TME-6015: AI/ML Workflow Design

Store Sales Forecasting: Time Series Analysis

Team – 9

Sarojini Sharon Robert Kennedy (3724629)
Sreemathy Maheswaramoorthy (3736080)
Sandya Vaigundam Sadasivam (3740860)

1. Problem Statement:

The goal of this project is to build a reliable Machine Learning (ML) or Artificial Intelligence (AI) model to predict store sales using historical data. The biggest challenge is predicting future sales trends, which is essential for successful inventory management, resource allocation and strategic decision making.

a. The value of solving the problem:

The problem is to forecast store sales over time based on various features such as store number, product family, and promotion status. Accurate sales forecasting is crucial for inventory management, resource planning, and overall business strategy. By solving this problem, retailers can optimize their stock levels, reduce waste, and improve profitability.

b. Who is the end user and how they can be impacted:

The end-users are likely to be retail managers, inventory planners, and business analysts. Accurate sales forecasts empower them to make informed decisions about inventory stocking, promotional strategies, and resource allocation. This, in turn, can lead to cost savings, increased revenue, and improved customer satisfaction.

c. Constraints and requirements:

Constraints include the reliance on historical sales data and the potential impact of external factors, such as economic conditions, holidays, and oil prices. These constraints emphasize the need for robust models capable of adapting to varying circumstances. On the other hand, key requirements involve regular updates of data to ensure model relevance and efficiency in handling dynamic factors like promotions and holidays. Additionally, seamless integration with business operations and data systems is essential to facilitate the timely implementation of model insights into inventory management and decision-making processes. Striking a balance between addressing these constraints and meeting these requirements is crucial for the successful deployment and continual effectiveness of the forecasting models in a real-world retail environment.

d. What is the type of ML problem:

The ML problem here is a multivariate time series forecasting regression task. This problem involves addressing challenges in capturing temporal dependencies and external factors, such as promotions, holidays, and economic conditions, which significantly influence sales patterns. Effective feature engineering is vital to represent these features accurately. The goal is to minimize the difference between predicted and actual sales values using regression models. The success of the models is evaluated using Root Mean Squared Error metric.

e. Success metrics:

The success metric used for evaluating the performance of the models is the Root Mean Squared Error (RMSE). RMSE is a commonly used metric to measure the average magnitude of the errors between predicted and actual values. It gives more weight to larger errors, making it sensitive to outliers.

1. ***Training the Models:***
   For each model (CNN, LSTM, CNN-LSTM), the training process involves minimizing the mean squared error (MSE) loss. This is done through iterations (epochs) of adjusting the model's parameters to minimize the difference between predicted and actual sales values.

2. ***Evaluation on Training and Validation Sets:***
   After training, the models are evaluated using the RMSE metric on both the training and validation sets. This evaluation helps assess how well the model generalizes to both seen (training) and unseen (validation) data.

3. ***Interpreting the RMSE Values:***
   The lower the RMSE, the better the model performance. It indicates a smaller average deviation of the predicted values from the actual values.

   In the code, the RMSE is calculated for each model on both the training and validation sets after training is complete.

4. ***Results Interpretation:***
   The reported RMSE values in the code provide a quantitative measure of how well the models have learned to predict store sales. Lower RMSE values indicate better predictive performance.

2. Solution Design

a. Literature review

i. Has this problem been encountered before?

The systematic literature review indicates a rich landscape of research in the domain of retail sales forecasting using deep learning models. Various studies have explored the application of Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), and hybrid models like CNN-LSTM in predicting store sales. These models are recognized for their ability to capture temporal dependencies, handle non-linear relationships, and outperform traditional methods. Evaluation metrics such as Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) are commonly employed to assess model performance.

ii. How was it solved? What is the state-of-the-art technique?

In a comprehensive exploration of Retail Sales Forecasting Using Deep Learning [1], the systematic literature review delves into the application of deep learning models, underscoring their pivotal role in enhancing supply chain efficiency and ensuring optimal stock availability. The study categorizes diverse deep learning frameworks such as ANN, RNN, CNN, LSTM, and MLP, with LSTM emerging as the prevailing architecture. While extolling the benefits of deep learning models, particularly their capacity to navigate non-linear relationships, the review candidly acknowledges challenges, including model

