Time Series Forecasting Project – Murat Tirkeshov

I. Introduction

Corporación Favorita is a large Ecuadorian grocery retailer, and the task is to predict daily unit sales for various products across numerous stores. The dataset provided (from a Kaggle competition[1]) includes several years of historical sales data, along with related features such as promotions, holidays, oil prices, and transactions. The goal of this assignment is to build and evaluate three high-performing forecasting models using this data:

1. **Classical time series model (SARIMAX)** – a Seasonal ARIMA model with exogenous variables.
2. **Machine learning model (CatBoost)** – a tree-based gradient boosting regression model.
3. **Deep learning models (ANN)** – a 1D Convolutional Neural Network for time series.

We will train each model to forecast sales and evaluate their predictions on a validation set using the Root Mean Squared Logarithmic Error (RMSLE) metric. RMSLE measures the relative error between predicted and actual values on a log scale and is the competition's evaluation metric. Finally, we will prepare the prediction output in the required Kaggle submission format (with id and sales columns

II. Data

For my final project, I decided to take a challenging time series forecasting problem using the Corporación Favorita Grocery Sales dataset from a Kaggle competition. This dataset contains several years of daily sales data from an Ecuadorian grocery retailer, covering 54 stores and 33 product families. I took this data because of my interest in machine learning and deep learning models, I wanted to experiment with advanced models on a real-world forecasting task. I knew it would be a large-scale problem (millions of rows of data and 54×33 store-family combinations to predict). Given the big volume of the sales data, daily sales per item across years, adding up to well over 3 million rows. I merged in additionally given related datasets and did cleaning. For feature engineering, I created new variables to help models better understand sales patterns: date attributes like day of week, month, and year to capture seasonality; a binary weekend flag to highlight behavioral shifts on weekends; and holiday/event indicators. These engineered features was done for the machine learning models to better prediction.

Before jumping into modeling, I plotted some visualizations to understand the sales patterns. These visualizations gave me insights into seasonality, trends, and factors:

Seasonality: There was a weekly seasonality in the sales data, as I observed from ACF and PACF of the store 1 and family combinations. Many stores in the dataset do peak business on weekends, with a dip on Mondays. This confirmed that capturing a 7-day cycle would be important

[1] The dataset comes from the Kaggle competition *Store Sales - Time Series Forecasting*, which focuses on predicting daily sales for Corporación Favorita. Participants forecast item-level sales using historical data, promotions, and store-level information. Source: Alexis Cook et al., Kaggle Competition.

(e.g., using a seasonal period of 7 in a SARIMAX model, or including day-of-week as a feature in machine learning models). I also noticed yearly seasonality: for example, sales tended to spike every December (around the holidays) for many product families.

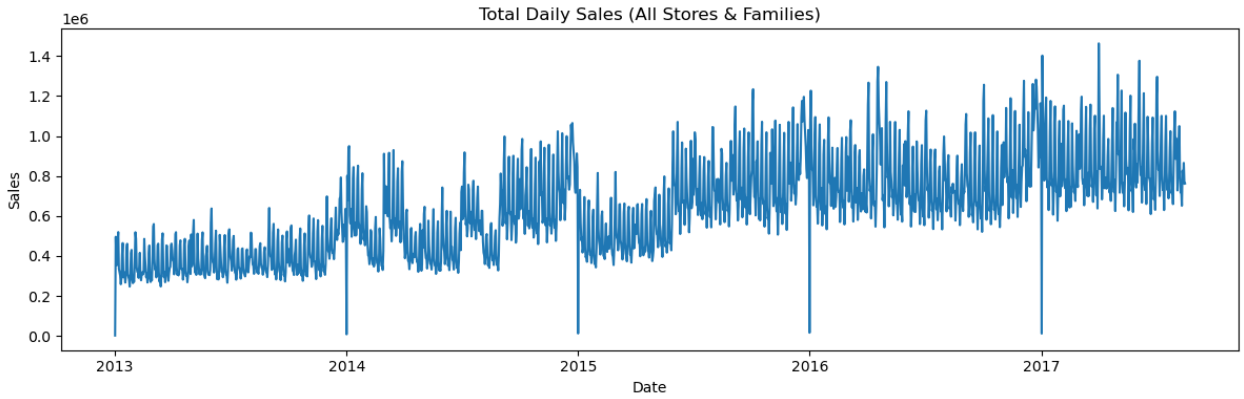

Figure 1: Total Daily Sales Across All Stores and Families (2013–2017)

Promotion Impact: By plotting the sales on promotion events, and holidays I observed significant impacts on sales. For instance, during holidays, sales for certain product families dropped sharply – likely because stores closed.

Correlations: I looked at correlations between different features and sales. One notable insight was the correlation between transactions and sales, which was extremely high (as expected, more transactions mean more sales). I also looked at the correlation between oil prices and sales over time. There appeared to be a mild negative correlation in certain periods, when oil prices dropped significantly, consumer spending power might have increased, slightly boosting sales. The effect wasn't huge, but it was enough to convince me to keep the oil price as a feature in case it helps capture macroeconomic trend effects. Additionally, I examined correlations across stores. Some stores in the same city or of the same type had correlated sales trends. These observations suggested that a model (one that can learn from multiple series) might benefit from sharing information among series, whereas separate univariate models might capture individual patterns better but wouldn't learn from each other. This trade-off was something I kept in mind when comparing model approaches.

