IE4211 MODELLING AND ANALYTICS

Group 22

Sales Prediction and Inventory Optimization for 77 Products in an E-Commerce Business

Name of Members:

Fan Xueqi (A0162513N)

Lai Sha (A0192220U)

Yang Ruijia (A0190688M)

He Tianyu (A0177829J)

A PROJECT REPORT SUBMITTED

FOR IE4211 MODELLING AND ANALYTICS

NATIONAL UNIVERSITY OF SINGAPORE

2021

Contents

1. Aim and Objectives

This project aims to develop a suitable Machine Learning model that could predict the sales of different products from different brands. Then, an inventory decision will be proposed for the company and the profit induced by the inventory decision would be calculated.

2. Methodology

The team tackled this problem by firstly exploring the dataset with the Pycaret package, followed by preliminary model selection and data processing. Two performance metrics, mean-squared-error and R-squared were used to evaluate the candidate models.

2.1. Pycaret Initial Data Exploration

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE	TT (Sec)
0	CatBoost Regressor	15.57	1520	37.54	0.8169	0.4728	0.5259	2.087
1	Extra Trees Regressor	17.22	1749	40.66	0.7768	0.5	0.6269	0.3813
2	Gradient Boosting Regressor	17.09	1759	40.22	0.7672	0.5501	0.6471	0.5198
3	Extreme Gradient Boosting	17.37	2027	43.16	0.7549	0.5008	0.5283	1.004
4	Light Gradient Boosting Machine	17.83	2026	43.52	0.7546	0.4947	0.5609	0.1277
5	Random Forest	17.48	1942	42.39	0.7545	0.5013	0.598	0.456
6	Random Sample Consensus	19.94	2330	45.68	0.6853	0.6647	1.002	0.9014
7	Huber Regressor	19.2	2346	45.59	0.6846	0.6144	0.7948	0.0931
8	TheilSen Regressor	21.58	2395	45.54	0.6694	0.7868	1.581	11.72
9	K Neighbors Regressor	21.72	2590	49.53	0.6653	0.5864	0.726	0.0096
10	Bayesian Ridge	21.3	2527	47.29	0.6624	0.6855	1.087	0.026
11	Elastic Net	21.32	2531	47.31	0.662	0.6894	1.1	0.0096
12	Lasso Regression	21.37	2536	47.36	0.6612	0.687	1.101	0.0082
13	Ridge Regression	22.25	2753	48.92	0.6341	0.7653	1.474	0.0056
14	Linear Regression	22.49	2778	49.18	0.6309	0.7842	1.543	0.0167
15	Orthogonal Matching Pursuit	22.43	2792	49.57	0.6275	0.7425	1.286	0.0056
16	Decision Tree	23.04	3212	53.84	0.6069	0.6426	0.7354	0.032
17	Lasso Least Angle Regression	34.77	4189	62.78	0.5008	1.138	2.896	0.0065
18	Passive Aggressive Regressor	24	3649	55.76	0.4885	0.6583	0.7964	0.0128
19	AdaBoost Regressor	49.3	4145	63.53	0.4463	1.462	5.147	0.2844
20	Support Vector Machine	26.37	5164	68.73	0.418	0.6588	0.7933	0.4168

3. Data Processing

3.1. Encoding Categorical Variables

The following four attributes: ‘brandID’,’Attribute1’,’Attribute2’,’Weekday’ were encoded as categorical variables for this study.

3.2. Feature Correlation

The team started feature selection by exploring feature correlations. We want to eliminate the redundant and insignificant attributes thus improving the prediction performance.

The results from correlation heatmap below have showed that **'attribute1', 'attribute2'** have high collinearity with **'avgOriginalUnitPrice'** and **'avgFinalUnitPrice'**, Therefore we decided to introduce a **'discount'** variable to reduce collinearity. The 'discount' variable is calculated by **['discount]=1-(['avgOriginalUnitPrice']-['avgFinalUnitPrice'])/['avgOriginalUnitPrice']**.

