Deep Learning

Predicting Sales of Walmart Products

Team 5

Rudraksh Mishra
Ambika Chundru

School of Graduate Professional Studies
Data Analytics

DAAN 570 – Deep Learning

Spring Semester, 2022

Document Control

Work carried out by:

Name	Email Address	Exhaustive list of Tasks
Rudraksh Mishra	rjm7016@psu.edu	- DA-RNN Model - CNN-LSTM - Data Preprocessing - Model Evaluation - Predicting Sales
Ambika Chundru	ajc7832@psu.edu	- CNN-LSTM - DA-RNN Model - Data Preprocessing - EDA - Predicting Sales

Revision Sheet

Date	Revision Description
Jan-28-2022	Introduction, Problem Statement, Challenges
Jan-30-2022	Related works, Importance and impacts
Feb-6-2022	Data Collection, Data Preprocessing
Feb-7-2022	Related work form scientific journals
Feb-20-2022	Methodologies
Feb-25-2022	Model Evaluation, Results and Comparisions
Feb-27-2022	Discussion of results, Feed Back, Refrences

TABLE OF CONTENTS

1 INTRODUCTION

In the retail industry, forecasting sales accurately is important to produce the required quantity of products at the right time. Though predicting sales, the retail companies can avoid wastages and shortages of products. Thinking in the same way, Walmart shared its real-time data on Kaggle as an M5 competition to enhance its forecasting models. The M5 competition is an iteration of Makridakis competitions to evaluate and compare the accuracy of forecasting methods on time series data. Forecasting competitions are essential for objectively evaluating present forecasting techniques, judging the accuracy of the latest ones, and providing empirical evidence on the ways to strengthen the concept and practice of forecasting. The m5 competition aimed to predict future sales at the product level, based on historical data of the largest retail company of the world. Unlike the previous M-competitions the M5 iteration is unique for multiple reasons:

It was hosted on Kaggle, which is a hub for data science practitioners. Hence the participation size increased drastically, the participants mainly focused on methods that can be divided into Machine Learning and "unstructured".

The data provided, not only contained time series data but the participants were also provided with explanatory variables to enhance forecasting accuracy.

M5 had grouped, highly correlated series, organized in a hierarchical, cross-sectional structure, which depicts the predicting structure for a retail company.

Additionally in this iteration, the series displayed intermittency, which is a typical structure for forecasting unit retail sales at a store or product level.

2 PROBLEM STATEMENT

The aim of the project is to diaplay the most accurate forecasts for a 5-year time series data set, which depicts the hierarchical unit sales of the largest retail company, Walmart. This means that 30, 490 forecasts must be made for each day.

3 CHALLENGES

There are two main challenges which we will face while building models. The time series dataset from Walmart has a lot of intermittent values. This means that, there might be no sales for a long period. This situation might have occurred due to, out of stock items or less shelving in the stores. Due to these intermittent values, the errors will increase, if the sales are forecasted at a normal level while the product is out of stock.

Secondly, we know that if the prediction horizon is then the accuracy of forecasting decreased as uncertainty increase with time. For example, would one believe on the next day's weather forecast or that of a day after 1 month. Since our aim is to produce forecasts not only for the next week, but for a 4-week period, our problem becomes complex due to the size of prediction horizon.

3.1 RELATED WORKS

Kaggle: Yes, this dataset has been used by many projects as this dataset was published for the purpose of a competition. Many people sumbited their projects that were done for forecasting time series data in both machine learning field and deep learning field. We reviewed many of the submissions for the M5 competition for predicting unit sales of products. Most of the models built for this competition were using SARIMA, LGBM, LSTM and XgBoost. It was observed that even though the SARIMA model outperformed the LSTM for the long-term prediction task but performed poorer on the short-term task. Additionally, it was observed that the LSTM model was more robust, compared to SARIMA which relied on data meeting some assumption on seasonality. We know that the accuracy of the model is assessed by the root mean square errors (MSE).

Scientific Papers:

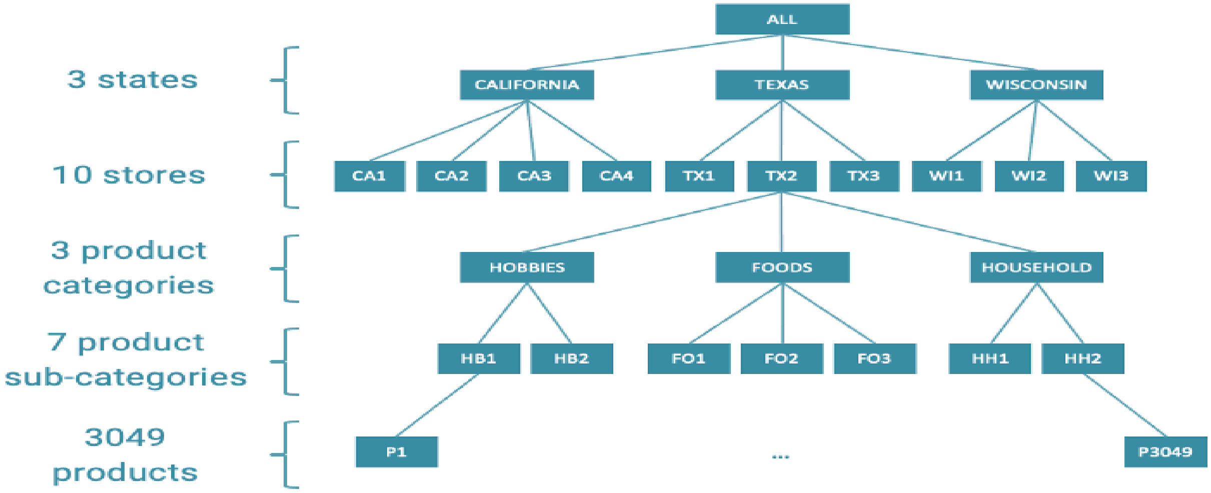
[1] Sales prediction in fashion retail field is a big problem which involves creative and effective tools, importantly in situations where the forecasts are done for the new products which do not have historical data of their sales. (Loureiro et al. 2018). [2] They outline the method of determining the effective features of data, data cleaning, and creating feature engineering in this study using visual analysis of supply chain data. They anticipate supply chain sales using a combination model based on lightGBM and LSTM, and they get good results. Then they compare the results of several models and analyze the model's benefits. (Weng et al. 2019). [3] The use of a long short-term memory (LSTM) neural network for anticipating sales quantities is investigated in this research. In addition, this publication developed a PSO-LSTM hybrid model in which the PSO method is utilized to maximize the number of hidden neurons in different LSTM layers, as well as the number of iterations for training the LSTM neural network. (He et al. 2022). [4] Forecasting has been recommended using the LSTM based technique for multivariate time series, while anomaly detection has been done using the LSTM Autoencoder combined with the OCSVM. (Nguyen et al. 2021).[5] In this research, they distill the convolutional network design with a different starting point (TCN). An additive design of TCN is added in front of a convolution network which is made up of dilated, causal 1D convolutional layers that have the same input and output lengths. This method has unique characteristics like - The architecture's convolutions being causal, which makes sure that there is no information leakage from the future to the past. Just like an RNN, the architecture may take any length sequence and translate it to a length-matched output sequence (Bai et al. 2018). Though these models are quite effective in predicting forecasts in retail industry, they are very different from out forecasting model. In our forecasting model we have used hybrid deep learning model. Which is very effective in predicting unit sales of products in retail industry. The methodology difference will be explained in further sections.

3.2 IMPORTANCE AND IMPACTS

It is very important for retail companies to meet the demand of the customer need and at the same time make sure that there is no wastage as this lead to decrease in revenue. To achieve this the retail market should be able to forecast the demand for different products and stock them up accordingly. By doing this the company will be able to will improve their revenue and at the same time reduce wastage.

