Global Journal of Management and Business Research: A Administration and Management
Volume 23 Issue 8 Version 1.0 Year 2023
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals
Online ISSN: 2249-4588 & Print ISSN: 0975-5853

Enhancing Demand Forecasting in Retail Supply Chains: A Machine Learning Regression Approach

By Tewogbade Shakir & Akinlose Modupe

Abstract- This investigation discusses the importance of supply chain management and the role of demand forecasting in the business circle and presents a review of literature on demand forecasting techniques, emphasizing the shift from traditional methods to more sophisticated statistical and machine learning approaches.

The study aims to contribute to existing knowledge on demand forecasting by utilizing machine learning regressors to predict orders in a Brazilian logistics company. It showed the use of the PyCaret Python library to develop robust regression models and validate key contributing features through feature importance plots. The performance of eighteen models, including Ridge, LASSO, XGBoost, Bayesian Ridge, Linear Regression, Gradient Boosting, KNN, Random Forest, among others, is evaluated using the Mean Absolute Error (MAE) metric.

In order to make accurate predictions, feature important plot was used to validate features that have a strong positive correlation with the target variable, such as non-urgent orders, urgent orders, and orders from the traffic controller sector. The performance of different regression models, including Linear Regression, Lasso Regression, Ridge Regression, Lasso Least Angle Regression, Bayesian Ridge, and Elastic Net were discussed.

GJMBR-A Classification: *LCC: HD30.28*

Strictly as per the compliance and regulations of:

Enhancing Demand Forecasting in Retail Supply Chains: A Machine Learning Regression Approach

Tewogbade Shakir [α] & Akinlose Modupe [σ]

Abstract- This investigation discusses the importance of supply chain management and the role of demand forecasting in the business circle and presents a review of literature on demand forecasting techniques, emphasizing the shift from traditional methods to more sophisticated statistical and machine learning approaches.

The study aims to contribute to existing knowledge on demand forecasting by utilizing machine learning regressors to predict orders in a Brazilian logistics company. It showed theuse of the PyCaret Python library to develop robust regression models and validate key contributing features through feature importance plots. The performance of eighteen models, including Ridge, LASSO, XGBoost, Bayesian Ridge, Linear Regression, Gradient Boosting, KNN, Random Forest, among others, is evaluated using the Mean Absolute Error (MAE) metric.

In order to make accurate predictions, feature important plot was used to validate featuresthat have a strong positive correlation with the target variable, such as non-urgent orders, urgent orders, and orders from the traffic controller sector. The performance of different regression models, including Linear Regression, Lasso Regression, Ridge Regression, Lasso Least Angle Regression, Bayesian Ridge, and Elastic Net were discussed. The linear regression model is identified as the best model for predicting the weekly order volume basedon the given features.

I. Introduction

According to Chopra and Meindil 2011, supply chain is a system that consists of a manufacturers, suppliers, transporters, distributors and retailers; where they all share information and goods and services are delivered among them to meet up with the demand ofconsumers. Considering the set of entities involved, supply chain is considered as a major aspect in the business circle. It is a whole cycle of procedures and end-to-end flow of materials, information and funds starting from the product design stage to fulfilment. As noted by Casson 2013, economics factors are the key factor in supply chain as it shows greater impact on size and shape. Beyond economic factors, Ketchen and Giunipero 2004 highlighted the effect of strategic choices on supply chain set up, coordination and operation.To deliver best and optimal solution, supply chain activities must be strategically laid out, coordinated and handled. Proper coordination will emanate from balancing the demand and supply ends. Various variables such as weather, market trends, and season, consumer behaviour exist between the two ends and thus coordinate planning must be fashioned out. The volatility in these variables prompt for demand forecasting. As mentioned by Hope and Fraser 2003, demand forecasting is one of the prerequisites for strategical planning in supply chain management. Many techniques and methods have been used to investigate contributing variables and forecasting of orders based on historical data. In this study machine learning regressors will be used to make order predictions using dataset acquired on the field for 60 days in a Brazilian logistics company. Machine learning is a sophisticated methods posit by advancement in computing technology where historical data are trained to build models which are subsequently used to make predictions. It exists as classification or regression means. For regression, predictions are made for continuous variables like demand in supply chain. According to Al-Jarrah et al 2015, machine learning is perfectly efficient in the analysis of voluminous data that exist in demand forecasting study. To effect more accurate demand prediction, robust regressors models will be developed using Python Library PyCaretwhile key contributing features are validated through feature importance plot. The various machine learning models (Ridge, LASSO, XGBoost, Bayesian Ridge, Linear Regression, Gradient Boosting, KNN, Random Forest and others) will run concurrently and their prediction performance measured with Mean Absolute Error (MAE): the mean absolute differences between predicted values and actual demand values. The lesser the MAE, the better the model predictions.

