Machine\_Learning\_project

(Dataset=”megastore”,random\_state=42)

Team SC\_34

|  |  |  |
| --- | --- | --- |
| Sec 1 | 20191700053 | احمد عوض سيد حسن |
| Sec 2 | 20201701070 | حسان جمال حسان حسني |
| Sec 7 | 20201701136 | محمود نورالدين عبدالله محمد |
| Sec 6 | 20191700810 | محمد بهاء الدين محمد يس |
| Sec 2 | 20201700179 | باسم عاطف السيد محمد |
| Sec 2 | 20191700831 | جابرييل مارك انطون |
| Sec 6 | 20191700500 | محمد أسامة حامد حفني |

# General Analysis:

Throughout the project we used Pandas Library to visualize out dataset as tables and Matplotlib library to visualize attributes as charts.

## DateTime Conversion:

In the dataset there was 2 columns that correspond to datetime “Order Ship” and “Order Date”, By using pandas.todatetime we converted this column to “Order Year: Corresponds to the year the order was placed” and “Order quarter: The quarter of year the order was placed”.

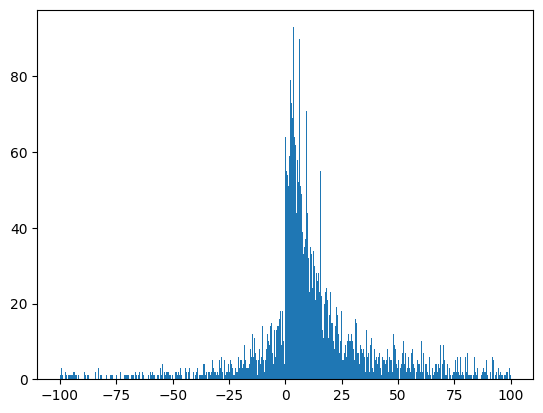
We did the same for “Order Ship”.

## Category-Tree conversion:

Using Regex, we converted the category tree column which corresponds to the format “{'MainCategory': X, 'SubCategory': Y}” We extracted X and Y As 2 columns called ”Main Category” and “Sub Category” ,also we made sure that no entry in the dataset contravene that format.

## Nulls Searching:

Using pandas.isna() function we figured that no nulls were present in the data.Outliers Searching:

Using Z-score, we searched for the outliers in profit and sales, and found out that outliers in profits are outliers in sales, Especially when the correlation of Sales with Profit is well over 55% , also the figure below shows the distribution of Profit which is the same distribution for Sales or at least close to ,so it would be unwise to delete them for now.

# Feature Engineering:

As for the categorical data, We used 3 transformation methods which are:

* transform\_ordinal\_means: Which takes a column and returns the mean of each unique value of this column with the Profit, then map it to the dataframe.
* transform\_ordinal: Which takes a column and return a logical order (1,2,3,etc) of the unique values of the column based on the means of the column with the profit, then map it to the dataframe.
* One\_hot\_encoding: Which creates a column in the dataframe for each unique value in the categorical feature

**The transform functions runs on train data, if any unseen data is given, its value will be 0 or The mean of Data (Depends).**

By using combinations of the 3 methods, we obtained the best input for the model in terms of processing speed and accuracy.

## Anova Test:

We created a function to calculate the p-value for each categorical column in the dataframe, When the p-value is less than 0.05 then that feature has a statistical significance on the dataframe. When running it on some features, it outputs that:

{'State': 3.0788218592530043e-52,

'Segment': 0.16303478233943347,

'Customer Name': 0.022875415355986837,

'Region': 0.008360591767109989,

'Main Category': 3.0370642607455033e-21,

'Sub Category': 8.083173426152449e-147,

'Product ID': 0.0}

So after all, Segment doesn’t have a significant effect on the dataframe, also for product id (“Too many Products”), the “Country” column is dropped because it has only one unique value,