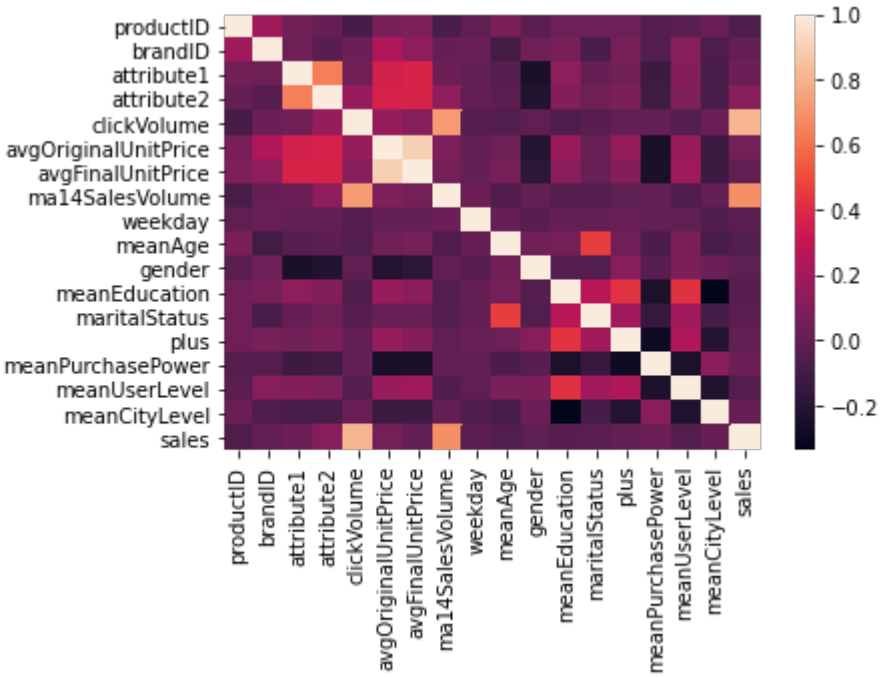

Figure 1. Correlation heatmap

3.3. Data Selection

We used 70% of the dataset as training data and the remaining 30% as the test data.

3.4. Preliminary Model Selection

According to the results given by the Pycaret package, the team decided to use the following four models to generate sales prediction and choose the optimal model from these four models.

- Simple Linear Regression
- Lasso Regression
- Ridge Regression
- Random Forest Regression
- Gradient Boosting Regression
- Support Vector Machine (Kernel = radial/linear/polynomial)

3.5. Performance metrics

Two performance metrics were selected to measure the goodness of fit for all candidate models. The selected performance metrics are: Mean squared error (MSE) and R-squared.

4. Linear Regression

Upon preliminary data processing, all features are used to predict sales using Simple Linear Regression. Standardization on features of numeric values has also been done. This is necessary in obtaining better feature importance illustration as shown in Figure x and y below. Since 'clickVolume' has the highest correlation with sales and in Figure X it is deemed insignificant in terms of coefficient, modelling without standardization is clearly problematic.
Moreover, 'ma14SalesVolume' has a high correlation to 'clickVolume'. In Simple Linear Regression, Lasso Regression and Ridge Regression another variable 'sales14click' has been introduced to represent the interaction between 'ma14SalesVolume' and 'clickVolume' to reduce MSE. It is the multiplication of these two variables.

