

Round 01 – Kaggle Competition

DataStorm 3.0

Sales Predictor

THE TEAM
DECISION_MAKERS

Kaggle Username - **ds22109**

Kaggle Display name - **ds22-109**

Kajanan Selvanesan

Laksika Tharmalingam

Ashokkumar Susitharan

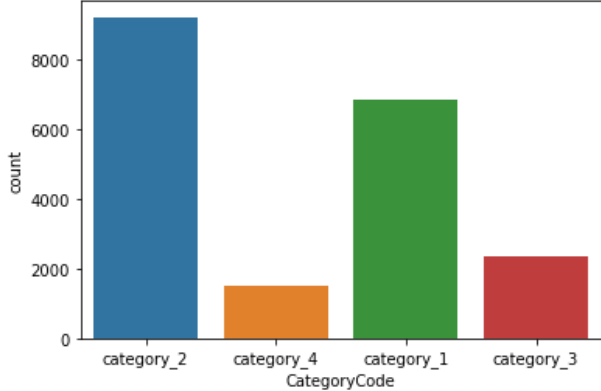
GitHub: <https://github.com/kajanan1212/Sales-Predictor>

Lowest Total MAPE: 60.77127

[Based on The Kaggle Scores]

Dataset: Training [shape (19921, 4)]

Step 01: Descriptive Analysis

Column	Description
CategoryCode	It indicates the category of an item. There are 4 categories. Number of items belongs to each category: 
ItemCode	This is a unique code of an item. There are 194 items.
DateID	It indicates the sold date of an item. Start: 1 st October 2021 End: 13 th February 2022
DailySales	Daily sold count for sold items. <pre>count 19921.000000 mean 7.351890 std 14.605342 min 1.000000 25% 2.000000 50% 3.000000 75% 7.000000 max 434.000000 Name: DailySales, dtype: float64</pre>

Dataset: Validation [shape (370, 4)]

Similar to the Train dataset. But the main difference is that Week and WeeklySales have been included instead of DateID and DailySales.

Dataset: Test [shape (377, 4)]

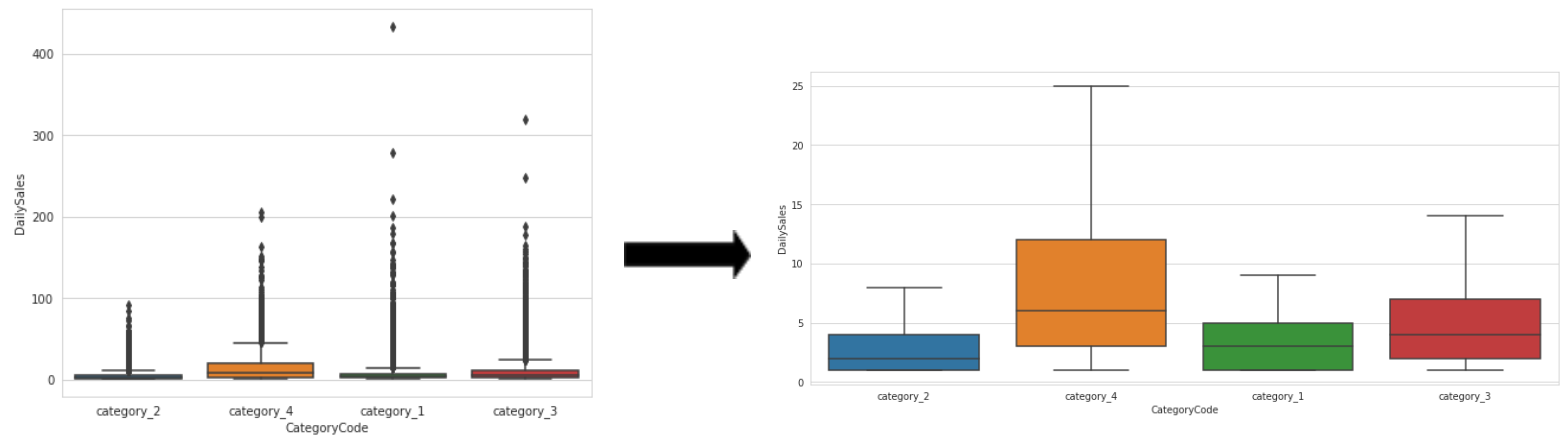
Similar to the Validation dataset. But there aren't any values for WeeklySales. We have to predict them.