4 DATA COLLECTION

The data set provided by Walmart for M5 completion through Kaggle consists of units sales of products sold in the United States of America. The data is organized in the form of grouped time series. It has unit sales of 3049 products which have been grouped into 3 product categories. The 3 categories are hobbies, food, and household these are further divided into 7 departments as given in the figure below. The products can be divided across the 10 stores into the 3 states of the USA namely, California, Texas, Wisconsin. It has been established that the states are stored were selected by Walmart to represent different shopping habits in different locations. In the same way, the product departments are chosen in such a way that they cover consumable and durable goods. The data about the events like SNAP, Superbowl are also included which can be used as features as they result in higher sales.

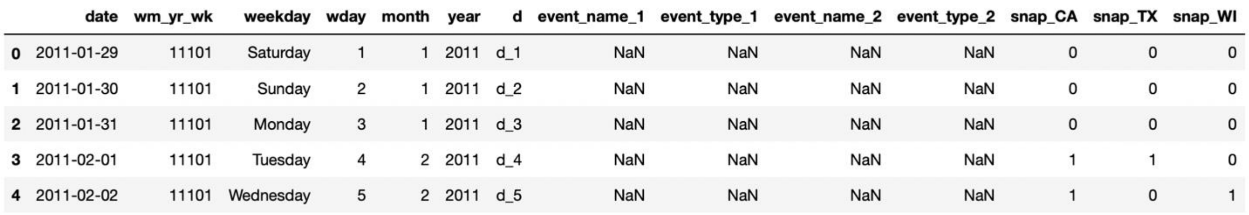


From M5 dataset we will be using the following 2 files for modeling**:**

File 1: “*calendar.csv”*

Contains information about the dates the products are sold.

- *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[1] purchases on the examined date. 1 indicates that SNAP purchases are allowed.



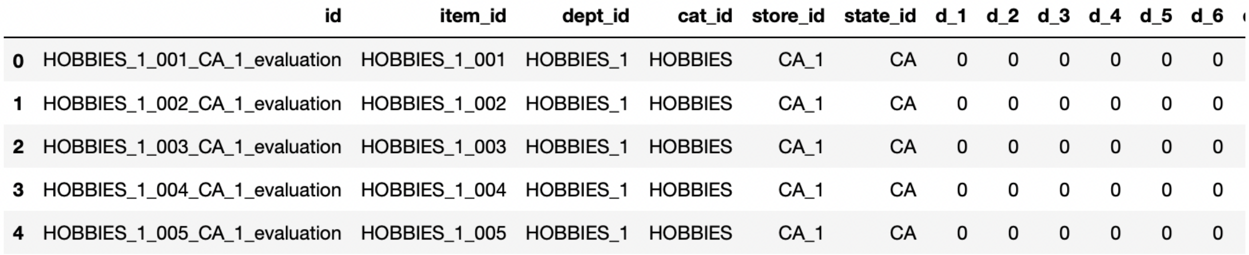
	date	wm_yr_wk	weekday	wday	month	year	d	event_name_1	event_type_1	event_name_2	event_type_2	snap_CA	snap_TX	snap_WI
0	2011-01-29	11101	Saturday	1	1	2011	d_1	NaN	NaN	NaN	NaN	0	0	0
1	2011-01-30	11101	Sunday	2	1	2011	d_2	NaN	NaN	NaN	NaN	0	0	0
2	2011-01-31	11101	Monday	3	1	2011	d_3	NaN	NaN	NaN	NaN	0	0	0
3	2011-02-01	11101	Tuesday	4	2	2011	d_4	NaN	NaN	NaN	NaN	1	1	0
4	2011-02-02	11101	Wednesday	5	2	2011	d_5	NaN	NaN	NaN	NaN	1	0	1

File 3: “*sales_train.csv*”

Contains the historical daily unit sales data per product and store.

- *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.

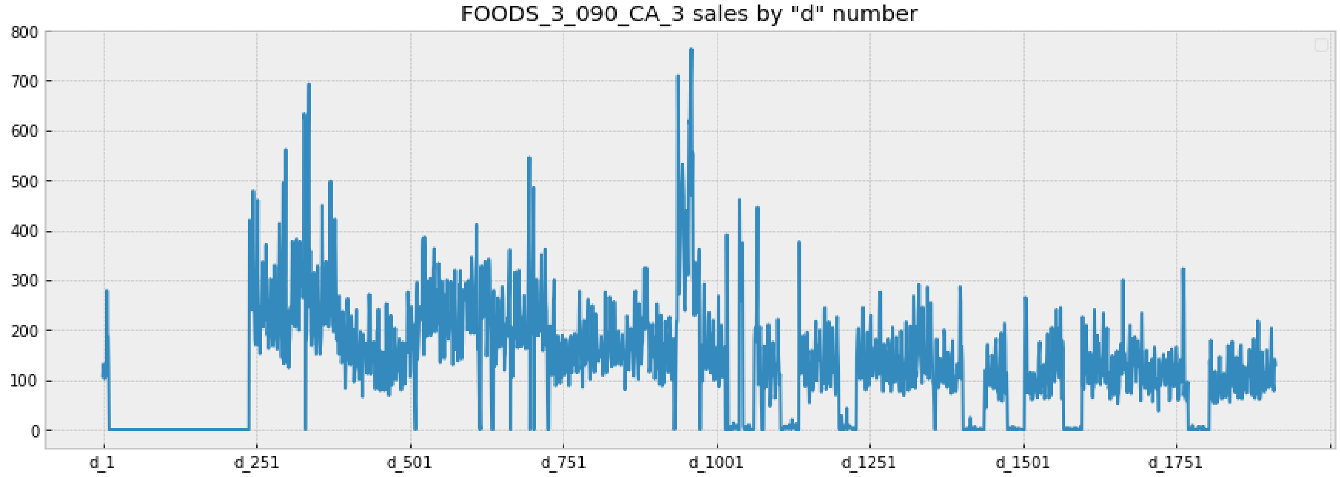
- *store_id*: The id of the store where the product is sold.
- *state_id*: The State where the store is located.
- *d_1, d_2, …, d_i, … d_1941*: The number of units sold at day *i*, starting from 2011-01-29.



	id	item_id	dept_id	cat_id	store_id	state_id	d_1	d_2	d_3	d_4	d_5	d_6
0	HOBBIES_1_001_CA_1_evaluation	HOBBIES_1_001	HOBBIES_1	HOBBIES	CA_1	CA	0	0	0	0	0	0
1	HOBBIES_1_002_CA_1_evaluation	HOBBIES_1_002	HOBBIES_1	HOBBIES	CA_1	CA	0	0	0	0	0	0
2	HOBBIES_1_003_CA_1_evaluation	HOBBIES_1_003	HOBBIES_1	HOBBIES	CA_1	CA	0	0	0	0	0	0
3	HOBBIES_1_004_CA_1_evaluation	HOBBIES_1_004	HOBBIES_1	HOBBIES	CA_1	CA	0	0	0	0	0	0
4	HOBBIES_1_005_CA_1_evaluation	HOBBIES_1_005	HOBBIES_1	HOBBIES	CA_1	CA	0	0	0	0	0	0

