 ซึ่งสามารถสรุปได้ว่า Model ที่ให้ผลดีที่สุดและเวลาที่ Training และ processing น้อยที่สุดจะเป็น LightGBM ดังนั้นแล้วการพยากรณ์-คาดคะเนของการขายสินค้าล่วงหน้าใน 28 วันของ Walmart ในทุกๆ SKU แยกตามหมวด หรือตามสาขาควรที่จะใช้ LightGBM ในการพยากรณ์ โดยที่ในแต่ละสาขาจะมีความต้องการซื้อสินค้าที่แตกต่างกัน ดูได้จากหมวดตัวอย่างสินค้าในกลุ่ม Hobbies, Food, Houshold มีความต้องการในแต่ละสาขา และในแต่ละรัฐไม่เท่ากันด้วย

อภิปรายผล



ภาพประกอบที่ 12 : การประมวลผลโดยใช้ LightGBM

จากภาพประกอบที่ 12 แสดงให้เห็นว่าวิธีการคาดการณ์ตามเทคนิคการเรียนรู้เชิงลึกสามารถปรับปรุงความแม่นยำในการพยากรณ์ได้เมื่อเทียบกับวิธีการพยากรณ์ในหลากหลายวิธี ในการหาข้อมูลความไม่แน่นอน ตัวอย่างเช่น การใช้ K-fold เพื่อดำเนินการฝึกโมเดลและการทำนาย จะมีชุดผลการทำนายที่แตกต่างกันไป

ชุดข้อมูล โมเดลที่นำเสนออนุกรมเวลาวิเคราะห์ข้อมูลที่มีความแตกต่างอย่างมากผ่านการปรับขนาด และการสุ่มตัวอย่างตามความเร็ว และสามารถเรียนรู้รูปแบบที่ซับซ้อน เช่น เทศกาลและการเติบโตที่ไม่แน่นอนเมื่อเวลาผ่านไป วิธีการนี้ใช้ได้กับการปรับไฮเปอร์พารามิเตอร์เพียงเล็กน้อยหรือไม่มีเลยในชุดข้อมูลที่หลากหลายและใช้ได้ดีกับชุดข้อมูลขนาดกลางที่มีอนุกรมเวลาเพียงไม่กี่ร้อย

ข้อเสนอแนะ

สามารถนำไปประยุกต์ใช้กับเทคนิคอื่นๆเช่น Neural Network เนื่องจากความยืดหยุ่นที่คล้ายกัน หรือใช้ Stacked ensemble model ที่ใช้ในการคาดคะเนจากหลายๆ โมเดล ซึ่งจะช่วยให้การพยากรณ์ให้ดูได้หลายมิติมากขึ้น รวมทั้งนำไปประยุกต์ใช้เพื่อคาดการณ์ในการวางแผนเพื่อการประเมินการขยายสาขา

เอกสารอ้างอิง

[1] University of Nicosia. (2022, Dec 10). *M5 Forecasting – Accuracy* [Online] Available: https://www.kaggle.com/competitions/m5-forecasting-accuracy/overview

[2] Muzaffer Uysal, Zvi Schwartz, Ercan Sirakaya-Turk. "Evaluating Forecasting Performance: Accuracy Mesure"**,** *Management science in hospitality and tourism.* pp289, 2017

[3] AllENL JW. (2022, Dec 10). *158755 M5 Forecasting* [Online] Available: https://www.kaggle.com/code/allenljw/158755-m5-forecasting/notebook#M5-Forecasting---Uncertainty

[4] SHAITENDER SINGH. (2022, Dec 10). *Light GBM - m5_forecasting_accuracy_forcasting* [Online] Available https://www.kaggle.com/code/shaitender/light-gbm-m5-forecasting-accuracy-forcasting