Retail Sales Projections Using SARIMA Modeling

David Culhane

DSC 630-T301 Term Project Milestone 4

Andrew Hua

**Data Preparation**

Preparing the data for analysis was the most time-consuming part of this process. It involved taking the three datasets (store information, weekly sales information, and weekly economic features) and merging them into a single dataset and then working with that to find the data for training and testing. The weekly economic features included the local CPI (consumer price index), unemployment percentage, fuel price, and temperature. The cost of fuel each week was found to be the same each week across all stores, so this was likely a national average instead of a local one. That field was dropped since it would likely have no predictive power.

The merging itself came as a two-step merge. The first was a clean, many-to-one merge between the sales and store data using the store numbers as the key field. The second merge involved the result of the first merge with the features data using the store number and date fields as the merge keys. This produced a dataset with weekly sales data and economic indicators for each store and department.

The dates in this dataset were listed as strings with a European dd/mm/YYY format, so these were converted into date objects using a function that was written to map onto the date feature via lambda function. The holiday field was also converted from True/False to 1/0 using a written function.

Since there were three different store types, it was also worth checking how they compare to each other. The three store types have different mean and median weekly sales values and store sizes. This suggested that splitting the store types into their own data frames would be a god idea. Further graphical exploration plotting the weekly sales data for each line of data (a single store and department combination) with the color of the point indicating the store’s type confirmed this, shown below.

A chart showing a variety of colored dots

Description automatically generated with medium confidence

We can see the bands of color for each point in the combined dataset. This reinforced my decision to split the dataset into three subsets by store type.

The five MarkDown columns were removed since they were mostly empty and imputing those values would have improperly skewed the data.

An important step for data preparation was determining the size of the training and testing sets. Since the weekly data spans from February 5th, 2010 to October 26th, 2012, I decided that six months of testing data should suffice. The split for the training and testing sets was made at April 1st, 2012.

However, the resulting datasets were the weekly sales data for each department in each store for that store type. This created a problem when it came to modeling. Aspects of those problems will be discussed in the next section. As a data preparation problem, having multiple datapoints for each given week was a problem for SARIMA modeling. To remedy that problem, a single department would need to be chosen so that the data becomes sequential. The training and testing sets for store 1, department 1 of store-type A (which will be referred to as A11); store 3, department 1 of store-type B (which will be referred to as B31); and store 30, department 1 of store-type C (which will be referred to as C301) were chosen from the training and testing sets to model with.

The final step was to further split each training and testing set into endogenous and exogenous variable sets for the SARIMA models. The endogenous variable is the actual dependent variable being modeled and needed to be stored individually. The exogenous variables are context variables used to inform the model. In the training sets, exogenous variables help inform the relationships in the model. In the testing sets, exogenous variables help inform the predictions made. Inside the six resulting sets of exogenous and endogenous variables, the index of the dataframe was made to be the date for the week. From here, modeling was able to start.

**Model Choices**

Between the completion of Milestone 3 and the modeling stage, I found that linear regression was not the best choice of model. Sales data is periodic and a simple linear model would not be able to capture that nature of the data. Luckily, I learned of SARIMA modeling: Seasonal Autoregressive Integrated Moving Average. SARIMA models take seasonal time-series data to make predictions using multiple short-term regression windows (AR), differencing the data if needed to stabilize trends (I), and moving averages of regression error (MA). The seasonal aspect comes from cyclical, periodic changes in the behavior of the data being modeled.

To find the best approach for SARIMA modeling, I decided to use two different models in order to compare them. I decided to use the statsmodels SARIMAX class function as well as the pmdarima library’s auto-ARIMA function.

SARIMA model parameters use trend and seasonal elements. The trend elements govern the ARIMA elements: p for autoregression order, d for difference order, and q for moving average order. These values can be determined by checking autocorrelation (ACF) and partial-autocorrelation (PACF) plots, also provided by the statsmodels library.

|  |  |  |
| --- | --- | --- |
|  | **ACF Plot** | **PACF Plot** |
| **A11** |  |  |
| **B31** |  |  |
| **C301** |  |  |

Values outside the highlighted band are considered significant. The value of p goes along with the PACF plot while the value of q depends on the ACF plot. Since the value at x = 1 was the last significant value before the plots moved towards 0 for working their way towards zero, I used 1 to be p and q for the statsmodels SARIMAX models.

The SARIMAX class function also contains a seasonal order parameter with P, D, and Q that follow the ARIMA values as well as m, the number of periods in a season in the data. Since the data is weekly and one “seasonal period” is considered to be a year, m was chosen to be 52. Once these parameters were chosen, the endogenous (weekly sales) and exogenous (other economic indicators) were fed to the model to train, fit, and make predictions.