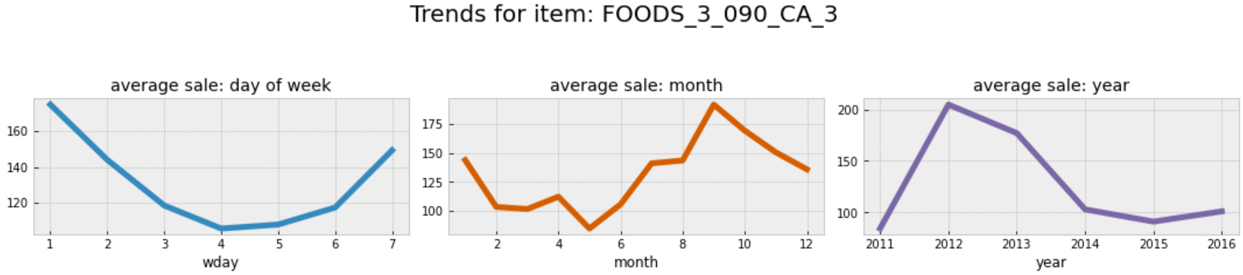
EDA

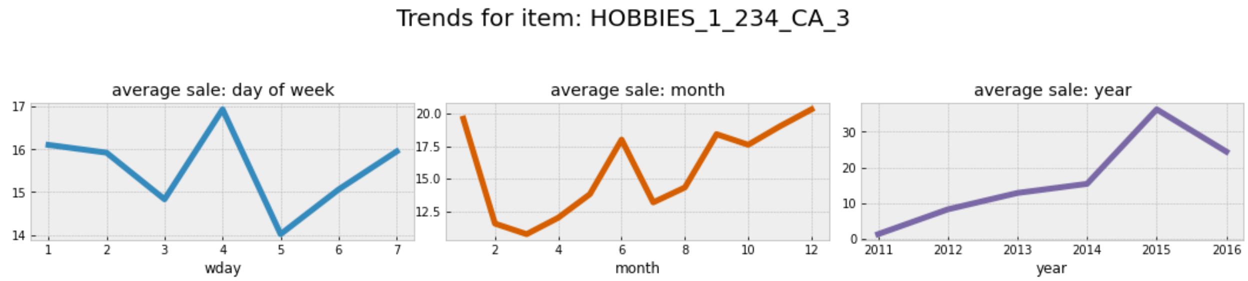
Lets take a random item that sell a lot and see how it's sales look across the training data

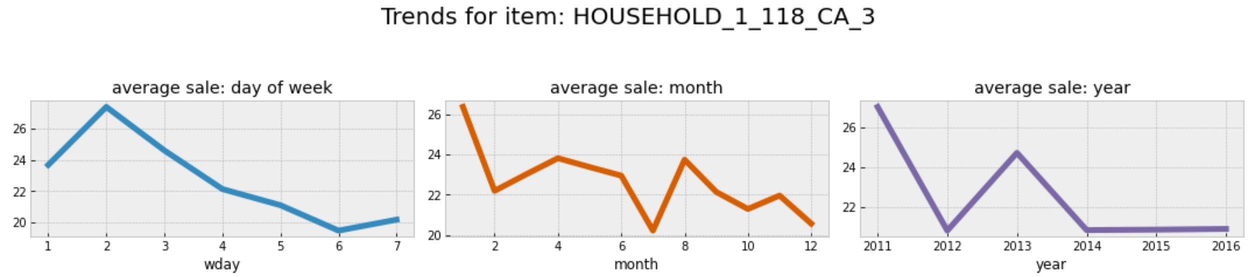


1) SALES BROKEN DOWN BY TIME VARIABLE

   Now that we have our example item lets see how it sells by:
   1) Day of the week
   2) Month
   3) Year

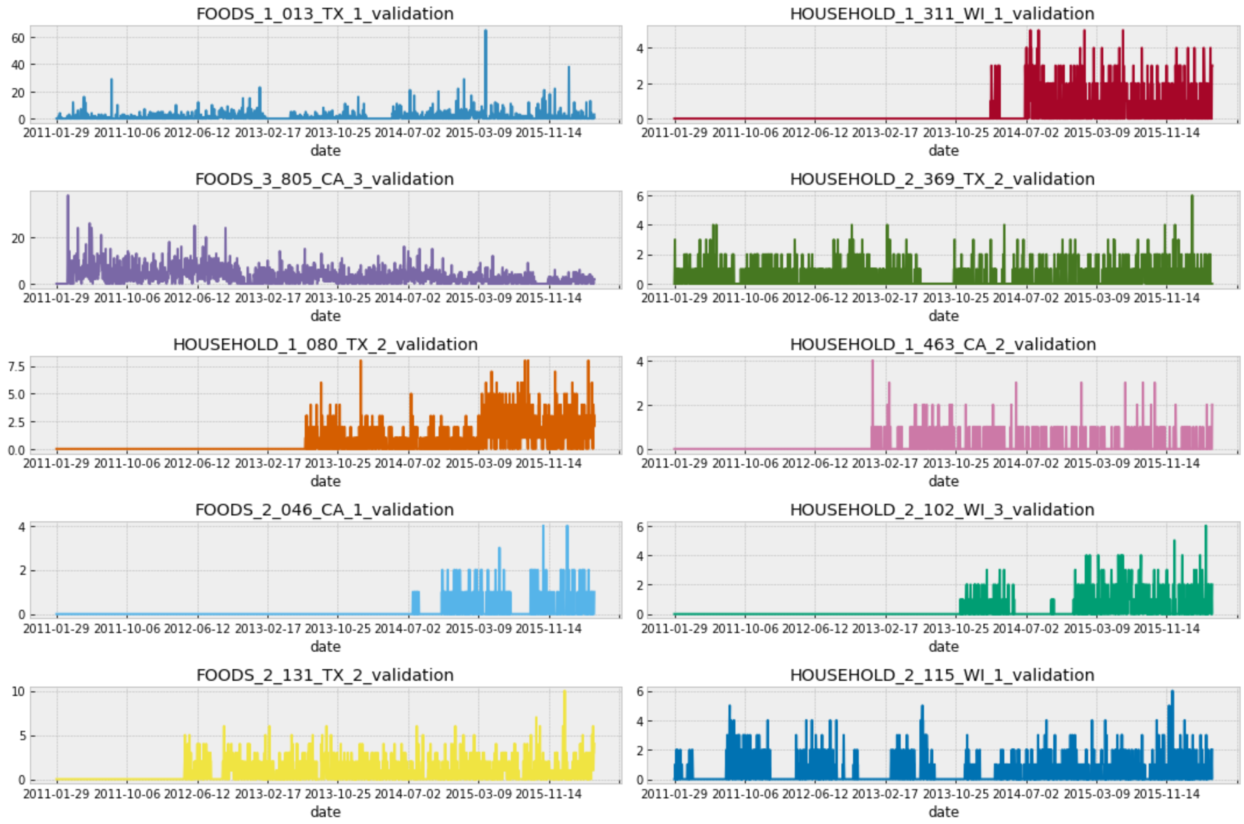




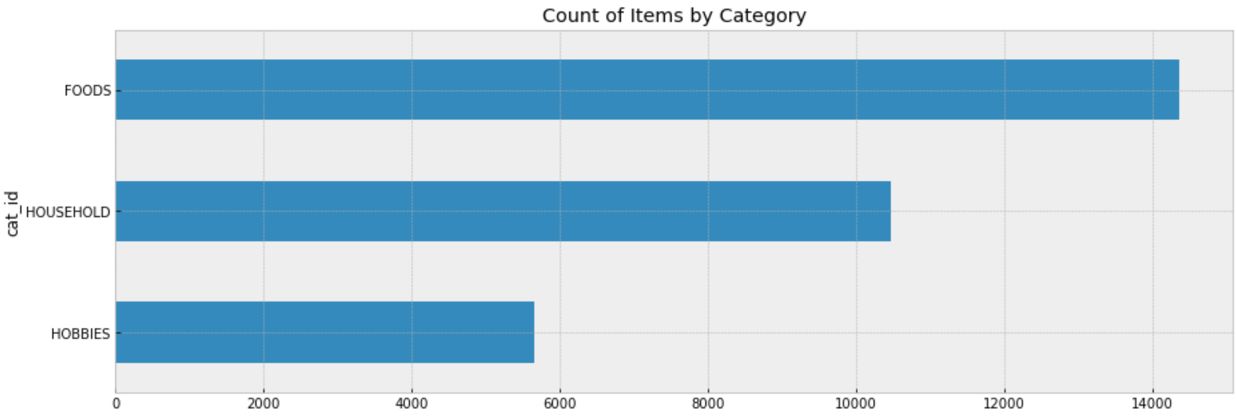


2) SALES OF DIFFERENT ITEMS OVER DATES:

Lets put it all together to plot 10 different items and their sales
Some observations from these plots:
1) It is common to see an item unavailable for a period of time.
2) Some items only sell 1 or less in a day, making it very hard to predict.
3) Other items show spikes in their demand possibly the "events" provided to us could help with these.