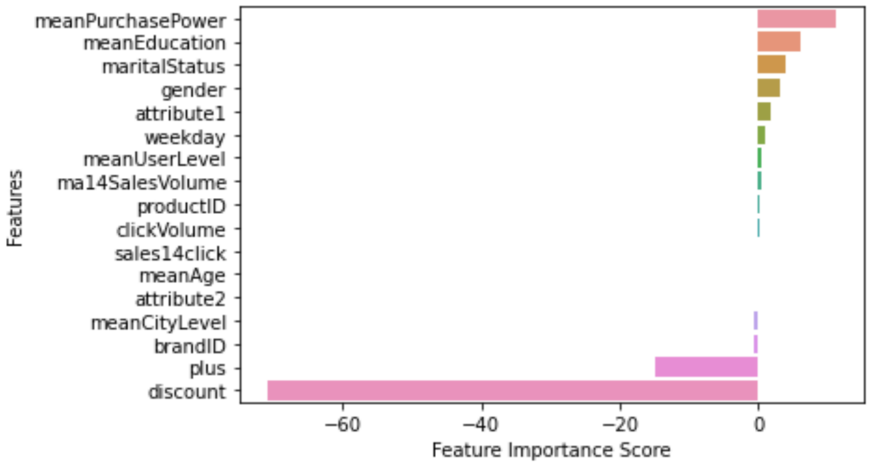

Figure 2. Feature importance without standardization

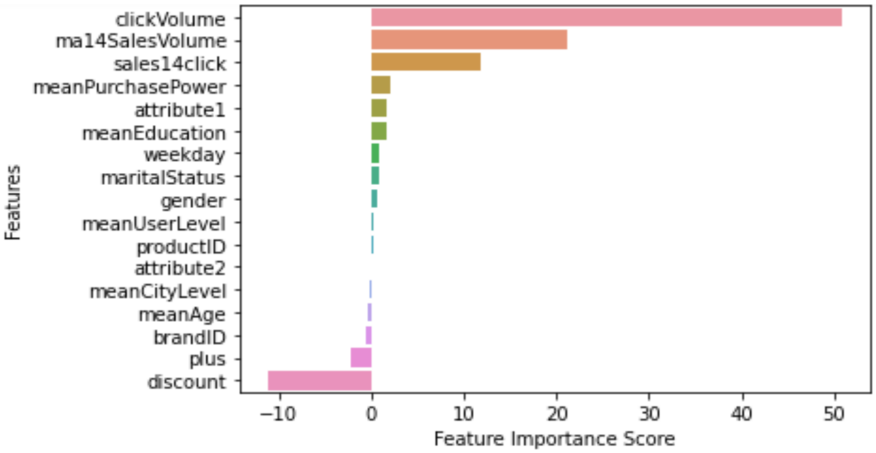

Figure 3. Feature importance with standardization

4.1. Simple Linear Regression

The results of the Simple Linear Regression are shown below.

MSE	Train R-squared	Test R-squared
2012	0.621	0.825

To minimise MSE, we select the following features to project sales and run the model again. They are: ‘sales14click’, ‘clickVolume’, ‘meanEducation’, ‘meanPurchasePower’, ‘meanUserLevel’, ‘meanCityLevel’, ‘discount’, ‘brandID’, ‘weekday’.
The results of the improved model are shown in the table below.

MSE	Train R-squared	Test R-squared
1990	0.621	0.825

4.2. Lasso Regression

Using the same selected features from Simple Linear Regression, we obtain the lowest MSE for Lasso Regression.
Using Cross Validation, the best alpha is selected to be equal to 1.

MSE	Train R-squared	Test R-squared
2027	0.603	0.824

4.3. Ridge Regression

Using all features for Ridge Regression leads to the lowest MSE. The results are shown below.

MSE	Train R-squared	Test R-squared
2072	0.631	0.820

5. Random Forest Regression

Note:

- “productID” is included
- “Discount” is implemented
- ‘Weekday” is modified to reflect weekday or not