complexity and interpretability limitations. Serving as a valuable reference, this review provides a nuanced understanding of the deep learning landscape in retail sales forecasting [1] . "Retail Demand Forecasting using CNN-LSTM Model" introduces a novel deep learning model, specifically a CNN-LSTM model with the Swish Activation Function, designed for predicting retail demand. Amidst challenges faced by retailers, notably during the COVID-19 pandemic, the authors compare their model with alternatives, showcasing its efficacy through lower Root Mean-Squared Error (RMSE). The methodology involves transforming time series algorithms, implementing diverse models, and scrutinizing activation functions. Empirical results accentuate the superior performance of the CNN-LSTM model with the Swish Activation Function [2]. Addressing inventory forecasting challenges for perishable items, "Evolving Deep CNN-LSTMs for Inventory Time Series Prediction" proposes a hybrid CNN and LSTM model, emphasizing automated optimal network architecture design through meta-heuristics. Comparative assessments with traditional SARIMA models underscore the superior accuracy of evolved CNN-LSTM models in predicting hourly sales for perishable items. These findings suggest the promise of evolutionary approaches in autonomously identifying effective neural network architectures for time series forecasting [3]. In "A hybrid deep learning framework with CNN and Bi-directional LSTM for store item demand forecasting," the authors advocate for accurate predictions in the retail sector, presenting a superior hybrid CNN-BiLSTM model that outperforms traditional methods. Leveraging CNN for feature extraction and BiLSTM for capturing temporal features, the model, optimized by the Lazy Adam optimizer, contributes to the evolving literature on CNN-LSTM architectures in demand forecasting [4]. "Future Sales Prediction For Indian Products Using Convolutional Neural Network-Long Short Term Memory" addresses the imperative of accurate sales forecasts for effective capital distribution and strategic planning in Indian stores. The proposed CNN-LSTM model, applied to real-time data from Big Mart stores, demonstrates remarkable effectiveness, achieving a maximum precision of 97%. The study's detailed methodology and experimental results establish the CNN-LSTM model's superiority in retail sales predictions [5]. In the realm of "A Multi-Parameter Forecasting for Stock Time Series Data Using LSTM and Deep Learning Model," a pioneering effort unfolds as the paper introduces a hybrid CNN-LSTM-RNN architecture designed for the simultaneous prediction of close and high prices within stock time series data. Undertaking a meticulous comparative analysis across diverse models, the study robustly establishes the proposed hybrid model's superiority, effectively addressing the inherent limitations of single parameter stock forecasting. By seamlessly integrating convolutional neural networks (CNN), long short-term memory networks (LSTM), and recurrent neural networks (RNN), the model not only outperforms its counterparts but also furnishes invaluable insights conducive to elevated decision-making within the complex dynamics of the stock market. This groundbreaking contribution extends beyond the conventional by introducing a sophisticated methodology for multi-parameter stock price forecasting, thereby endowing investors with enhanced tools for strategic decision-making, marking a significant advancement in the field[6].

iii. What were the limitations to that solution? (Gap in solution)

In the exploration of retail sales forecasting, leveraging deep learning (DL) models, particularly Long Short-Term Memory (LSTM), has revealed both advantages and challenges. While LSTM demonstrates prowess in handling non-linear relationships and outperforming other methods, its complexity introduces interpretability challenges, rendering these models as "black boxes." This complexity poses difficulties for end-users, such as retail managers and business analysts, in understanding the rationale behind predictions. Additionally, the systematic literature review acknowledges the potential

biases inherent in DL models, raising concerns about the fairness and transparency of the forecasting process [1]. Similarly, in retail demand forecasting during dynamic scenarios like the COVID-19 pandemic, the integration of Convolutional Neural Network (CNN) and LSTM effectively lowers Root Mean-Squared Error (RMSE). However, challenges persist in predicting optimal stock levels during unpredictable events, emphasizing the ongoing difficulty in forecasting demand, especially in uncertain periods. The proposed model marks progress but acknowledges the need for further refinement to handle extreme variations [2]. In the context of evolving deep CNN-LSTMs for inventory time series prediction, despite the success in automating architecture design, the intricate nature of predicting highly perishable food items with variable demand remains a complex challenge. This complexity necessitates further exploration, including the incorporation of external datasets for multivariate time series prediction and leveraging high-performance computing for more extensive architectural possibilities [3]. Furthermore, the hybrid deep learning framework with CNN and Bi-directional LSTM for store item demand forecasting showcases superiority over traditional methods but confronts the inherent challenge of balancing model complexity and interpretability. The study recognizes that effective use of the hybrid model depends on the careful integration of CNN for feature extraction and BiLSTM for capturing temporal dependencies, highlighting the ongoing trade-off between complexity and interpretability in real-world retail environments [4]. In the realm of future sales prediction for Indian products using CNN-LSTM, while the proposed hybrid model demonstrates high precision, challenges arise from the reliance on historical data and assumptions of continuity in past sales patterns. The study acknowledges the potential impact of unexpected events or shifts in consumer behavior, emphasizing the need for continuous adaptation and improvement to enhance predictive accuracy [5]. Finally, the multi-parameter forecasting for stock time series data using LSTM and deep learning introduces a novel approach but recognizes that the proposed CNN-LSTM-RNN model is not universally applicable. The study highlights challenges related to noise, non-linearity, and volatility in stock data, acknowledging the influence of specific dataset characteristics on the model's performance and emphasizing the importance of considering risk factors in future work [6].

iv. What are you proposing that is “novel”?