3) COMBINED SALES OVER TIME BY TYPE

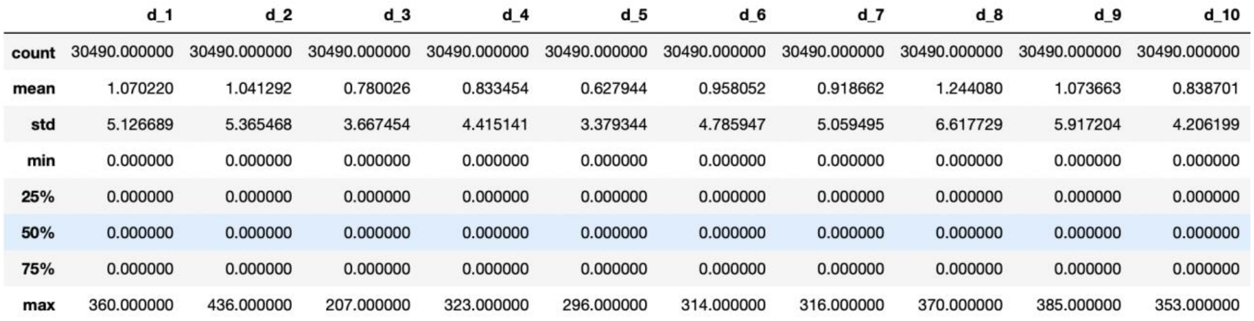


5 DATA PREPROCESSING

To start with we will using back fill to fill the sales of products when they are out of stock. Then we applied downcasting as it helps in reducing the amount of storage used by the dataframes which in turn allows them to speed up the operations performed on them. By observing the sell price data set we can conclude that multiple products have price as zero which means that that product has not been listed and similarly in the sales data set those products have zero sales as well for first 350 days. We then cutoff the data by 350 days. Apart from utilizing sales as a time series element, festive or sporting events can have a big impact on sales. A day before

Thanksgiving or the Superbowl, for example, buyers are more inclined to buy additional snacks or meals. Similarly, with the SNAP program, we anticipate more sales during the day. As a result, we added five new features: event 1, event 2, SNAP CA, SNAP WI, and SNAP TX. Because each shop location's SNAP program days are different, we require three distinct SNAP features. Then we split the data set for training, testing and validation. The validation data set has 141 days of sales for 30490 products and the training data set has 1591 days of sales for 30490 products. After which we use minmax scalar to normalize our data. It's also crucial that our features be scaled throughout the columns. Each column reflects a single day's worth of sales. This ensures that the sales values are between 0 and 1, which aids the LSTM model's gradient descent optimization procedure.

The below data can be used to describe the sale per each day.



	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_10
count	30490.000000	30490.000000	30490.000000	30490.000000	30490.000000	30490.000000	30490.000000	30490.000000	30490.000000	30490.000000
mean	1.070220	1.041292	0.780026	0.833454	0.627944	0.958052	0.918662	1.244080	1.073663	0.838701
std	5.126689	5.365468	3.667454	4.415141	3.379344	4.785947	5.059495	6.617729	5.917204	4.206199
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
75%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
max	360.000000	436.000000	207.000000	323.000000	296.000000	314.000000	316.000000	370.000000	385.000000	353.000000

6 METHODOLOGY

1) CNN-LSTM

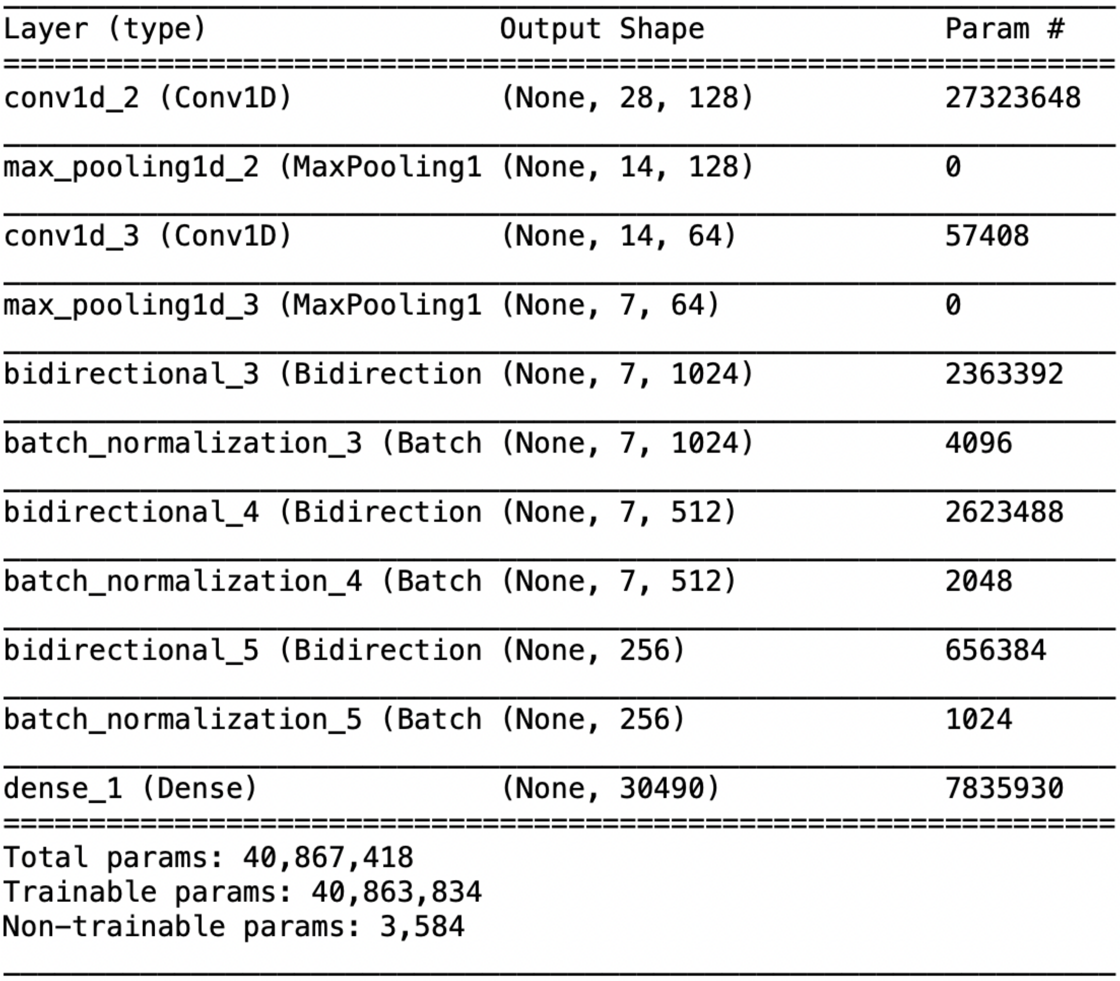
CNN has the characteristic of paying attention to the most obvious features in the line of sight, so it is widely used in feature engineering. LSTM has the characteristic of expanding according to the sequence of time, and it is widely used in time series. According to the characteristics of CNN and LSTM, our timer series forecasting is done

They constitute the appropriate methodology to deal with the noisy and chaotic nature of time-series forecasting problem and lead to more accurate predictions. Long short-term memory (LSTM) networks and convolutional neural networks (CNNs) are probably the most popular, efficient and widely used deep learning techniques

The basic idea of the utilization of these models on time-series problems is that LSTM models may efficiently capture sequence pattern information, due to their special architecture design, while CNN models may filter out the noise of the input data and extract more valuable features which would be more useful for the final prediction model. However, standard CNNs are well suited to address spatial autocorrelation data, they are not usually adapted to correctly manage complex and long temporal dependencies [4], while in contrast LSTM networks although they are tailored to cope with temporal correlations, they exploit only the features provided in the training set. Therefore, a time-series model which exploits the benefits of both deep learning techniques could improve the prediction performance.