Also we dropped these column because they have no logical meaning "Row ID","Order ID","Customer ID","City","Postal Code","Ship year","Ship quarter","Product Name"

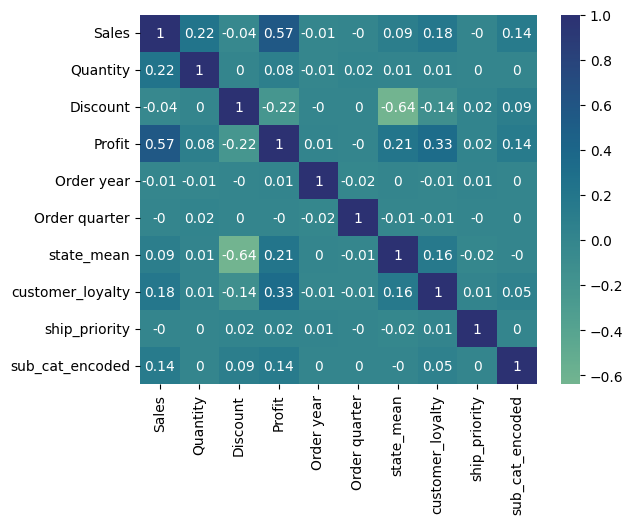
Each of these columns in dropped to prevent dependency and data duplication.

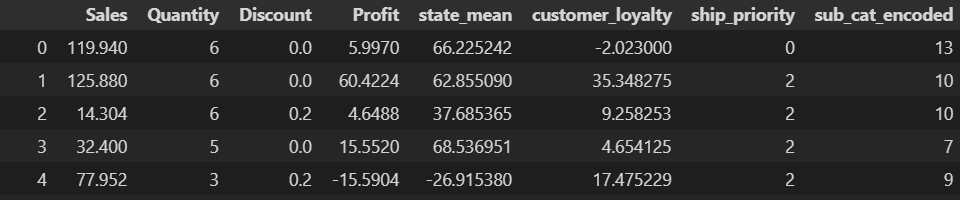


## New Columns:

After using anova, we need to encode the significant columns with the methods mentioned above.

We created several new columns for now, after extracting main and sub categories and order year and quarter As following:

* “state\_mean”: Which carries the mean profit for each state and map it to the dataframe using transform\_ordinal\_means.
* “sub\_cat\_encoded”: Which carries the logical order for each subcategory using transform\_ordinal.
* “ship\_priority”: Which carries the logical order for each Shipping mode using transform\_ordinal.

Now our numerical dataset (Excluding: “Main Category”, ”Region”) looks like

And We can run correlation on each of these columns.

## Feature Selection:

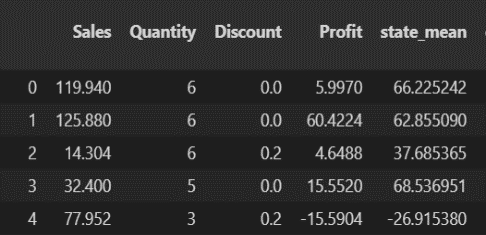
## Correlation:

Using Pearson’s R correlation and by using the help of seaborne, we can create a heatmap of the numerical features as the figure below.

The figure shows the correlation of each feature with any other feature, we can see that customer\_loyalty, state\_mean and Sales have a high correlation with Profit .

So we created a function: feature\_select\_numerical

This function drops any feature with <= 0.05 correlation.