Overall, our proposed project's novelty lies in its holistic approach to store sales forecasting, combining a systematic literature review, multivariate time series analysis, and the use of various deep learning models. This comprehensive strategy distinguishes it from the more specialized focuses of the referenced research papers. In comparison to the mentioned research papers, our proposed approach brings several novel contributions to the realm of retail sales forecasting using deep learning. While these papers focus on leveraging various deep learning architectures (such as CNN-LSTM, LSTM, CNN, RNN) for sales prediction, our approach stands out in several aspects. Our approach emphasizes multivariate time series forecasting, incorporating various factors like store number, product family, and promotion status, enabling a more comprehensive understanding of sales dynamics. We also highlight the challenge of dealing with external factors, such as economic conditions and holidays, proposing the need for robust models capable of adapting to such variations. The emphasis on Root Mean Squared Error (RMSE) as the success metric, along with its interpretation, sets a clear benchmark for evaluating the proposed models' effectiveness. We delve into the training process, discussing how models are trained to minimize mean squared error (MSE) loss, and emphasize evaluating models on both training and validation sets post-training to assess generalization capabilities. Additionally, our stress on seamlessly integrating forecasting models with business operations and data systems underscores the practicality and real-world implementation of

these models in decision-making processes. While the papers primarily concentrate on comparing and evaluating deep learning models for sales forecasting, our proposal presents a comprehensive approach encompassing feature engineering, robustness to external factors, clear success metrics, training details, and integration strategies. This broader scope aims to provide a more holistic solution catering to the complexities of retail sales forecasting in real-world scenarios.

v. References of previous related work:

[1] Eglite, Linda, and Ilze Birzniece. 2022. "Retail Sales Forecasting Using Deep Learning: Systematic Literature Review." Complex Systems Informatics and Modeling Quarterly 53–62. doi: 10.7250/csimq.2022-30.03.

[2] Anon. n.d. "Retail Demand Forecasting Using CNN-LSTM Model | IEEE Conference Publication | IEEE Xplore." Retrieved November 29, 2023 (https://ieeexplore.ieee.org/abstract/document/9752283).

[3] Xue, Ning, Isaac Triguero, Grazziela P. Figueredo, and Dario Landa-Silva. 2019. "Evolving Deep CNN-LSTMs for Inventory Time Series Prediction." Pp. 1517–24 in 2019 IEEE Congress on Evolutionary Computation (CEC).

[4] Joseph, Reuben Varghese, Anshuman Mohanty, Soumyae Tyagi, Shruti Mishra, Sandeep Kumar Satapathy, and Sachi Nandan Mohanty. 2022. "A Hybrid Deep Learning Framework with CNN and Bi-Directional LSTM for Store Item Demand Forecasting." Computers and Electrical Engineering 103:108358. doi: 10.1016/j.compeleceng.2022.108358.

[5] Kaunchi, Pooja, Tushar Jadhav, Yogesh Dandawate, and Pankaj Marathe. 2021. "Future Sales Prediction For Indian Products Using Convolutional Neural Network-Long Short Term Memory." Pp. 1–5 in 2021 2nd Global Conference for Advancement in Technology (GCAT).

[6] Anon. n.d. "Mathematics | Free Full-Text | A Multi Parameter Forecasting for Stock Time Series Data Using LSTM and Deep Learning Model." Retrieved November 30, 2023 (https://www.mdpi.com/2227-7390/11/3/590).

b. From Perspective of ML Workflow

i. Dataset Selection:

The dataset is chosen from Kaggle, Dataset link: Kaggle - Store Sales Time Series Forecasting. It contains train.csv, transactions.csv, stores.csv, holidays.csv, oil.csv

The attribute in the dataset aligns with the outlined project requirements. These datasets collectively provide information on store sales, transactions, store metadata, holiday events, and daily oil prices. This selection ensures a comprehensive representation of both temporal and external factors influencing store sales which would be appropriate to forecast store sales using time series analysis.

ii. Data pre-processing:

The dataset preprocessing steps performed in the provided code can be broken down as follows:

1. ***Date Conversion and Label Encoding:***
   The 'date' column in both the training (train) and test datasets is converted to a datetime format for ease of handling temporal data. Subsequently, Label Encoding is applied to categorical variables, specifically 'family' and 'store_nbr,' using the `LabelEncoder` from scikit-learn. This transforms categorical labels into numerical representations.