CNN-LSTM was utilized, using batch normalization as a regularization option. CNN is divided into two sections (conv1d followed by max pooling layer). The conv1d layer aids in the analysis and extraction of sales patterns over a one-week period (just like extracting "edges" patterns in a typical conv2d on an image). The max pooling layer then summaries these patterns into wider, generalizable patterns, attempting to eliminate noise from daily sales spikes and troughs.

The outputs from CNN are then input into a three-layer bidirectional LSTM model. The number of LSTM units is flipped from 512 to 256 to 128. This has the feature of simplifying and summarizing sales trends, allowing for more accurate forecasting as it allows long temporal dependencies. By minimizing the chance of overfitting, batch normalization improves the model's resilience. The below image gives the model summary



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_2 (Conv1D)            (None, 28, 128)           27323648
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 14, 128)           0
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 14, 64)            57408
_________________________________________________________________
max_pooling1d_3 (MaxPooling1 (None, 7, 64)             0
_________________________________________________________________
bidirectional_3 (Bidirection (None, 7, 1024)           2363392
_________________________________________________________________
batch_normalization_3 (Batch (None, 7, 1024)           4096
_________________________________________________________________
bidirectional_4 (Bidirection (None, 7, 512)            2623488
_________________________________________________________________
batch_normalization_4 (Batch (None, 7, 512)            2048
_________________________________________________________________
bidirectional_5 (Bidirection (None, 256)               656384
_________________________________________________________________
batch_normalization_5 (Batch (None, 256)               1024
_________________________________________________________________
dense_1 (Dense)              (None, 30490)             7835930
=================================================================
Total params: 40,867,418
Trainable params: 40,863,834
Non-trainable params: 3,584
_________________________________________________________________
```

After we build the model we, we fit the model with the train and validation data. We have also initialized early stop function to monitor validation loss. We have used a batch size of 64 and 70 epochs.

The vanishing gradient problem is overcome with the rectified linear activation function, allowing models to train faster and perform better. For the reason we have used ReLU as our activation function. One of the main assumption is that zero sales indicate that the item is out of stock. One of the main limitation is that many products are introduced late to a particular store.

One of the main Machine learning model we came across was XGBoost. Both XGBoost LSTM models can predict multi-step ahead, But XGBoost is better with smaller datasets. Since our dataset is quite huge LSTM worked better on it.