III. Models

SARIMAX: For my first model, I decided to start with SARIMAX, a classic time series approach well-suited for data with seasonal patterns and the option to add exogenous variables. plots had shown a weekly cycle, so using a 7-day seasonality in SARIMAX was a natural choice. I began by testing SARIMAX on just one store-family combination out of the 1,782 possible, and I was impressed by how well it captured the ups and downs of the weekly sales cycle. Encouraged by these early results, I extended the approach to every single store-family pair, setting up a loop to train a separate SARIMAX model for each one. I used a standard seasonal ARIMA configuration for series, with fixed (1, 1, 1) for seasonal and non-seasonal lag values, but didn't go for full parameter grid searches since I was already feeling the time demand. In the end, SARIMAX gave me the best overall accuracy (lowest RMSLE) compared to my other models. The downside, though, was the time it took, fitting over 1,700 models was time consuming and left me with no time to fine-tune or rerun for improvements. It became clear that, while SARIMAX

was the best for now, it wasn't practical for my final submission. I couldn't afford to rerun or tweak it for all series again. So, I ultimately had to set aside this approach and keep SARIMAX's results as a benchmark to beat with other methods.

CatBoost: Given the scalability challenges I faced with SARIMAX, I turned to machine learning models that could handle the massive dataset more efficiently, choosing **CatBoost** for its practical speed and ability to handle categorical variables like store ID, family, and city. Unlike SARIMAX, which required a separate model for each of the 1,782 series, CatBoost allowed me to treat the whole dataset as a single, unified supervised learning problem: predicting daily sales based on features like lagged sales, store metadata, day-of-week, month, and holiday flags. This approach was better because I could avoid the loop of fitting separate models. Training CatBoost, was surprisingly quick, it only took minutes once the features were set up. In terms of accuracy, CatBoost didn't quite match SARIMAX's fit for each individual series. The RMSLE for CatBoost on the validation set was a lot higher than SARIMAX, indicating we are not using the data fully.

Deep learning models: The results of my deep learning models were not quite what I hoped for: both the ANN and CNN ended up with higher RMSLE errors on the validation set compared to CatBoost, and they also required much more time to train and tune. I didn't have enough time to properly tune their hyperparameters (like layers, learning rates, or neurons). Despite these challenges, I'm glad I gave the deep learning models a shot. Ultimately, this project gave me a clear hierarchy of model performance: SARIMAX accurate (but slow), CatBoost wasn't as good but way faster, and the ANN/CNN models were interesting experiments.

Random forest: After working through SARIMAX, CatBoost, and neural networks, I decided to take a shot at Random Forest as a kind of experiment. I chose not to do any parameter tuning, just let the Random Forest fit as much as it could, thinking that maybe all my parameter constraints were holding back the models' ability to really find the patterns in the data. It was an interesting experiment: the Random Forest on the full dataset took about half an hour, and to my surprise, its performance was close to what I saw with SARIMAX. I made a submission to Kaggle using the Random Forest's predictions and I saw signs of overfitting, but even so, the results were a lot better than what I got with CatBoost. It made me think that maybe my parameter tuning in CatBoost was limiting its potential. In the future, I'd like to experiment more with how far I can push CatBoost's flexibility while still controlling for overfitting, because it is the fastest model.

IV. Results and Conclusion

SARIMAX was the most accurate model, achieving an overall RMSLE of around 0.42 on the validation set for store-family combinations (as calculated across multiple series in my Jupyter Notebook). However, the downside of SARIMAX was the computation time, it took many hours to run across all 1,782 series, and I couldn't afford to rerun it or fine-tune parameters for each series. So, despite its strong performance, SARIMAX wasn't practical for my final Kaggle submission.

CatBoost, on the other hand, was much faster and could handle the categorical variables directly. I was able to train a single model on the entire dataset in just a few minutes, and while its RMSLE (around 1.0 in validation) wasn't as low as SARIMAX's, it was exceptional enough to provide a good set of final predictions. My first Kaggle submission with CatBoost achieved a RMSLE of close to 2.5, a lot higher than I hoped, but a good learning experience.

Despite giving time to shape the data for neural network input and train the models, they consistently had higher RMSLEs (1.79 for ANN and 1.01 for CNN on validation) and took much

longer to train. On other hand, I think they would need more careful tuning to really compete. Still, setting up these networks was a valuable experience, even if they didn't outperform the simpler models.

Random Forest was a pleasant surprise. Despite concerns about overfitting, Random Forest reached a RMSLE of about 0.49 on validation, significantly better than CatBoost's result. My Kaggle submission with this Random Forest also outperformed my CatBoost submission, at around 0.7 RMSLE, which still indicates that I have overfitted with random forest.

I must admit I may have bitten off more than I could chew. The dataset's size and complexity made this project quite demanding. There were moments I felt overwhelmed dealing with so much data and multiple files. However, I also learned about handling real-world data and scaling up forecasting models. Even though it was hard, I don't regret choosing a big dataset.

Overall, this project was an interesting journey through time series forecasting. I learned that a flexible classical models like SARIMAX still are better, when they're trained to each series' unique patterns. My deep learning attempts were disappointing, they didn't beat the other models. If I were to do this again, I might pick a simpler dataset to avoid such a massive volume. Despite the hurdles, I'm happy with the lessons learned and the fact that I got to see how different forecasting techniques stack up in the real world big data.