Step 02: Data Cleaning

1. Missing Values Handling (There is no any missing values)

2. Outliers Handling

We recursively removed the outliers in each category until we got a training dataset with no outliers in any categories.



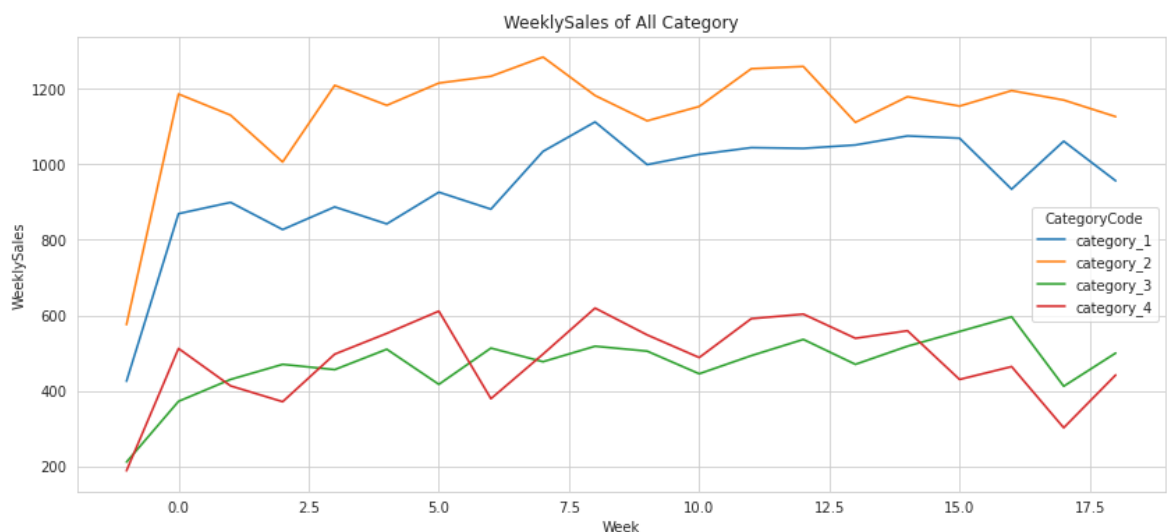
Step 03: Exploratory Analysis

We plotted graphs for each category and item to see how the DailySales had been changed with Date. From the graphs, we couldn't come to a better conclusion. And we found mutual information between ItemCode and DailySales. It was 0.24212.

Step 04: Feature Engineering

1. Feature Transformation

In the training dataset, we converted the DateID into the number of passed weeks from the date 04-10-2021. In the other two dataset, we did the same thing for the Week attribute. And in the training dataset, we had calculated the WeeklySales from the DailySales with the help of a newly created Week column. Below graph is based on the training dataset.