We have used MSE as our loss function. It measures the average of the squared difference between the original and predicted values. it also has the effect of putting more weight on large errors. Since extreme values occurs due to events it is necessary for the model to record them.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2$$

We have used MAE as It represents the average of the absolute difference between the actual and predicted values. It measures the average of the residuals in the dataset.

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}|$$

2) DA-RNN

Another approach that we adopted for time series prediction was the use of a state of the art called DA-RNN(dual-stage attention-based RNN). In LSTM we encode the input as a fixed-length vector and use the decoder to generate a translation. One problem with encoder-decoder LSTM networks is that their performance will deteriorate rapidly as the length of input sequence is large. To resolve this issue, the attention-based encoder-decoder network uses an attention mechanism to select parts of hidden states that is It takes all window size sales into account then assigning relative importance to each one of them.

In the first stage, we develop a new attention mechanism to adaptively extract the relevant driving series at each time step by referring to the previous encoder hidden state. In the second stage, a temporal attention mechanism is used to select relevant encoder hidden states across all time steps. These two attention models are well integrated within an LSTM-based recurrent neural network (RNN) and can be jointly trained using standard back propagation. In this way, the DA-RNN can adaptively select the most relevant input features as well as capture the long-term temporal dependencies of a time series appropriately. It is robust to noisy inputs.

There are three parameters in the DA-RNN, i.e., the number of time steps in the window T, the size of hidden states for the encoder m, and the size of hidden states for the decoder p. To determine the window size T, we conducted a grid search over T ∈ {3, 5, 10, 15, 25}. The one (T = 10) that achieves the best performance over validation set is used for test. For the size of hidden states for encoder (m) and decoder (p), we set m = p for simplicity and conduct grid search over m = p ∈ {16, 32, 64, 128, 256}.

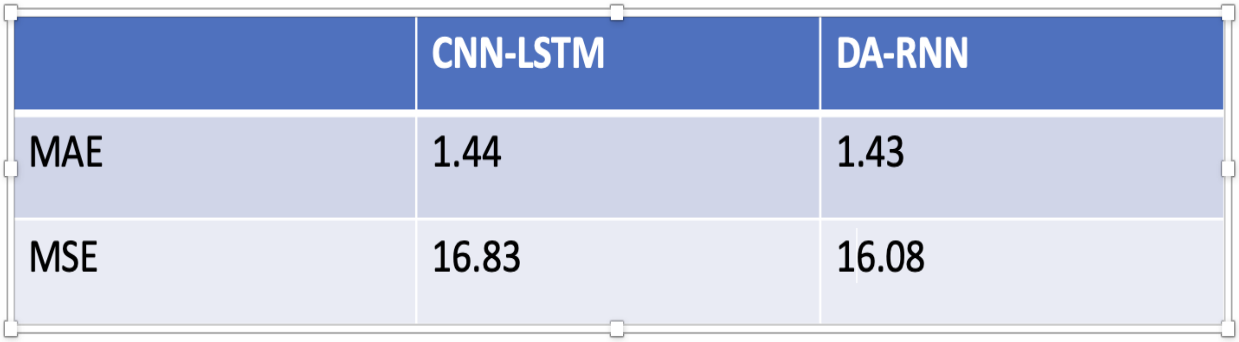
We have used MSE as our loss function. It measures the average of the squared difference between the original and predicted values. it also has the effect of putting more weight on large errors. Since extreme values occurs due to events it is necessary for the model to record them.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2$$

We have used MAE as It represents the average of the absolute difference between the actual and predicted values. It measures the average of the residuals in the dataset.

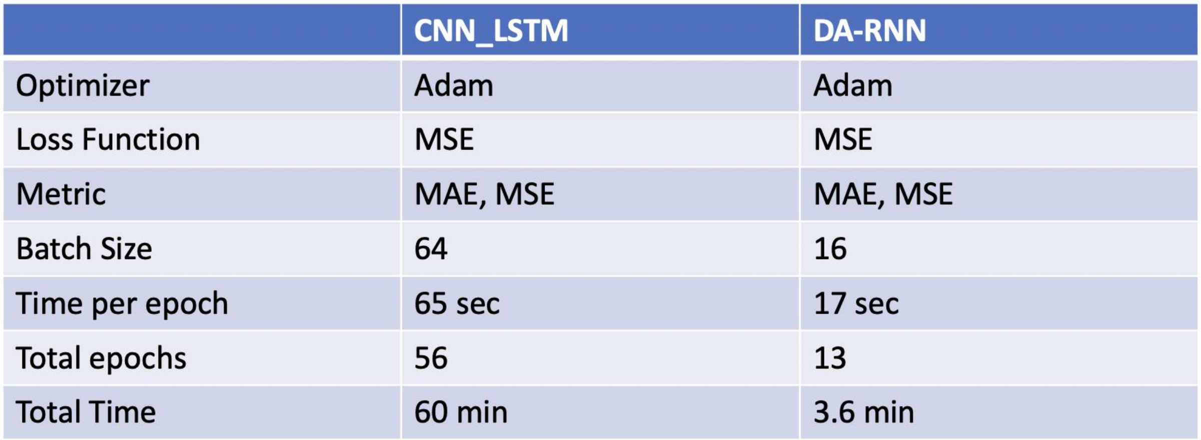
$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}|$$

7 MODEL EVALUATION



	CNN-LSTM	DA-RNN
MAE	1.44	1.43
MSE	16.83	16.08

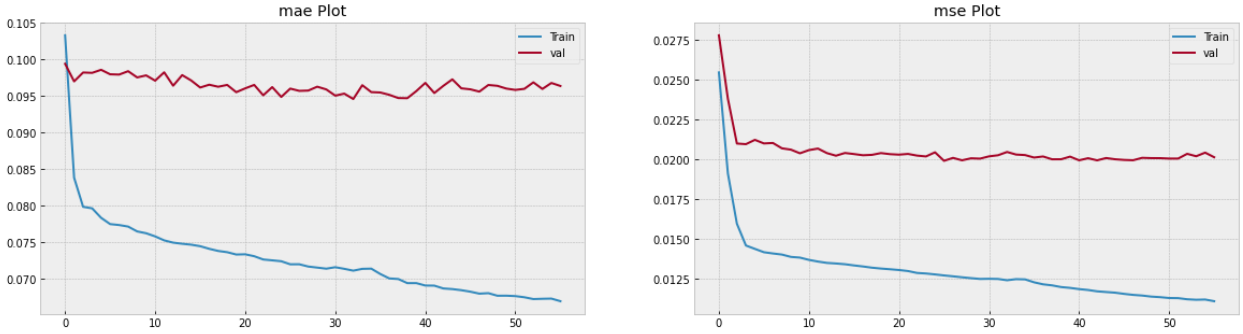
8 MODEL HYPERPARAMETER COMPARISONS



	CNN_LSTM	DA-RNN
Optimizer	Adam	Adam
Loss Function	MSE	MSE
Metric	MAE, MSE	MAE, MSE
Batch Size	64	16
Time per epoch	65 sec	17 sec
Total epochs	56	13
Total Time	60 min	3.6 min