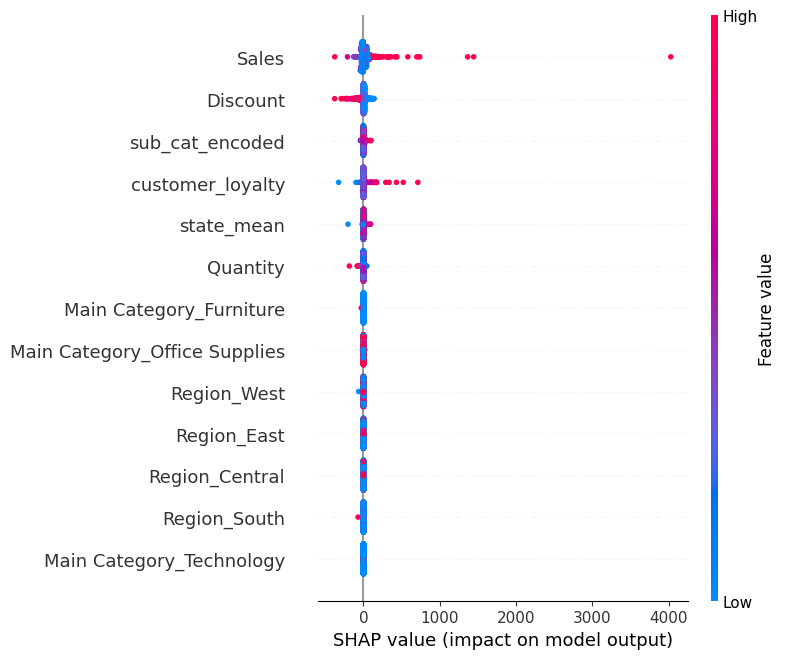
After that we one-hot-encode [“Main Category”, ”Region”] so our final dataframe looks like this:

## SHAP Explainer:

### Option1:

SHAP (SHapley Additive exPlanations) is a model-agnostic technique used for explaining the predictions of machine learning models. It is based on the concept of Shapley values from cooperative game theory. SHAP provides local explanations for individual predictions and global explanations for overall model behavior. Its output “SHAP value” determines whether the feature is impacting the model output or not.

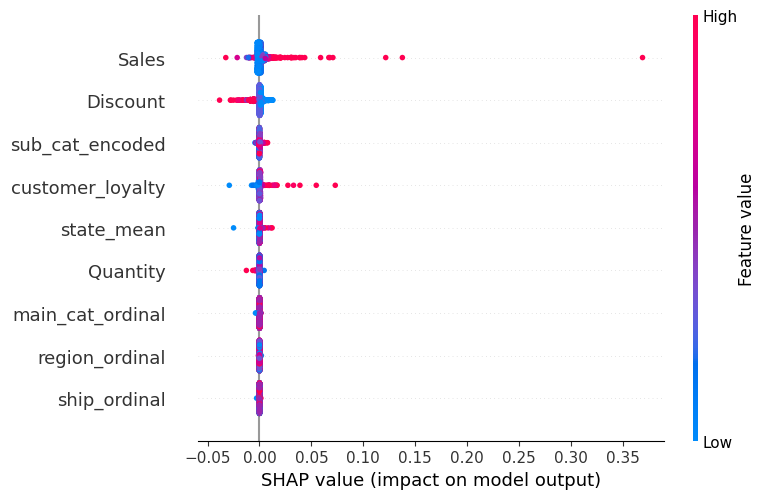
For more insights regarding the concept: https://www.nature.com/articles/s42256-019-0138-9

The shap plot for the given features above outputs this plot.

So using this chart we can see which attributes affecting the model the most. Then we try to change the other features to find the best R2 score.