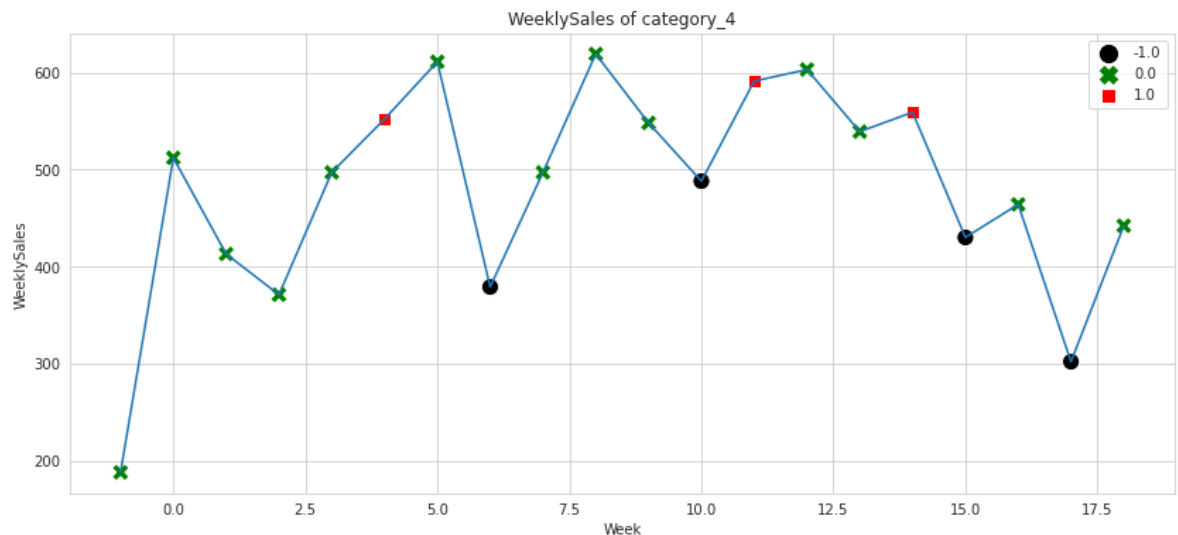
2. Feature Creation

We created features like Phase and Weight. We believed they would be useful and meaningful features for the given domain.

Phase: Start of a month(MS), First middle of a month(MM1), Second middle of a month(MM2) and End of a month(ME).

Weight: Week has holidays => -1, Normal week => 0 and Week has festivals => 1.

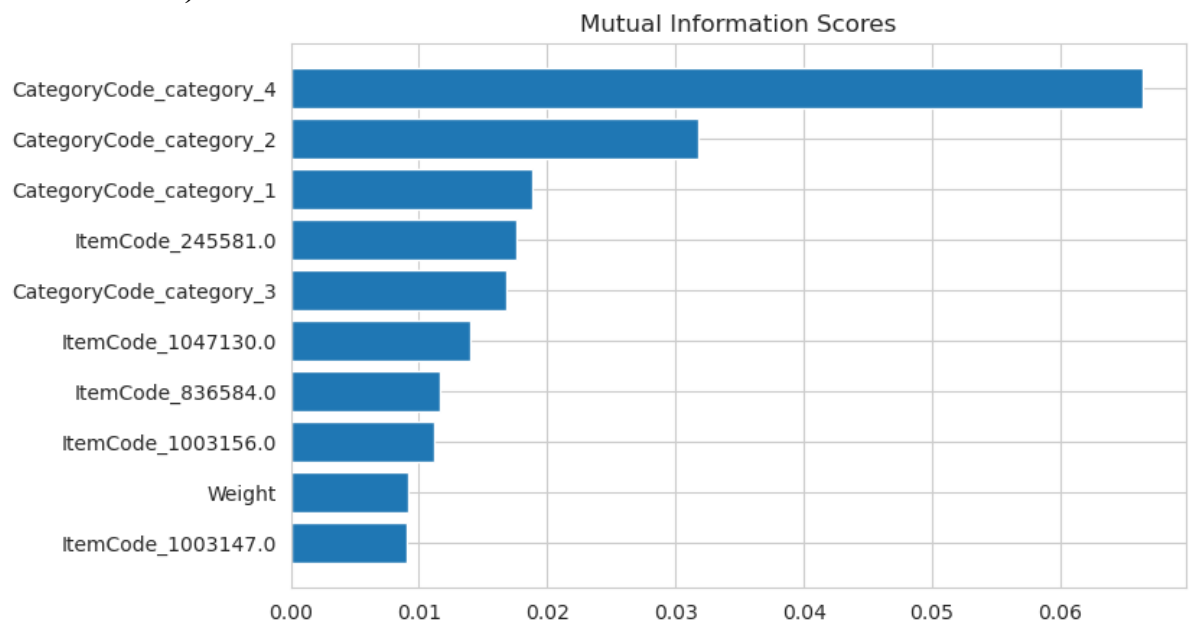
After the creation, we noticed that specific category items have relatively low WeeklySales on the holidays and relatively higher WeeklySales on the festival's time.