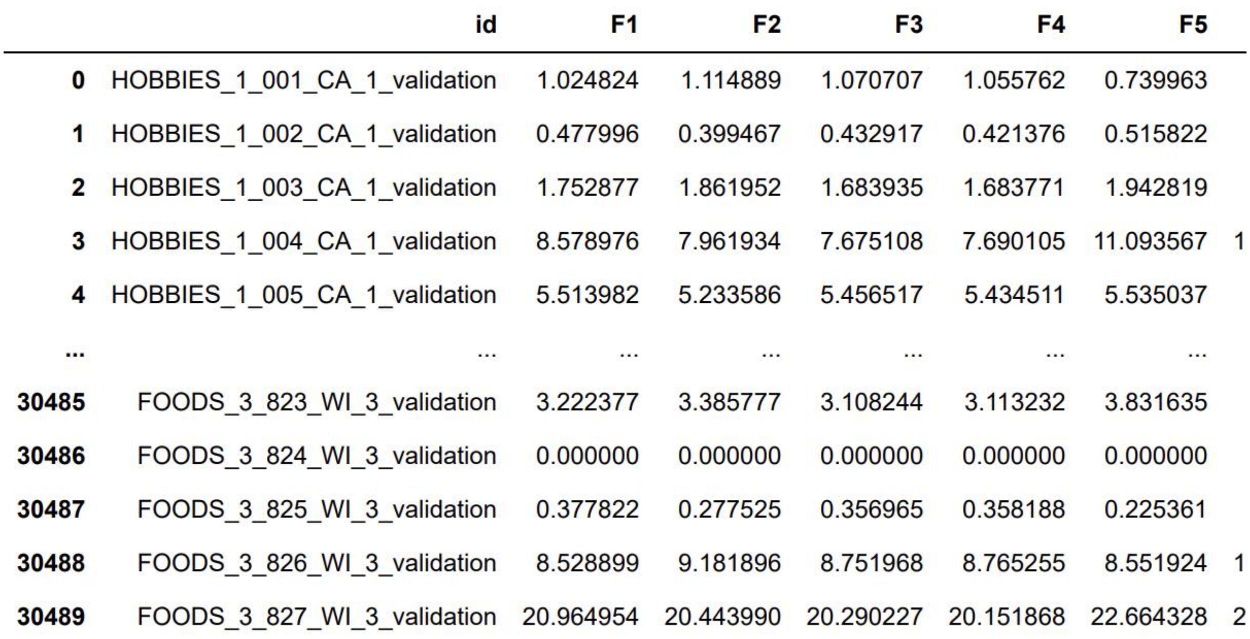
9 RESULTS AND INTERPRETATION

1) CNN-LSTM
   Training Validation Loss Plots:



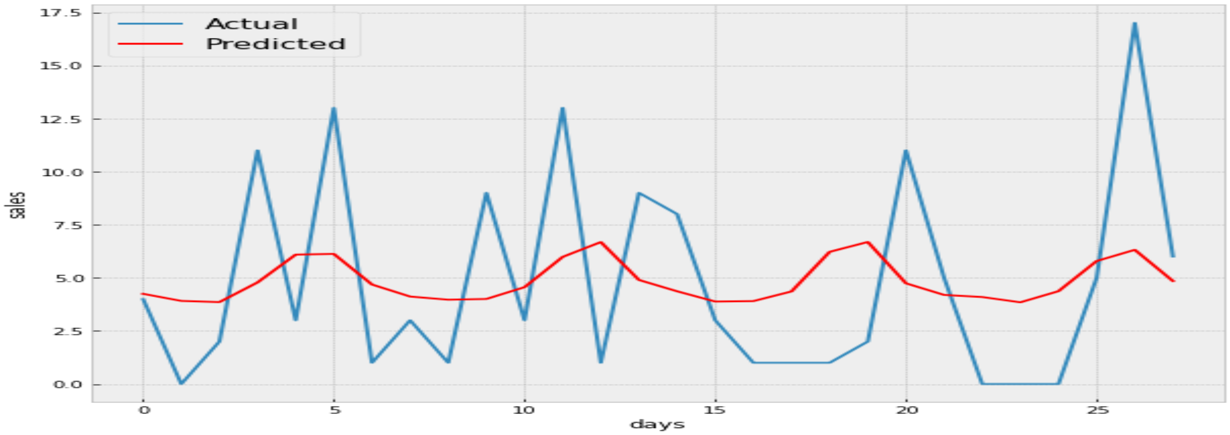
The validation loss has decreased and converged over time, and we can observe it in the above MSE and MAE graphs

Predictions:

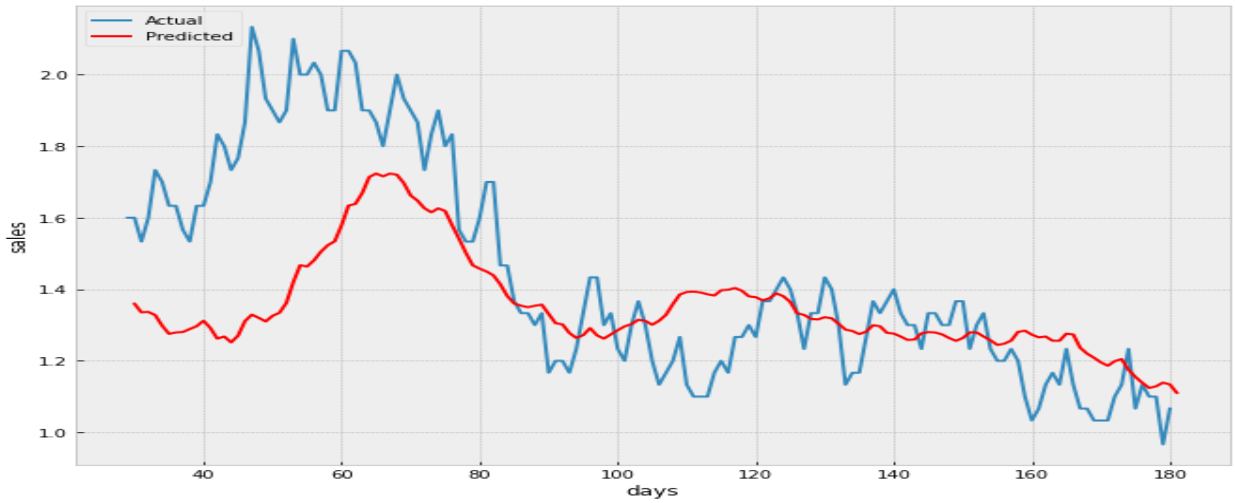


	id	F1	F2	F3	F4	F5	
0	HOBBIES_1_001_CA_1_validation	1.024824	1.114889	1.070707	1.055762	0.739963	
1	HOBBIES_1_002_CA_1_validation	0.477996	0.399467	0.432917	0.421376	0.515822	
2	HOBBIES_1_003_CA_1_validation	1.752877	1.861952	1.683935	1.683771	1.942819	
3	HOBBIES_1_004_CA_1_validation	8.578976	7.961934	7.675108	7.690105	11.093567	1
4	HOBBIES_1_005_CA_1_validation	5.513982	5.233586	5.456517	5.434511	5.535037	
...	...	...	...	...	...	...	
30485	FOODS_3_823_WI_3_validation	3.222377	3.385777	3.108244	3.113232	3.831635	
30486	FOODS_3_824_WI_3_validation	0.000000	0.000000	0.000000	0.000000	0.000000	
30487	FOODS_3_825_WI_3_validation	0.377822	0.277525	0.356965	0.358188	0.225361	
30488	FOODS_3_826_WI_3_validation	8.528899	9.181896	8.751968	8.765255	8.551924	1
30489	FOODS_3_827_WI_3_validation	20.964954	20.443990	20.290227	20.151868	22.664328	2

The above table is the predicted sales for each product over 28 days. Here Fi represents days were i= 1, 2,3 ⋯ and the rows represents the products. Then we use the model to predict the sales of each product for next 28 days. Below are the two graphs we depict the trend of actual and predicted sales of two products over 28 days and 180 days respectively. We can see that we get a better prediction trend for 180 days than that of 28 days.



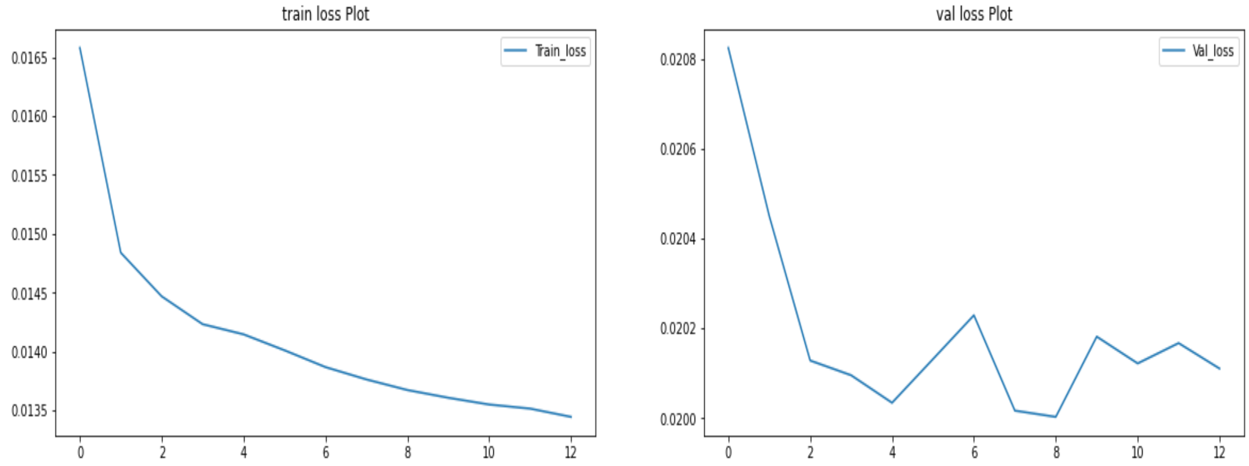
CNN-LSTM Model 28 days prediction trend



CNN-LSTM Model 180 days prediction trend

2) DA-RNN

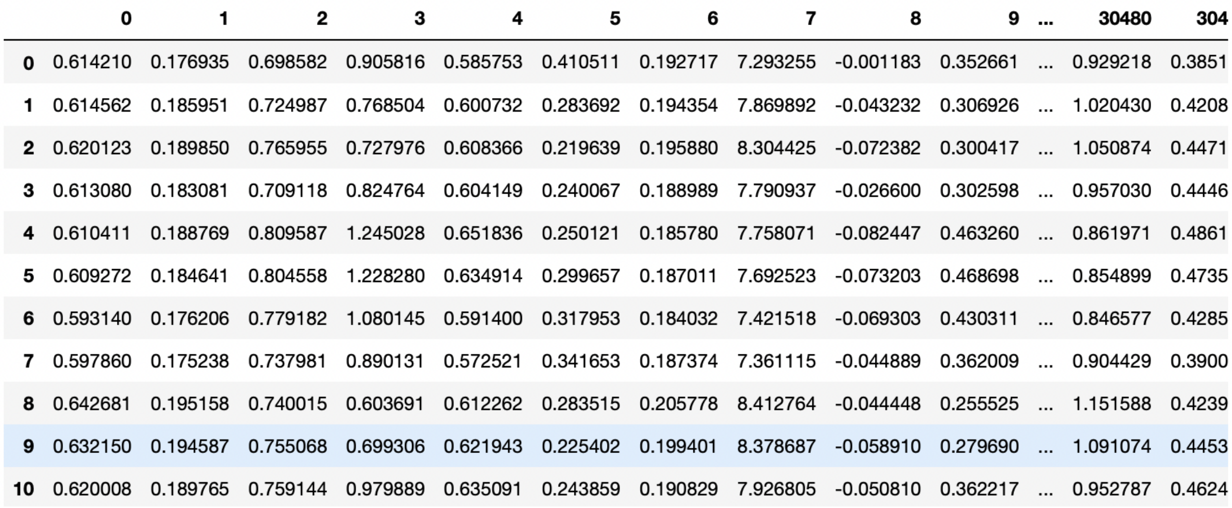
Training Validation Loss Plots:



The validation loss has decreased and converged over time.

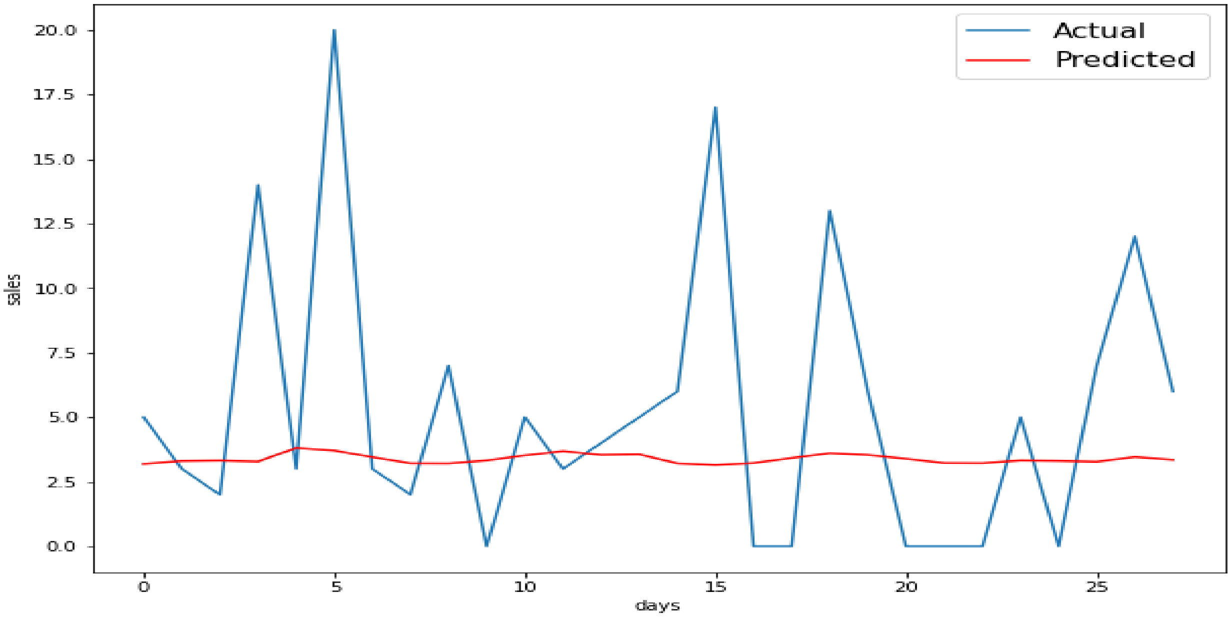
Prediction:

The above table is the predicted sales for each product over 28 days. Here Fi represents days were i= 1, 2,3 ··· and the rows represents the products. Then we use the model to predict the sales of each product for next 28 days.

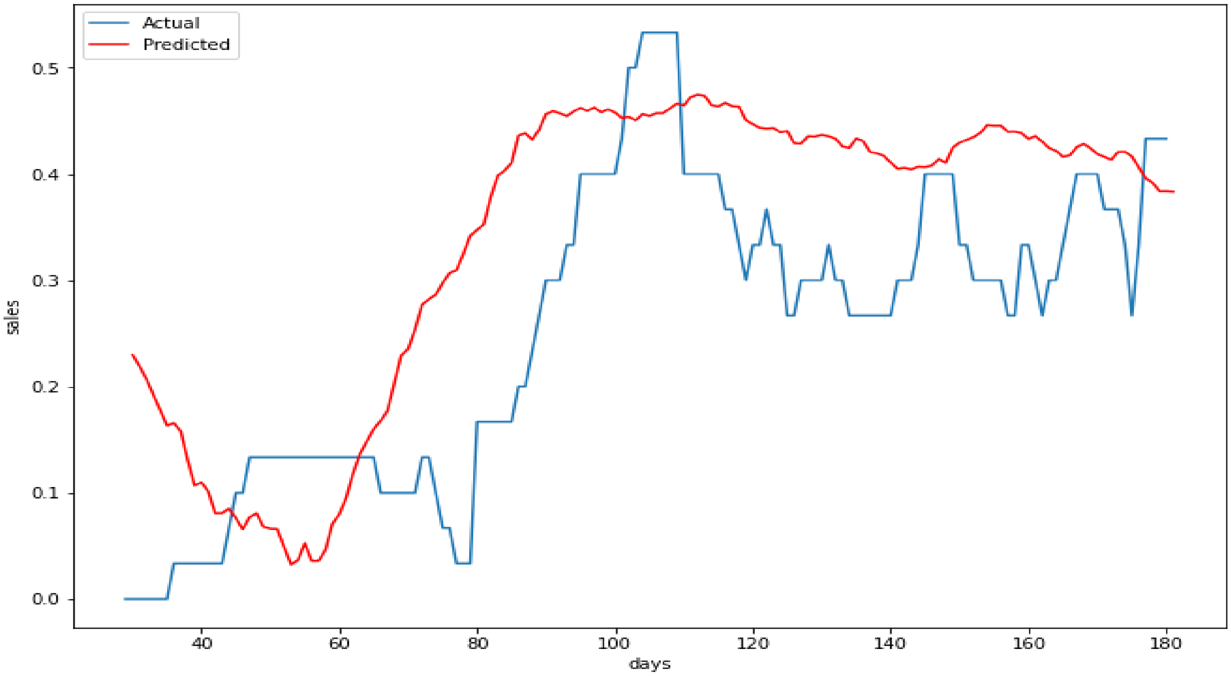


	0	1	2	3	4	5	6	7	8	9	...	30480	304
0	0.614210	0.176935	0.698582	0.905816	0.585753	0.410511	0.192717	7.293255	-0.001183	0.352661	...	0.929218	0.3851
1	0.614562	0.185951	0.724987	0.768504	0.600732	0.283692	0.194354	7.869892	-0.043232	0.306926	...	1.020430	0.4208
2	0.620123	0.189850	0.765955	0.727976	0.608366	0.219639	0.195880	8.304425	-0.072382	0.300417	...	1.050874	0.4471
3	0.613080	0.183081	0.709118	0.824764	0.604149	0.240067	0.188989	7.790937	-0.026600	0.302598	...	0.957030	0.4446
4	0.610411	0.188769	0.809587	1.245028	0.651836	0.250121	0.185780	7.758071	-0.082447	0.463260	...	0.861971	0.4861
5	0.609272	0.184641	0.804558	1.228280	0.634914	0.299657	0.187011	7.692523	-0.073203	0.468698	...	0.854899	0.4735
6	0.593140	0.176206	0.779182	1.080145	0.591400	0.317953	0.184032	7.421518	-0.069303	0.430311	...	0.846577	0.4285
7	0.597860	0.175238	0.737981	0.890131	0.572521	0.341653	0.187374	7.361115	-0.044889	0.362009	...	0.904429	0.3900
8	0.642681	0.195158	0.740015	0.603691	0.612262	0.283515	0.205778	8.412764	-0.044448	0.255525	...	1.151588	0.4239