2. ***Dropped Unnecessary Columns:***
   The columns 'family' and 'store_nbr' are dropped from the training dataset (`train`) as they have been encoded and are no longer needed. Columns irrelevant for modeling, such as 'family(t)' and 'store_nbr(t),' are dropped from the 'series' DataFrame. This reduces redundancy in the dataset.

3. ***Filtering by Date:***
   The dataset is filtered to include only records with a date on or after '2017-01-01.' This step focuses the analysis on a specific time period, potentially aligning with a more recent period of interest.

4. ***Data Aggregation:***
   The training dataset (`train`) is then rearranged and aggregated based on the columns 'family,' 'store_nbr,' 'date,' and 'onpromotion.' The sales values are aggregated using the mean function. This aggregation is likely performed to obtain a more manageable and informative representation of the data for time series analysis.

5. ***Series Transformation to Supervised Learning:***
   The function `series_to_supervised` is defined to transform the time series data into a supervised learning format. It creates lag features by shifting the data for a specified window size and generates target values for a specified lag. The resulting DataFrame, named 'series,' now includes lagged and target columns.

6. ***Data Filtering based on Lagged Columns:***
   Rows in the 'series' DataFrame are filtered to retain only those where the 'store_nbr(t)' column matches the lagged 'store_nbr(t-%d)' column and similarly for 'family' and 'onpromotion' columns. This ensures consistency in the lagged features.

iii. Model selection:

Three distinct models were employed: Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM) Neural Network, and a hybrid CNN-LSTM model. Each model was chosen based on its unique architectural strengths and suitability for addressing specific characteristics of the time series data.

1. ***Convolutional Neural Network (CNN):***
   The CNN model was effective in capturing spatial dependencies within the input data. The model utilizes a convolutional layer with a kernel size of 2, a stride of 1, and Rectified Linear Unit (ReLU) activation. It is followed by a max-pooling layer

and dense layers. This model was choosen as it is suitable for capturing local patterns in the temporal data.

2. ***Long Short-Term Memory (LSTM) Neural Network:***

LSTM networks are specialized for handling sequential and time-dependent patterns. The LSTM model in this code is designed with one LSTM layer to process input sub-sequences, followed by dense layers for interpreting the overall input sequence. LSTMs are particularly adept at capturing long-term dependencies in time series data. Her, the efficient Adam optimizer and Mean Squared Error (MSE) loss function are used for training the model.

3. ***Hybrid CNN-LSTM Model:***

The hybrid model combines the strengths of both CNN and LSTM architectures. Each sample is divided into subsequences, where the CNN processes each subsequence, and the LSTM interprets the results before making predictions. This model is known for its efficiency in prediction, precision, and stability. It utilizes the Adam optimizer and MSE loss function.

The choice of model depends on the characteristics of the time series data and the specific patterns one aims to capture. CNNs are effective for local patterns, LSTMs excel at capturing sequential dependencies, and the hybrid model leverages the strengths of both. The selection process involved understanding the nature of the data and our problem statement, experimenting with different architectures, and evaluating their performance on validation sets to determine the most suitable model for forecasting.

iv. Training/ Fine-tuning:

After the dataset was preprocessed and split into training and validation sets, the models underwent training and fine-tuning. Here's a brief overview of the process:

<u>Convolutional Neural Network (CNN):</u>

Input Configuration: The CNN model was configured with the number of input time steps set to 1 and the number of features as 2, defined by the input_shape argument.

Architecture: The model consisted of a convolutional layer with a kernel size of 2, a ReLU activation, and a max-pooling layer. Dense layers followed the convolutional layer.

Training Parameters: The model used the Adam optimizer for stochastic gradient descent and optimized the Mean Squared Error (MSE) loss function.

Fine-Tuning: The model was trained on the training set with 76 epochs, and the training process was monitored using validation data. Dropout with a rate of 0.2 was employed to prevent overfitting.

<u>Long Short-Term Memory (LSTM):</u>

Input Configuration: Like the CNN model, the LSTM model had the number of input time steps set to 1 and the number of features as 2.

Architecture: The LSTM model consisted of one LSTM layer, followed by dense layers to interpret the input sequence.

Training Parameters: The model used the Adam optimizer and optimized the Mean Squared Error (MSE) loss function.

Fine-Tuning: The model underwent training for 100 epochs, and the training process was monitored using validation data.

Hybrid CNN-LSTM Model:

Input Configuration: The hybrid model divided each sample into subsequences, with the CNN processing each subsequence and the LSTM interpreting the results before making predictions.

Architecture: The model utilized a combination of CNN and LSTM layers, and the TimeDistributed layer applied CNN to every temporal slice of the input.

Training Parameters: The model used the Adam optimizer and optimized the Mean Squared Error (MSE) loss function.