5.1. Before tuning: MSE

Before tuning: MSE (without K-fold)	1887
Before tuning: MSE (with K-fold)	2214

5.2. Feature Importance

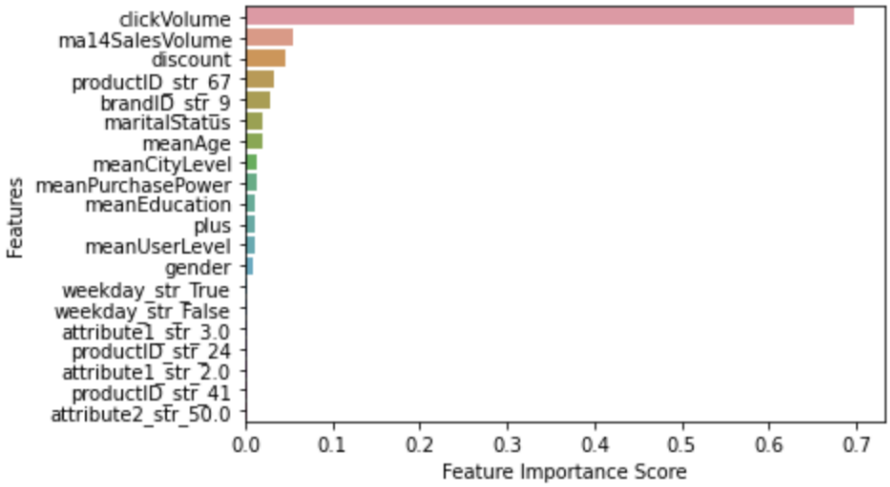

Figure 4. Feature importance of Random Forest

5.3. Feature Selection

Top 10 features were selected and a new model was run with the selected features. 'avgOriginalUnitPrice','avgFinalUnitPrice' are converted to “discount”. “productID”, “brandID” are categorical variables and they are one-hot encoded.

Features used for the new model:

'clickVolume','ma14SalesVolum','discount','productID','brandID','maritalStatus','meanAge','meanCityLevel','meanPurchasePower','meanEducation'

5.4. Parameter Tuning

Tables below summaries the best parameter tuned with and without K-fold validation.

Without K-fold:

max_depth	16
min_samples_leaf	3
n_estimators	100

ccp_alpha	0

With K-fold:

max_depth	15
min_samples_leaf	1
n_estimators	100
ccp_alpha	0

5.5. After tuning: MSE and R-square

	Before tuning	After tuning
MSE (without K-fold)	1887	1779
R-square	0.834	0.846
MSE (with K-fold)	1887	1833
R-square	0.834	0.840

6. Gradient Boosting Regression

6.1. Model Set Up

The gradient boosting model set up follows the overall set up for this project. ‘ProductID’ was excluded to avoid overfitting, new variable ‘Discount’ was introduced; attributes ‘brandID’, ‘Attribute1’, ‘Attribute2’, ‘Weekday’ were encoded as categorical variables. As a result, 53 features were selected for model fitting. Moreover, features 'meanAge','maritalStatus','meanUserLevel' were excluded due to low relevance.