The pmdarima library’s auto-ARIMA function works in a similar manner but creates multiple models and compares them to each other when given starting values of p, d, P, and Q values with the option to toggle whether the model generated should be seasonal or not. This takes the choice out of the hands of the user and the various models are scored using either mean squared error (the default metric) or mean absolute error (can be chosen).

**Model Results and Interpretation**

After fitting models for each A11, B31, and C301 using SARIMAX and auto-ARIMA, I displayed the auto-ARIMA parameters and organized the scoring metrics.

The resulting parameters from the auto-ARIMA model were as follows in the form of (p, d, q) (P, D, Q, m): **A11** (0,0,3)(1,0,0,52); **B31** (2,0,1)(1,0,0,52); **C301** (0,0,1)(1,0,0,52). These were chosen by the auto-ARIMA algorithm because the had the best mean squared errors when compared to other combinations of parameters.

The models’ projections were scored using root mean squared error (RMSE), mean absolute error (MAE), and R2 values to evaluate the models statistically and graphs to visualize the actual versus predicted performances.

For ease of organization, I put these metrics together into a dataframe with the standard deviation and median values of each store’s weekly sales.

SARIMAX Metrics:

A screenshot of a graph

Description automatically generated

Auto-Arima Metrics:

A screenshot of a graph

Description automatically generated

We can see from the images of the tables that the error values for RMSE and MAE are large. It should be noted that these values are in terms of the dependent variable, the weekly sales of the department. These values themselves were in the thousands of dollars. The error values were fractions of the standard deviation and median sales values, which should temper judgements. The R2 values were not fantastic either. This means that the projections didn’t line up as well with the actual values in some places and that the exogenous variables used in the model did not explain a majority of the variance in the data.

We can organize the graphs below to compare the models and their projections.

|  |  |  |
| --- | --- | --- |
|  | **statsmodels**  **SARIMAX** | **pmdarima**  **Auto-ARIMA** |
| **A11** |  |  |
| **B31** |  |  |
| **C301** |  |  |

We can see in each of the plots of the actual sales data that there are notable periodic spikes and drops for certain times of the year. SARIMA models are supposed to take these behaviors as well as other contextual information to make predictions. For each of the predictions, we can see that there are spots where they follow the actual data well and others where there are noticeable differences.

A11’s plots tend to stay in similar ranges to the actual sales values each week with minor differences. A-type stores tended to have higher sales overall, so deviations from the actual data weighed heavier for RMSE and MAE values. These error values were still only a few thousand dollars across a whole week and the plots look decent.

B31’s plots had the highest R2 values among those plotted here. The projections followed the actual data closely with MAE being around $1,000 for a week’s sales and most of the errors came from over-projections in May 2012. B31’s model clearly performed better than A11’s and C301’s.

C301’s plots show dramatic differences in September 2012. In that month for both models, notable drops were projected that mirrored September 2011 but the actual data showed increases in sales similar to September 2010. After the drop though, the projections increase in sync with the actual weekly sales. These large differences from the drop likely led to the significantly worse R2 values shown in the tables above.

**Conclusion and Recommendations**

SARIMA models do well when it comes to forecasting retail sales data as a form of univariate time series data. The main problem with the models here is that they likely do not have enough training data. The stores used for modeling each had 143 rows of data, with 113 rows in the training set. Situations where the training data does different things at the same point in a period were found in the data for C301: September 2010 and 2011. The model then had to make a choice of what to do for C301’s projections when it came to September 2012. The model chose incorrectly, and it hurt the model’s statistical metrics. In an actual retail environment, those in charge of the models making these predictions would have more than 2.75 years of data to work with. The 113 rows of training data out of 143 total rows would indicate that 80% of the data went to the training set, which is a justifiable split. More data would have been better for all the models, however.

Otherwise both SARIMA models accomplished their tasks. statsmodels’ SARIMAX function did appear to perform slightly better than pmdarima’s auto-ARIMA function. SARIMAX also did this with less time in computation. Each auto-ARIMA model had to create and test more than a hundred models and select the best since the specified start parameters for p, q, P, and Q were 1 and the default maximum values for each were 5. The SARIMAX models did not have this issue since the values for p, d, q, P, D, and Q were specified after reading from ACF and PACF function charts. So the “correct” choice of model would very well depend on the situation. If the goal is to fully automate model creation based on the data fed to it, auto-ARIMA would be a better choice while SARIMAX would be better for tailoring models to individual situations.