Fine-Tuning: The hybrid model was trained for 100 epochs, and the training process was monitored using validation data.

v. Hyperparameter tuning strategy:

After thorough testing and experimentation, the optimal hyperparameters were selected for the neural network training process. A maximum of 100 epochs was set, incorporating early stopping with a patience of 25. The learning rate was fine-tuned, with 0.001 proving to be the most effective, and a batch size of 64 was chosen over 128 based on superior performance. The careful selection of these parameters reflects a thoughtful and systematic approach to achieving optimal model performance.

Additionally, the hyperparameter tuning strategy likely involved experimenting with different configurations, observing the model's performance on the validation set, and selecting the combination that yielded the best forecasting results For the CNN model, a single convolutional layer with a kernel size of 2, stride of 1, and Rectified Linear Unit (ReLU) activation was used. A max-pooling layer was applied, and dense layers followed. The model was compiled using the Adam optimizer and mean squared error (MSE) loss function. A dropout rate of 0.2 was implemented to prevent overfitting.

Similarly, for the LSTM model, one LSTM layer was used with efficient Adam optimization and MSE loss. The model was trained for hundred epochs, and hyperparameters were fine-tuned to optimize the trade-off between bias and variance.

In the hybrid CNN-LSTM model, the architecture combined CNN and LSTM for more efficient predictions. This involved dividing samples into subsequences, where CNN processed each subsequence, and LSTM interpreted the results before making predictions. The model was trained using Adam optimization and MSE loss.

vi. Evaluation metrics:

The evaluation metrics employed for this approach is Root Mean Square Error (RMSE), a widely used measure for assessing the accuracy of regression models. RMSE calculates the square root of the average squared differences between predicted and actual values, providing a comprehensive understanding of prediction errors. These metrics offer a quantitative assessment.

Of the models' ability to accurately predict sales data, with lower RMSE values indicating better performance. The consistent improvement in RMSE values from individual models to the hybrid model reinforces the effectiveness of the latter in capturing intricate temporal patterns and enhancing predictive accuracy.

3. Implementation

a. Code

CNN

```
##CNN MODEL

# Set parameters
epochs = 100
batch = 64
lr = 0.001
adam = optimizers.Adam(lr)

X_train_series = X_train.values.reshape((X_train.shape[0],
X_train.shape[1], 1))
X_valid_series = X_valid.values.reshape((X_valid.shape[0],
X_valid.shape[1], 1))
print('Train set shape', X_train_series.shape)
print('Validation set shape', X_valid_series.shape)

# model
cnn = Sequential()
cnn.add(Conv1D(filters=128, kernel_size=2, activation='relu',
input_shape=(X_train_series.shape[1], X_train_series.shape[2])))
cnn.add(MaxPooling1D(pool_size=2))
cnn.add(Flatten())
cnn.add(Dense (128, activation='relu'))
cnn.add(Dropout(0.2))
cnn.add(Dense (128, activation='relu'))
cnn.add(Dropout(0.2))
cnn.add(Dense (34, activation='relu'))
cnn.add(Dropout(0.2))
cnn.add(Dense(1))
cnn.compile(loss='mse', optimizer=adam)
cnn.summary()

callback = tf.keras.callbacks.EarlyStopping(monitor='val_loss',
patience=25)
```

```
cnn_history = cnn.fit(X_train_series, Y_train,
validation_data=(X_valid_series, Y_valid),
callbacks=[callback],epochs=epochs, verbose=2)
```

LSTM

```
lstm = Sequential()
lstm.add(LSTM(256, activation='relu',
input_shape=(X_train_series.shape[1], X_train_series.shape[2])))
lstm.add(Dense(128))
lstm.add(Dropout(0.2))
```

b. Create repository:

A repository was created in github to update the code: https://github.com/sreemathymahes/TME6015-Team-9

c. Generate results/ plots:

We Implemented few data visualization to understand the distribution of sales among different product family and store sales.