### Option2:

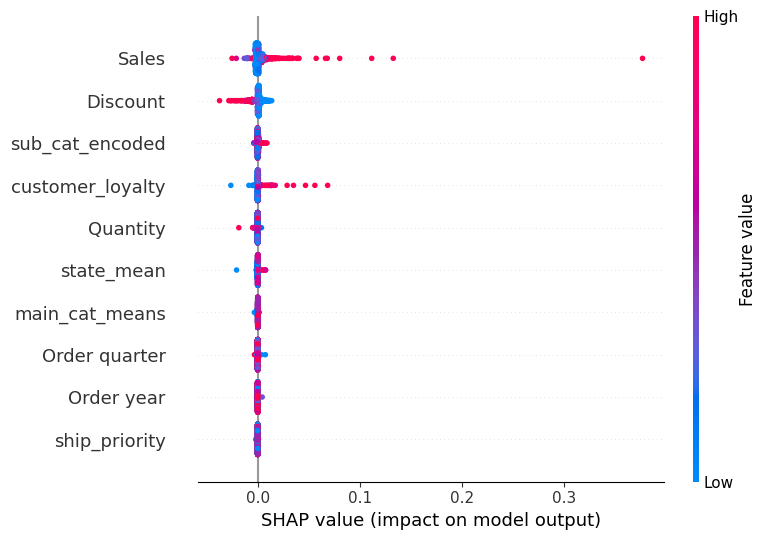
We added ship\_priority again , main\_cat\_encoded which uses transform\_ordinal and region ordinal which also uses transform\_ordinal.

The SHAP chart looks like this is option 2.

So the added columns doesn’t have an impact on the model.

### Option4:

By adding order\_year and Order\_quarter and removing the region\_ordinal.



We also tried to one-hot-encode everything but it gave us the worst score and performance. And it was Option3

**NOTE: Customer\_loyalty Was Removed due to data leakage in all options, The following conclusion does not involve Customer loyalty.**

# Regression:

## Feature Scaling:

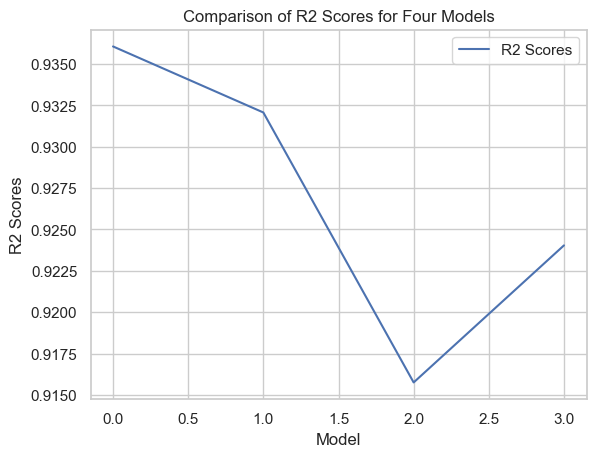
We used MinMaxScaler to scale the features between [1,-1]

Before Using Any Option, We Splatted the Data into train and test, of test size 0.20

## Option1:

Option 1 was the best of the 4 options, We used 4 different algorithms to find the best outcome and score

|  |  |  |
| --- | --- | --- |
| Model | R2 Score | MSE\*e-5 |
| Random Forest | 0.918 | 2.6832 |
| Polynomial Regression | 0.811 | 6.1729 |
| Gradient Boosting | 0.921 | 2.5833 |
| Linear Regression | 0.524 | 15.584 |

The Chart Below Clarifies how the 4 options Performed in the Random forest Algorithm, (I Also Tried other algorithms to these options but none was better that Random Forest)

# Conclusion:

The .zip File Contains 6 Files.

Utils.Py: Containing all the utilities function we Talked about above.

Split.Py: Some of the General Preprocessing plus splitting into 2 datasets (Train-Test)

Train.Py: Reads the Train dataset from Split.py and Applies Preprocessing and saves the models. The PreProcessing is saved in a .pkl file while the scaler and the models are saved in .joblib file. The Preprocessing is the same we used in Option1.

Test.Py: Reads the Test dataset from Split.py and loads the preprocessing and models saved by Train.py.

GeneralAnalysis.ipynb: That Contains a lot of Trial-and-error methods of preprocessing and general charts of the datasets, in addition to the SHAP models we talked about in the previous section.

Preprocessing\_option1.pkl: Contains the saved dictionaries of custom Label-Encoding we talked about in the utility functions.

rf\_option1: Contains a dictionary of [“Model”] and [“Scaler”], The [“Model”] key has a value of {“model\_name”:”model\_variable”} and the scaler key has value of(MinMaxScaler).