# References

Brownlee, J. (2019, August 21). *A Gentle Introduction to SARIMA for Time Series Forecasting in Python*. Retrieved from Machine Learning Mastery: https://machinelearningmastery.com/sarima-for-time-series-forecasting-in-python/

Justin, L. a. (2022, September 7). *How to build ARIMA models in Python for time series forecasting*. Retrieved from Youtube: https://youtu.be/-aCF0\_wfVwY?si=zB6hYwpk5-N7MPZ4

M. Abdullah, S. Y. (2024). *Retail Sales Data with Seasonal Trends & Marketing*. Retrieved from Kaggle: https://www.kaggle.com/datasets/abdullah0a/retail-sales-data-with-seasonal-trends-and-marketing

*SARIMAX*. (2024, October 3). Retrieved from statsmodels: https://www.statsmodels.org/stable/generated/statsmodels.tsa.statespace.sarimax.SARIMAX.html#statsmodels.tsa.statespace.sarimax.SARIMAX

Singh, M. (2017). *Retail Data Analytics*. Retrieved from Kaggle: https://www.kaggle.com/datasets/manjeetsingh/retaildataset

TeKnowledGeeK. (2021, October 21). *Time Series Forecasting using SARIMAX and compared with ARIMA*. Retrieved from Youtube: https://youtu.be/JO0gFP\_q4uc?si=HAYi1WxAkZWEM3yp

# Milestone 3: Preliminary Analysis

***Will I be able to answer the questions I want to answer with the data I have?***

The data to be used for this analysis will most definitely accomplish the goal of making sales predictions. The main question will be if the models employed are able to predict the weekly sales total of a given department of a given store. The data to be used in the models have data regarding store, department, the week, sales data, and other data that can affect sales. These fields should be able to work together to predict sales numbers.

***What visualizations are especially useful for explaining my data?***

Line charts and stacked area charts would likely be best for explaining aspects of the data. Line charts will be able to all the various aspects of sales and aspects that affect sales on a two-dimensional plane. Any plot could tie a couple factors together and a third could be employed by use of color. Stacked area charts would be able to see store-wide sales by stacking the department sales numbers of each week on top of each other. No matter the graph, the aspect of time must be preserved if possible.

For any line chart though, it would be advised to use no more than a couple of departments in any plot. The sheer number of store and department combinations would make any plot attempting to plot them all entirely unreadable. Ideally, only one graph per pair of store number and department would be used and individually analyzed.

I used a scatterplot to analyze how the different store types experience different amounts of sales. No differentiation was given for individual departments and it was clear that each store type had clear areas, allowing for the decision to be made to split the data by the type of store to make sure that the models used are not compromised by the store type.

***Do I need to adjust the data and/or driving questions?***

The data has been slightly adjusted. All three dataset components have been merged to make sure each line in the resulting set has a sales number for a department as well as store and department number and influencing factors like temperature, gas prices, and unemployment. The three different store types had different sales profiles, so they were each given their own dataframe to be used for modeling.

The fields related to anonymized sales data unfortunately had to be removed. All five of those fields only had data for dates after November 2011. That meant each field was missing more than 60% of its data. My fear was that imputing these fields would dramatically skew the model if those fields were incorporated. To avert that, those five fields were removed.

***Do I need to adjust my model/evaluation choices?***

I will not need to adjust my model choice since the objective has not changed even though we are moving from a possibility of one model to a certainty of three models to use. A linear regression or LOESS model would likely be the best model to use.

The main question will be at what date in the data is the line drawn to split the dataset into training and testing sets. Random selection is not an advisable path since factors in the data are time-dependent. It could make sense to use the 2010 and 2011 data as part of the training set and 2012 data as the testing set even though the 2012 data ends in October.

Model evaluation should proceed as planned, though use of a ROC chart and its AUC will not fit the objective of this analysis and prediction since they lend themselves to categorical target variables. R2 and RMSE scores will still provide assessments for the predictions made by the models.

***Are my original expectations still reasonable?***

I believe that my original expectations are reasonable. The store and department fields will be turned into dummy variables for each dataframe before being given to the models. The factors affecting sales at each store over a given week (temperature, gas price, CPI, unemployment, and time of year) will be used from the testing data to help predict the sales after the model has been trained. These predictions can be scored to evaluate the model’s performance and judgements can be made from those scores.

# Milestone 2

Predictive analytics can be applied throughout various business spaces in order to give a business an idea of what to expect in the future. All a business needs to apply predictive analytics is vast amounts of data, either acquired through its own records organically or by purchasing it as a business asset. Retail environments specifically can make use of predictive analytics to inform almost every aspect of their business operations to forecast when customers will be shopping, what these customers will be buying, how much staff should be on hand, when to order more of any individual product for their inventory, and the list could continue to go on. During the period I spent working in retail, I was able to access the result of predictive analytics in real time, our store’s sales goals. These predictions would be the basis for staffing decisions and be a measuring stick for how our store performed for management.