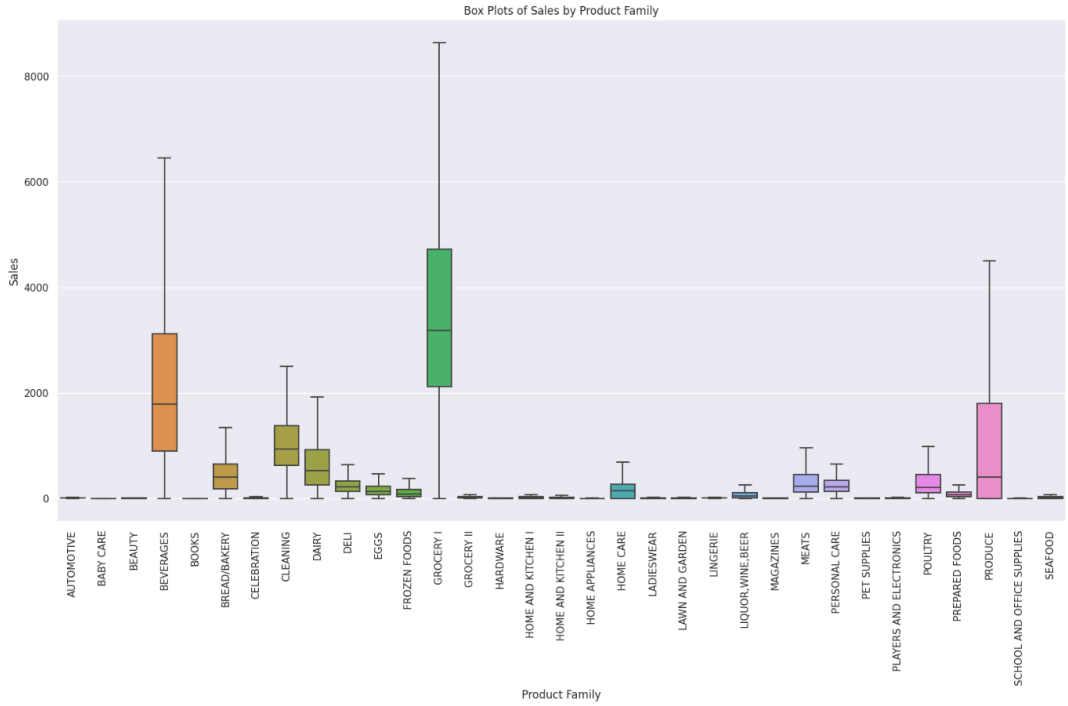

Figure 1: Box plot of sales for each Product Family from the dataset.

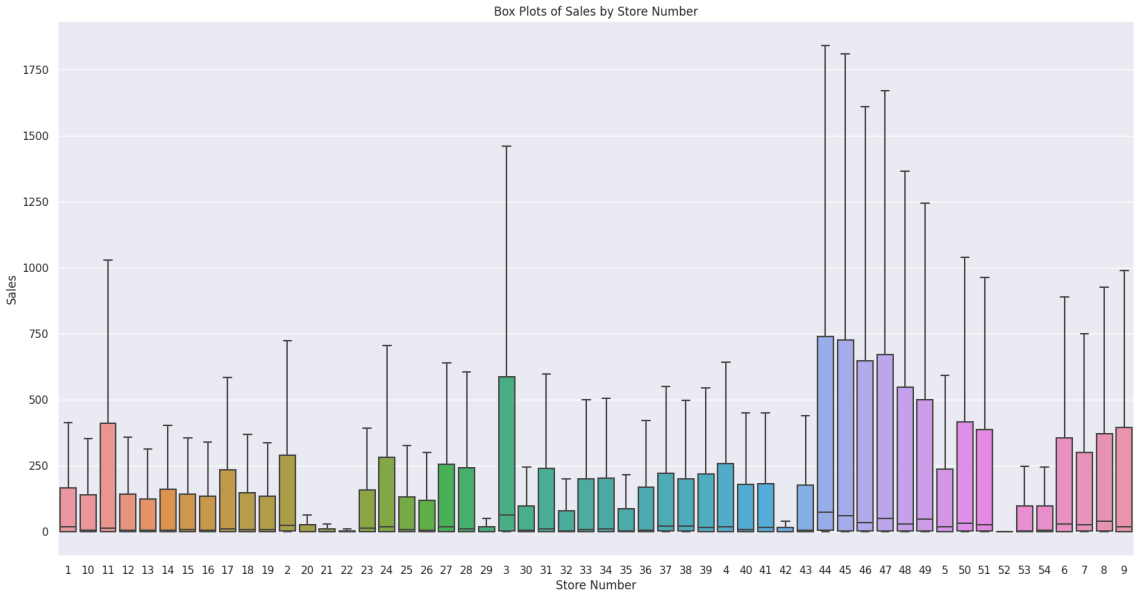

Figure 2: Box plot of Sales with respect to Stores from the dataset.