6.2. Feature Importance

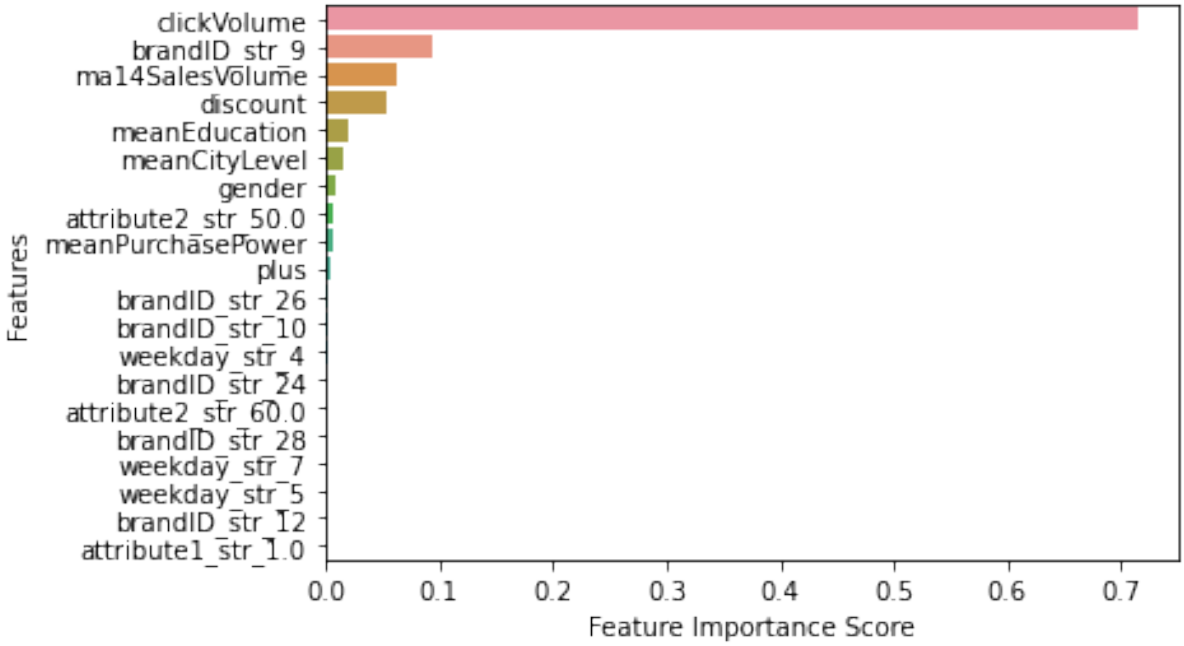

Figure 5. Feature importance of Gradient Boosting Regression

6.3. Parameter Tuning

Learning_rate	0.1
N_estimators	64
Max_depth	5
Max_features	None (the graph does not converge)

6.4. Model after parameter tuning

Best Fit Model	GradientBoostingRegressor(random_state=31,learning_rate=0.1,n_estimators=64,max_depth=5)
After tuning MSE	1400
Final R-squared	0.88

7. Support Vector Machine

In this section, three support vector machine models with linear, radial, and polynomial kernels were analysed.

7.1. Model results summary

We observe that, as standardization parameter C increases, the MSE for each model decreases initially, then it decreases after a certain value. Hence, we evaluate C value as well as other parameters such as degree of polynomial kernel. The best results for each model are pasted below. Noted, for rbf, as C increases, the MSE decreases continuously, this holds true even until C=10000, hence C=100 is chosen empirically.

Model	MSE	R^2
C=54, kernel='linear'	1077.138	0.859
C=100, kernel='rbf'	1344.567	0.824
C=100, kernel='poly', degree=3	1048.187	0.863

7.2. Limitations

It is important to emphasize the fact that the prediction of the Data-test is unsatisfactory for two reasons using SVM. Firstly, there are many negative values in the predicted sales values, the most negative value is -100. Secondly, the highest prediction is over 1000. These considerations make the prediction reliability doubted.

8. Evaluation of Results

	Before tuning		After tuning	
Model	Test MSE	R square	After tuning test MSE	R-squared
Simple Linear Regression	2012	0.825	1990	0.825
Lasso Regression	2027	0.824	NA	NA
Ridge Regression	2072	0.820	NA	NA
Random Forest Regressor	1887	0.834	1779	0.846
Gradient Boosting Regressor	1601	0.831	1400	0.875

Support Vector Machine (Linear/Radial/ Polynomial)	1077 1344 1048	0.859 0.824 0.863	NA	NA

8.1. In-sample Analysis

K-fold Cross Validation was used to find the train and test MSE for each model with their best parameters. The result is summarised in the table above. Even though the K-fold CV method gives a higher test MSE as compared to the train_test_split (train_size=0.7), the lowest MSE for both ways of calculating MSE is still Gradient Boosting Regressor. Therefore, Gradient Boosting Regressor with the tuned parameters is the final model for prediction.

9. Inventory Decisions

With the best model we have, **GradientBoostingRegressor(random_state=31,learning_rate=0.1,n_estimators=64,max_depth=5),** sales for the Data-test.csv is predicted. The sales is assumed to follow an exponential distribution. The inventory decisions are made by following the prediction methods taught in the lecture.

Assume the sales will follow exponential distribuion

In [23]:
```
Price=20
Cost=12
salvage=8
Over=Cost-salvage
Under=Price-Cost
```

In [43]:
```
inventory=np.rint(-pred_sales * np.log(1 - Under/(Under + Over)))
inventory
```

Out[43]:
```
array([ 68.,  26.,  13., ..., 268.,  16.,  24.])
```

10. Out-of-sample Profit

Profit is calculated by the following equation

$$\sum sales \times (Price - Cost) - \sum (inventory - sales) \times (Cost - Salvage)$$

The final out-of-sample profit = $677,148