If a major retail chain is able to use predictive analytics to predict the performances of its stores, that gives it an edge when it comes to making predictions for how the company should report to its shareholders and make a litany of other related decisions. For this reason, I have chosen sales prediction/forecasting as the topic for my term project. The data I plan on using for making these predictions comes from Kaggle at <https://www.kaggle.com/datasets/manjeetsingh/retaildataset>. The dataset itself spans three Excel CSVs, a features dataset, a sales dataset, and a stores dataset. Together, they describe weekly sales data across 45 retail stores across different regions from 2010-2013. Each store has its own departments and sales information.

The features dataset contains fields for the store number (anonymized to numbers 1-45), the date the week starts with, the average temperature in the store’s region for the week, the cost of fuel in the store’s region that week, five markdown fields used to mark sales taking place after November 2011, the value of the consumer price index in the store’s region that week, the unemployment rate in the store’s region that week, and whether or not the week is considered a holiday week. Across these fields there are 8,190 rows of data. The stores dataset contains information about the type and size of each store. Lastly the sales dataset has fields for store number, department number, the date the week started with, the department’s sales that week, and whether the week was considered a holiday week. Since there are 45 stores each with their own departments, the sales dataset is much larger than the features dataset, clocking in at 421,570 rows of data to use.

With this glut of data, we will need to learn a model or set of models through either store-wide sales approach or department by department approach. Since the nature of our target variable, sales, is continuous, a linear regression model will likely be the approach taken. LOESS modeling and linear modeling with polynomial functions could also come into play since they can follow the nature of sales is not necessarily linear. Additional exploration in this space can be explored. The goal will be to see if our model(s) can successfully predict the sales figures towards the end of the dataset. Since the data we are using is chronological, I intend to split the data into training and testing sets by year, with the last year of data serving as the testing set and the rest of the data serving as the training set. If the data is split randomly as most other situations would have me do, there is a chance that some time periods through the year may not be captured sufficiently by the model(s).

I do recognize, however, that there is a risk of overfitting for the model with a couple years of complete data being used in the training set. To test the model, predictions can be made using data from the features dataset regarding dates, fuel, temperature, and other non-sales numbers. Given the continuous nature of the target variable, scoring metrics of choice should include R2, root mean squared error, and ROC curve and its area under the ROC curve.

The data being used does not contain customer information, information that could hint towards what retail chain this data is about, or anything else of the sort. Therefore, there should not be any ethical implications with this endeavor in predictive analytics.

If this data doesn’t end up working out for this analysis, I could pivot from analyzing data about multiple stores to data from a single store with data about product IDs and other anonymized information, found on Kaggle here: <https://www.kaggle.com/datasets/abdullah0a/retail-sales-data-with-seasonal-trends-and-marketing>. This dataset has around 30K rows and would provide a more granular look into the same phenomenon.

Hopefully the size of this dataset doesn’t prove to be an issue. Most of the data is numerical and with only some categorical data. Some of the categorical data has numerical values, so those will need to be re-categorized as such during data preparation. Afterwards, creating the final dataset will require a couple merges to make sure only one dataset needs to be created. My hope is that merging onto the sales dataset will be the correct course of action. That way, the final dataset will have store, department, weather and other non-sales data, and the sales-oriented data in a single set to be used for model creation. It would also cap the size of the dataset at the size of the sales dataset, 421,570 rows. The last thing needed would be even more than 421K rows of data to feed into any given model.

One concern with these datasets is that they do not all span the same timeframes. The features dataset spans from 2010-2013 while the sales dataset spans from 2010-2012. The solution to this problem is likely to be to only model and predict using data from 2010-2012, with 2012 data being the testing data.

# References

Brownlee, J. (2019, August 21). *A Gentle Introduction to SARIMA for Time Series Forecasting in Python*. Retrieved from Machine Learning Mastery: https://machinelearningmastery.com/sarima-for-time-series-forecasting-in-python/

Justin, L. a. (2022, September 7). *How to build ARIMA models in Python for time series forecasting*. Retrieved from Youtube: https://youtu.be/-aCF0\_wfVwY?si=zB6hYwpk5-N7MPZ4

M. Abdullah, S. Y. (2024). *Retail Sales Data with Seasonal Trends & Marketing*. Retrieved from Kaggle: https://www.kaggle.com/datasets/abdullah0a/retail-sales-data-with-seasonal-trends-and-marketing

*SARIMAX*. (2024, October 3). Retrieved from statsmodels: https://www.statsmodels.org/stable/generated/statsmodels.tsa.statespace.sarimax.SARIMAX.html#statsmodels.tsa.statespace.sarimax.SARIMAX

Singh, M. (2017). *Retail Data Analytics*. Retrieved from Kaggle: https://www.kaggle.com/datasets/manjeetsingh/retaildataset

TeKnowledGeeK. (2021, October 21). *Time Series Forecasting using SARIMAX and compared with ARIMA*. Retrieved from Youtube: https://youtu.be/JO0gFP\_q4uc?si=HAYi1WxAkZWEM3yp