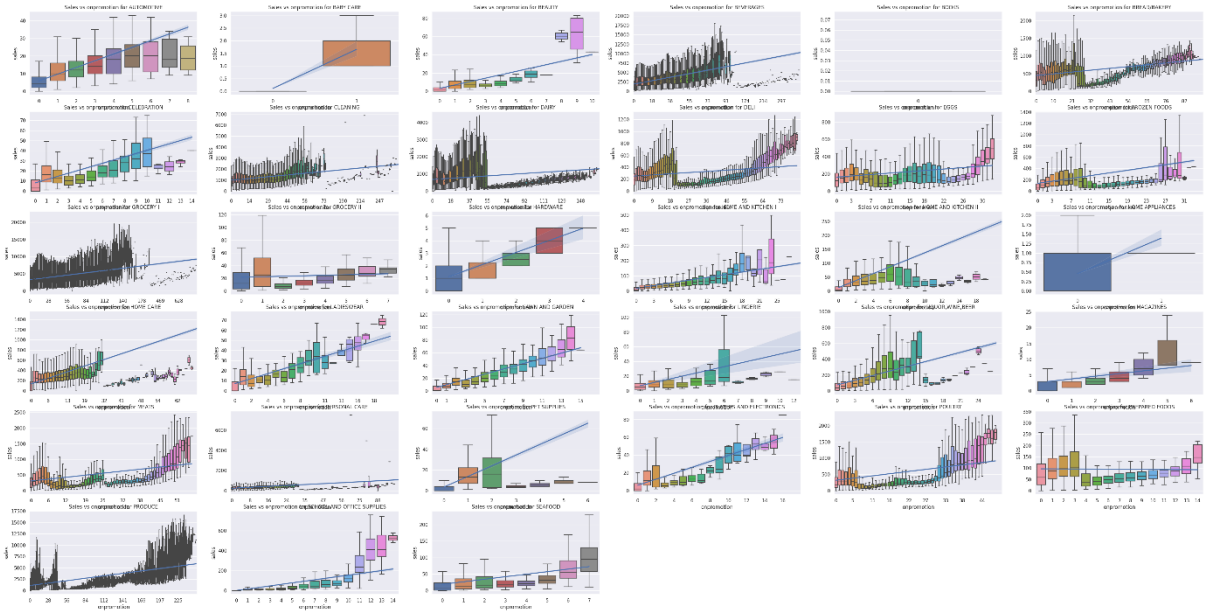

Figure 2: Comparing Sales vs onpromotion for each Product family.

Afterward, the dataset underwent data pre-processing, involving the following steps:

1. **Label Encoding:**
   Label encoding was applied to the 'family' and 'store_nbr' columns in the train dataset.
2. **Date-related Operations:**
   Date-related operations were conducted, including parsing dates and converting them to a datetime format.
3. **Filtering and Rearranging Data:**
   - Data was filtered based on specific conditions, such as focusing on records starting from a particular date.
   - The data was rearranged, and a group-by operation was performed on 'family,' 'store_nbr,' 'date,' and 'onpromotion' columns.
4. **Transforming into Supervised Learning Format:**
   - The dataset was transformed into a supervised learning format using a custom function.
   - The function created lag features to predict future values based on a specified window and lag size.
5. **Train-Validation Split:**
   - The dataset was split into training and validation sets for model training and evaluation.
   - These pre-processing steps were crucial for preparing the data for subsequent modeling tasks, ensuring that it is in a suitable format for training and testing machine learning models.

