Demand forecasting for items used at Daikin

Problem statement:

Daikin applied is a leading manufacturer of heating and air conditioning devices for residential, commercial, and industrial applications. Each of these devices are made up of various parts. Our problem statement is to build a machine learning model which can look into each of these and forecast its future monthly demand.

Setups:

Extracting data from synapse:

The data can be retrieved using Azure Synapse analytics.

Path: Home -> Azure Synapse Analytics -> daadluatuse2entsw -> Open synapse studio -> Data -> daadluatentsdb (SQL) -> Views -> ora\_ebs.ebs\_mtl\_transactions\_v -> New SQL script

SQL query:

SELECT \* all fields \*

FROM [ora\_ebs].[ebs\_mtl\_transactions\_v]

WHERE ITEM\_NUMBER = '910118733'

AND TRANSACTION\_TYPE\_NAME = 'WIP Issue'

AND ORGANIZATION\_CODE = 'FBO'

I downloaded the data as csv files for further processes which allowed me to work locally as well.

Setting up ML studio:

Create a new Azure ML Studio space or use and existing one.

We created a new space, DAADLMLPOC with the resource group desuse2sioprg.

Running jupyter notebooks on ML studio:

Once the ML studio is launched:

* Create a compute instance
* Upload your data and notebooks under Notebooks
* Start the compute instance and run your code
* Make sure to stop the instance once you are done with the code runs.

Data analysis

We analyzed the data by using different visualization methods such as boxplots, histograms, scatterplots, and correlation chart. The correlation plot showed that the number of units sold had a strong correlation only with the transaction date field.

Removal of null values and unwanted columns:

drop\_columns['ORGANIZATION\_ID','PARTITION\_ID','GL\_PCL','PARENT\_TRANSACTION\_ID','CURRENCY\_CODE','SERIAL\_NUMBERS’, 'TRANSACTION\_SOURCE\_NAME',

'SHIPMENT\_NUMBER','TRANSFER\_TRANSACTION\_ID','TRANSFER\_LOCATOR','LPN',

'TRANSFER\_LPN','TRANSFER\_ORG','TRANSFER\_SUBINVENTORY']

items = items.drop(drop\_columns,axis=1)

unequalColumns = ['DEPARTMENT\_CODE','LOCATOR','SOURCE\_CODE','SOURCE\_LINE\_ID']

items = items.drop(unequalColumns,axis=1)

Data visualization plots:

Boxplots:

Diagram

Description automatically generated with medium confidence

Monthly trend for each item:

Chart, line chart

Description automatically generated

Correlation plot:

A picture containing chart

Description automatically generated

Understanding and using the date field:

The transaction date field could be split into multiple fields such as

* Year
* Month
* Week
* Day
* Is a holiday
* Season

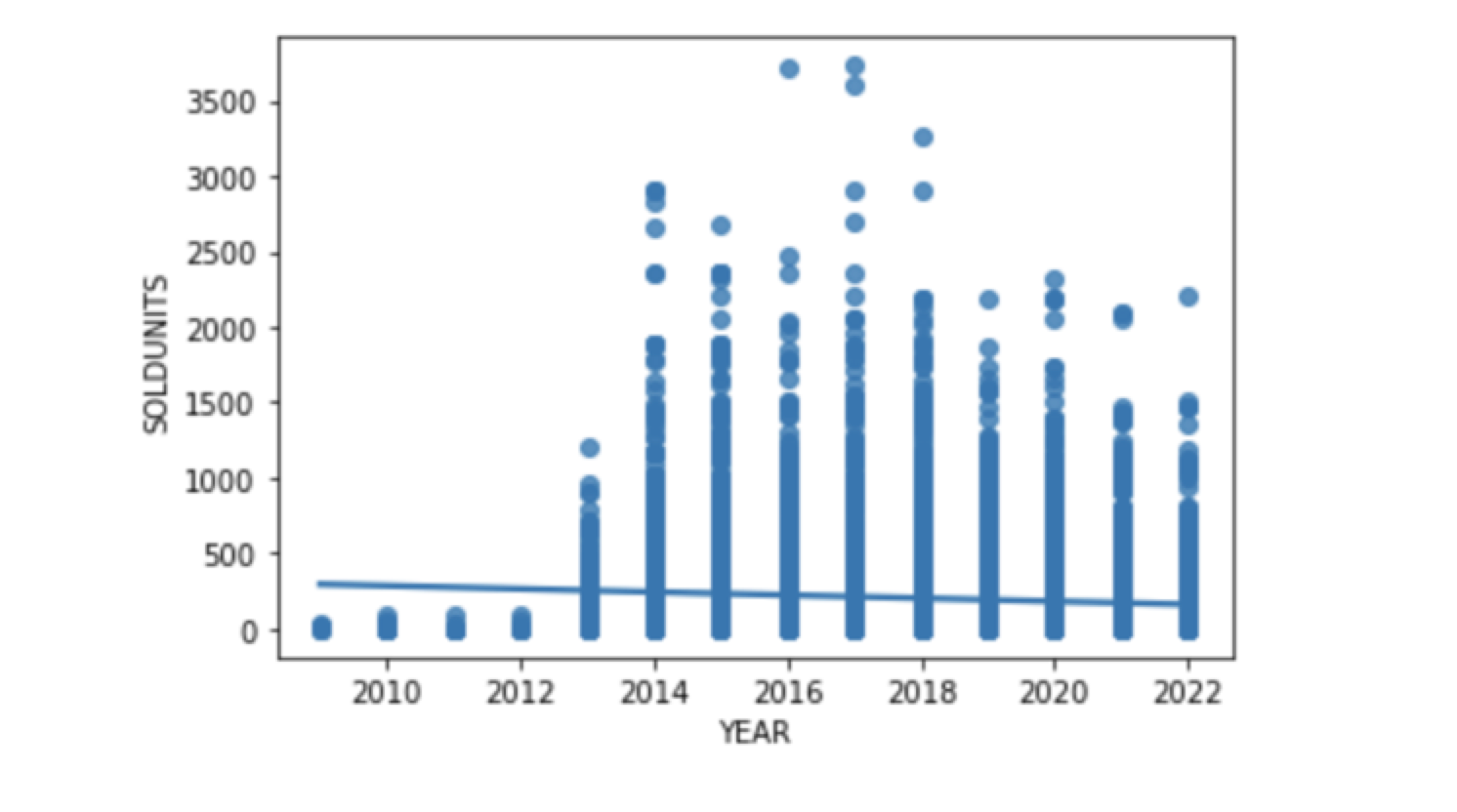
Models explored

Linear Regression model:

The first regression model explored was the linear regression model. A Linear Regression Model describes the relationship between a dependent variable Y and one or more independent variable/s X. The model didn’t turn out to be a good fit for the data we had.

Img1. Good fit Img2.Fit on our data

Chart, scatter chart

Description automatically generated 

Other models explored:

I explored various other models checking the R2 values for each of it such as Random Forest regressor, K Nearest Neighbors and Prophet. Most of them either gave a low R2 score or an undesirable Mean Absolute Error value. Out of these, Prophet did seem like one of the more viable options which could be considered.

Img3. Different models

and their R2 scores

Graphical user interface

Description automatically generated with medium confidence

Light Gradient Boost regressor:

Light Gradient Boosting refers to a class of ensemble machine learning algorithms that can be used for classification or regression predictive modeling problems.

Ensembles are constructed from decision tree models. Trees are added one at a time to the ensemble and fit to correct the prediction errors made by prior models. This is a type of ensemble machine learning model referred to as boosting.

Models are fit using any arbitrary differentiable loss function and gradient descent optimization algorithm. This gives the technique its name, “*gradient boosting*,” as the loss gradient is minimized as the model is fit, much like a neural network.

Application, table, Excel

Description automatically generated

Chart, line chart, histogram

Description automatically generated

The red graph represents the actual data

The green graph represents the predicted data

The X-axis is the index value (The row number in the dataframe)

The Y-axis is the number of units sold.

A line plot sometimes gets confusing as it is not necessary that every point on the line is a data point. So, I decided it to convert it into a scatter plot.

Chart, scatter chart

Description automatically generated

Conclusion:

* We can see that while the results are not perfect, the model is able to predict the trend of the number of items sold.
* As bucketing the values into monthly, weekly or daily buckets messed up the graph and made it difficult it to read, created a table with the actual value and predicted value with the date fields for more readability.
* Looking at the table we see quite a bit of over-prediction which comes from averaging out the number of items sold over the years.

Future work:

* The model can be fine-tuned by training it one year at a time or dropping a few years which are way in the past and had different trend for the item then.
* Predictions can be made for each day of coming months and then a function can be added to sum all the predicted counts to get a monthly count for each of the items.
* Currently, I downloaded the data in csv format from synapse, isolated each item and ran the model one at a time. A data flow needs to be in place to send one item a time through the model.
* A filter can also be added to select the date range for which the prediction is required.

Notes:

* Any R2 score close to 1 is a good R2 score.
* Mean absolute error refers to the magnitude of difference between the prediction of an observation and true value of that observation.

Text

Description automatically generated