9	0.632150	0.194587	0.755068	0.699306	0.621943	0.225402	0.199401	8.378687	-0.058910	0.279690	...	1.091074	0.4453
10	0.620008	0.189765	0.759144	0.979889	0.635091	0.243859	0.190829	7.926805	-0.050810	0.362217	...	0.952787	0.4624

Below are the two graphs we depict the trend of actual and predicted sales of two products over 28 days and 180 days respectively. We can see that we get a better prediction trend for 180 days than that of 28 days.



DA-RNN Model 28 days prediction trend



DA-RNN Model 180 days prediction trend

10 DISCUSSION OF RESULTS

From the above outputs we can say that, by predicting sales, the companies can improve their revenue as the avoid wastage and low stocks. As we can observe a good trend of sales over time. For many days the exact sales can be predicted using these two models. One of the main limitations out model doesn't cover all the features we can include more features like end of the month, start of month or week. By adding mode features we can achieve a better model for the prediction of sales. Moving standard deviation can also be explored. Using this model, the companies can meet the demand and provide the supply for the days even when the demand is high. One of the main thing we learn form this project is, by including more features we can improve our predictions

11 YOUR FEEDBACK

-

- We a positive experience while doing this project, as the project deliverables were very structured which allowed us to shape the process of our project development.
- The scientific review helped us a great deal to explore state of the art solutions for the forecasting on time series data.
- Additionally, the office hours allocated for meeting the professor was really helpful.

12 REFERENCES

[1] Loureiro, Ana LD, Vera L. Miguéis, and Lucas FM da Silva. "Exploring the use of deep neural networks for sales forecasting in fashion retail." *Decision Support Systems* 114 (2018): 81-93.

[2] Weng, Tingyu, Wenyang Liu, and Jun Xiao. "Supply chain sales forecasting based on lightGBM and LSTM combination model." *Industrial Management & Data Systems* (2019).

[3] He, Qi-Qiao, Cuiyu Wu, and Yain-Whar Si. "LSTM with Particle Swam Optimization for Sales Forecasting." *Electronic Commerce Research and Applications* (2022): 101118.

[4] Nguyen, H. D., et al. "Forecasting and Anomaly Detection approaches using LSTM and LSTM Autoencoder techniques with the applications in supply chain management." *International Journal of Information Management* 57 (2021): 102282.

[5] Bai, Shaojie, J. Zico Kolter, and Vladlen Koltun. "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling." arXiv preprint arXiv:1803.01271 (2018).

[6] Livieris, Ioannis E., Emmanuel Pintelas, and Panagiotis Pintelas. "A CNN–LSTM model for gold price time-series forecasting." *Neural computing and applications* 32.23 (2020): 17351-17360.

[7] Lu, Wenjie, et al. "A CNN-LSTM-based model to forecast stock prices." *Complexity* 2020 (2020).

[8] Qin, Yao, et al. "A dual-stage attention-based recurrent neural network for time series prediction." *arXiv preprint arXiv:1704.02971* (2017).

[9] Ensafi, Yasaman, et al. "Time-series forecasting of seasonal items sales using machine learning–A comparative analysis." *International Journal of Information Management Data Insights* 2.1 (2022): 100058.