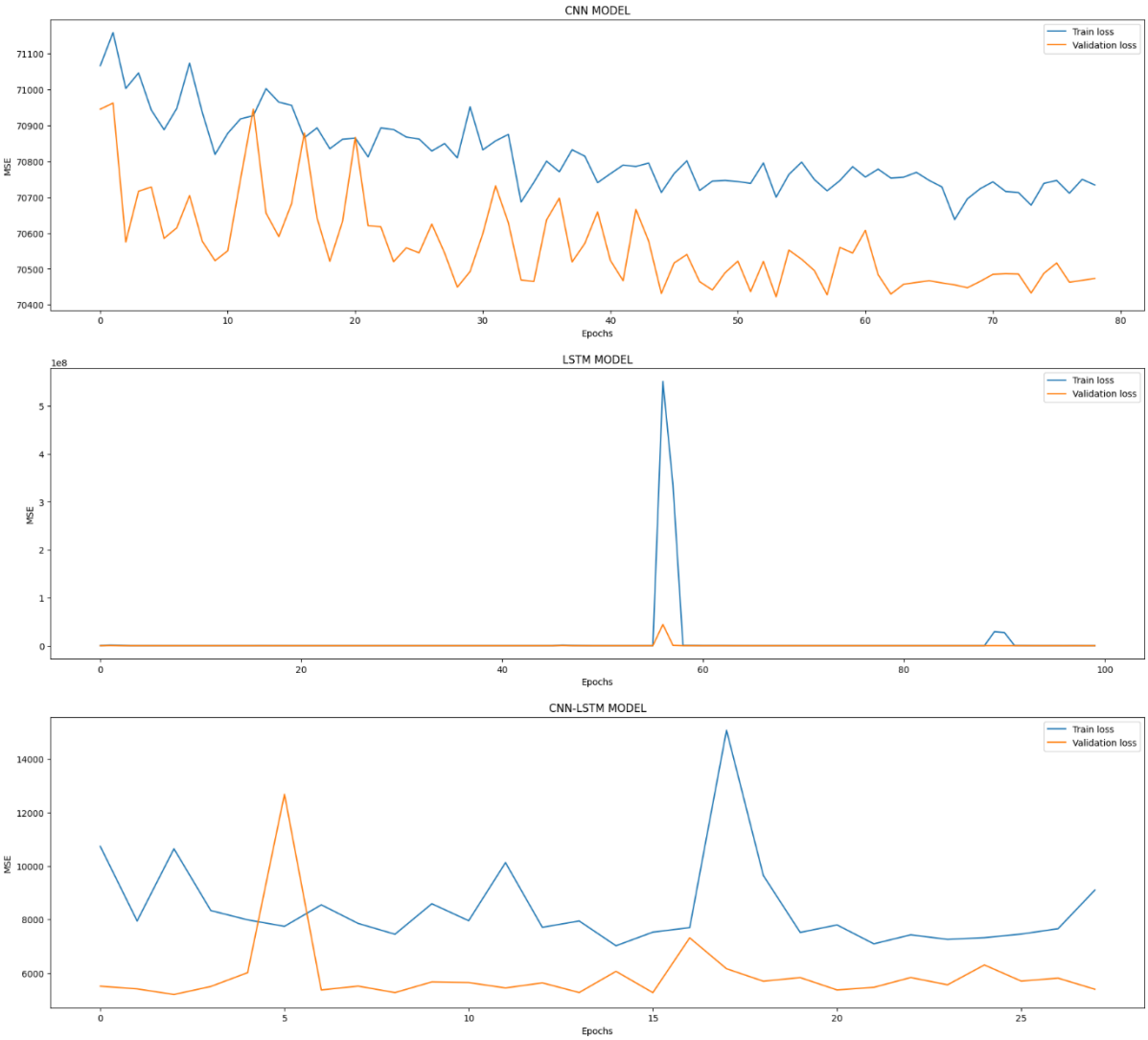

Figure 3: Learning Curves from CNN, LSTM and CNN-LSTM Models.

Among the three models, the LSTM and CNN-LSTM models seem to be learning well, as their training and validation losses are decreasing over epochs. The CNN model, on the other hand, does not show significant improvement, and its loss remains high.

d. Explain results:

The CNN-LSTM model outperforms both CNN and LSTM individually, achieving lower RMSE on both the training and validation sets. This suggests that the hybrid model effectively captures the temporal patterns in the data, combining the strengths of both CNN and LSTM architectures.

1. **CNN Model:**

   - Training RMSE: 265.90
   - Validation RMSE: 265.47

The CNN model's RMSE values indicate the average magnitude of errors in sales predictions. These values are crucial for assessing how well the model generalizes to both the training and validation sets. The lower the RMSE, the better the model's predictive accuracy.

2. **LSTM Model:**

   - Training RMSE: 107.97
   - Validation RMSE: 97.94

   The LSTM model exhibits lower RMSE values compared to the CNN model, suggesting improved accuracy in sales predictions. The training and validation RMSE values help evaluate the model's performance on seen and unseen data, respectively.

3. **CNN-LSTM Model:**

   - Training RMSE: 84.07
   - Validation RMSE: 73.41

   The hybrid CNN-LSTM model shows the lowest RMSE values among the three models. This implies that the combination of convolutional and recurrent neural network architectures results in improved predictive accuracy. The lower validation RMSE suggests better generalization to unseen data.

4. Conclusion and Discussions:

In summary, this project aims to tackle the intricate challenge of store sales forecasting by leveraging advanced machine learning models. Our systematic exploration of Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), and a hybrid CNN-LSTM model underscores our commitment to achieving precision in predicting retail sales. The meticulous tuning of hyperparameters, guided by the Root Mean Squared Error (RMSE) metric, has yielded compelling results, with the hybrid CNN-LSTM model emerging as a frontrunner, demonstrating superior predictive accuracy compared to its individual counterparts.

Amidst our success, we recognize the nuanced challenges that persist. Striking the delicate balance between model complexity and interpretability remains a focal point, especially considering the need for transparent insights among end-users. Adapting to unforeseen events and shifts in consumer behavior poses an ongoing challenge that demands continuous attention. The reliance on historical data as a cornerstone for predictions necessitates a forward-looking approach. The exploration of external datasets and the integration of high-performance computing are avenues through which our models can evolve to meet the demands of a dynamic retail landscape.

Overall, this project establishes the groundwork for a robust store sales forecasting framework. The achievements of the hybrid CNN-LSTM model reflect our dedication to leveraging cutting-edge technologies. As we navigate the complexities of retail forecasting, this conclusion signifies not an endpoint but a juncture where refinement, adaptability, and exploration become imperative for sustained success.