3. Feature Encoding

We encoded the features such as CategoryCode, ItemCode, Phase by using *One Hot Encoding* technique.

4. Feature Selection Based on Feature Importance(Using Mutual Information)



We selected the features which have a mutual information score greater than 0.0001. In this way, we could reduce the dimension of our encoded dataset.

Step 05: Evaluate The Models on Training Dataset Using Cross-Validation for Model Selection

Initially selected models for this project are *LinearRegression*, *KNeighborsRegressor*, *DecisionTreeRegressor*, *RandomForestRegressor*, and *SVC* because these are pretty good for regression problems, simplest models, and interpretable.

To select the best model from the above list, we did the *5-Fold Cross-Validation* (We used *MAPE* scoring within the Cross-Validation) to our encoded training dataset. The results are shown here.

We selected the *Random Forest Regressor* as our best model based on the results.

	Model	ScoreMean	Score Standard Deviation
3	RandomForestRegressor	5.445308e+01	7.130853e+00
2	DecisionTreeRegressor	5.510833e+01	7.239937e+00
1	KNeighborsRegressor	5.574691e+01	1.861969e+00
4	SVC	5.856447e+01	8.343438e+00
0	LinearRegression	2.151169e+14	5.264901e+13

Step 06: Tuning for The RandomForestRegressor

We tried to tune the *n_estimators* and *max_depth* of *RandomForestRegressor* using *GridSearchCV*. Best two are shown here.

	param_n_estimators	param_max_depth	mean_test_score	std_test_score
5	1	21	50.646327	7.182567
12	41	41	50.871124	7.322822

Therefore we had found our best model with the best hyperparameters. That was *RandomForestRegressor(random_state=42, n_estimators=1, max_depth=21)* [We used 42 as *random_state* wherever we needed to maintain the consistency of the results.]

From these observations, we created two *BaggingRegressor* which have *RandomForestRegressor* with the difference hyperparameters(*[n_estimators=1, max_depth=21]*, *[n_estimators=1, max_depth=41]*) as their *base_estimator*. We tried to tune the *n_estimators* of the both *BaggingRegressor* using *GridSearchCV*. Best results are shown here.

1	21			
	param_n_estimators	mean_test_score	std_test_score	
1	21	50.880203	6.914308	
3	61	50.949356	6.926332	
1	41			
	param_n_estimators	mean_test_score	std_test_score	
2	41	50.620240	7.367974	
1	21	50.709402	7.432198	

Therefore we had found our another best model with the best hyperparameters. That was *BaggingRegressor(base_estimator=RandomForestRegressor(random_state=42, n_estimators=1, max_depth=41), random_state=rs, n_estimators=41)*.

Step 07: Best Model Selection

For this purpose, we used the validation dataset. First we trained our two best models using the entire training dataset. Then we validate our trained models by giving unseen data(validation dataset) to our trained models. Here, we used the *MAPE* score to validate. And also we did the *Cross-Validation* to the dataset which was a combination of the training and validation datasets. Based on both results, eventually we came with one best model which was *BaggingRegressor* (*base_estimator=RandomForestRegressor(random_state=42, n_estimators=1, max_depth=41),random_state=rs,n_estimators=41*).

	Model	MAPE_score		Model	ScoreMean	Score Standard Deviation
1	BaggingRegressor	68.370056	1	BaggingRegressor	47.113977	6.014527
0	RandomForestRegressor	69.256885	0	RandomForestRegressor	51.084091	4.221496

Step 08: In Test Dataset, Predict The Values for The WeeklySales

We had trained our best one using a combination of both the training and validation datasets. We used that model to predict the values.

Discuss any additional attributes that the retail management should start collecting?

- ☐ Were there any festivals during the recorded weeks/days?
- ☐ Were there any holidays during the recorded weeks/days?
- ☐ Were there any offers/discounts during the recorded weeks/days?
- ☐ What was the weather condition during the recorded weeks/days?

Based on the results you achieved, list interventions that the management team can take as next steps?

- ☐ You won't need to maintain a lot of category 4 items in your warehouse when there will be any holidays in the upcoming weeks.