II. Literature Review

One of the important steps in supply chain planning is demand forecasting. By standard forecasting includes element of uncertainty as it is often hard to predict perfectly in a consistent manner. Thus, because of volatility involved in demand and uncertainty in forecasting, many research have been carried out over the years to improve decision making in supply chain. Carbonneau, Laframboise and Vahidor 2008 classified naïve method, average method, moving average and trend as traditional methods and viewed them as solution approach based on linear demand idealization. Due to emerging complexity in demand situations in supply chain field, non-linear techniques were proposed in their paper to handle the complexity. They made used of machine learning techniques-neural

Author α σ: e-mail: pingcommercial@gmail.com

network, recurrent neural networks and SVM to predict future demand and confirmed better outcomes compared to the traditional methods. MAE was used to compute error analysis for their model predictors in order to evaluate which among them best fits the study dataset. The results revealed that recurrent neural networks and SVM yielded most accurate predictions. A similar investigation by Wang 2012 using large paper enterprise dataset showed that support vector regression produced excellent performance. A more comprehensive study is seen in the work of Rivera-Castro et al 2019 where their experiments was performed using thirteen techniques that cut across times series and machine learning (Adaboost, ARIMAX, ARIMA, Bayesian Structure Time Series (BSTS), Bayesian Structural Time Series with a Bayesian Classifier (BSTS Classifier), Ensemble of Gradient Boosting, Ridge Regression, Kernel Regression, LASSO, Neural Network, Poisson Regression, Random Forest and Support Vector Regression). Symmetric Mean Absolute Percent Error (SMAPE) was used to evaluatethe model's performance and Adaboost yielded best model with SMAPE of 0.17 followed byensemble of Random Forest 0.18. Review of literatures has shown paradigm shift from traditional approach (Historical Analogy, Scenario Planning, Moving Averages, Exponential Smoothing and others) of predicting demand to more sophisticated tools of statistical methods and machine learning. In order to improve existing knowledge on demand forecasting, this study will follow the path of Rivera-Castro et al 2019 by making using of advanced Python Library that yield prediction across many regressions' algorithms simultaneously. Pycaret is an open-source machine learning library and it is low-code. It is distinct in its ability to train and evaluate multiple models with few line of codes. Another improvement is mode of error analysis which is run through MAE, MSE, RMSE, RMSLE and MAPE for each of the models. Thus, there is opportunity for utilization of many metricsto gauge prediction performances.

III. Methodology

a) *Data Pre-processing*

The dataset utilized for this investigation is from UCI repository Daily Demand Forecasting Orders - UCI Machine Learning Repository. It was acquired on the field by a Brazilian Logistic company for a period of sixty days. The attribute of the dataset was shown using Pandas library at pre-processing stage.

```
import pandas as pd
```

```
df = pd.read_excel('supply chain.xlsx')
```

```
df.head()
```

	Week of the month (first week, second, third, fourth or fifth week	Day of the week (Monday to Friday)	Non-urgent order	Urgent order	Order type A	Order type B	Order type C	Fiscal sector orders	Orders from the traffic controller sector	Banking orders (1)	Banking orders (2)	Banking orders (3)	Target (Total orders)
0	1	4	316.307	223.270	61.543	175.586	302.448	0.000	65556	44914	188411	14793	539.577
1	1	5	128.633	96.042	38.058	56.037	130.580	0.000	40419	21399	89461	7679	224.675
2	1	6	43.651	84.375	21.826	25.125	82.461	1.386	11992	3452	21305	14947	129.412
3	2	2	171.297	127.667	41.542	113.294	162.284	18.156	49971	33703	69054	18423	317.120
4	2	3	90.532	113.526	37.679	56.618	116.220	6.459	48534	19646	16411	20257	210.517

Figure 1: Importing Data using Pandas

The dataset was validated for occurrence of missing values in the coding environment.

```
# Checking for missing values
df.isnull().sum()
```

```
Week of the month (first week, second, third, fourth or fifth week    0
Day of the week (Monday to Friday)                                    0
Non-urgent order                                                      0
Urgent order                                                          0
Order type A                                                          0
Order type B                                                          0
Order type C                                                          0
Fiscal sector orders                                                  0
Orders from the traffic controller sector                             0
Banking orders (1)                                                    0
Banking orders (2)                                                    0
Banking orders (3)                                                    0
Target (Total orders)                                                 0
dtype: int64
```

Figure 2: Checking for Missing Values

Summary statistics of the dataset was evaluated to understand the distribution of thevariables. This provides insights relevant to our analysis.

```
# Summary statistics
df.describe()
```

	Week of the month (first week, second, third, fourth or fifth week	Day of the week (Monday to Friday)	Non-urgent order	Urgent order	Order type A	Order type B	Order type C	Fiscal sector orders	Orders from the traffic controller sector	Banking orders (1)	Banking orders (2)	Banking orders (3)	
count	60.000000	60.000000	60.000000	60.000000	60.000000	60.000000	60.000000	60.000000	60.000000	60.000000	60.000000	60.000000	6
mean	3.016667	4.033333	172.554933	118.920850	52.112217	109.229850	139.531250	77.396133	44504.350000	46640.833333	79401.483333	23114.633333	30
std	1.282102	1.401775	69.505788	27.170929	18.829911	50.741388	41.442932	186.502470	12197.905134	45220.736293	40504.420041	13148.039829	8
min	1.000000	2.000000	43.651000	77.371000	21.826000	25.125000	74.372000	0.000000	11992.000000	3452.000000	16411.000000	7679.000000	12
25%	2.000000	3.000000	125.348000	100.888000	39.456250	74.916250	113.632250	1.243250	34994.250000	20130.000000	50680.500000	12609.750000	23
50%	3.000000	4.000000	151.062500	113.114500	47.166500	99.482000	127.990000	7.831500	44312.000000	32527.500000	67181.000000	18011.500000	28
75%	4.000000	5.000000	194.606500	132.108250	58.463750	132.171000	160.107500	20.360750	52111.750000	45118.750000	94787.750000	31047.750000	33
max	5.000000	6.000000	435.304000	223.270000	118.178000	267.342000	302.448000	865.000000	71772.000000	210508.000000	188411.000000	73839.000000	61

Figure 3: Summary Statistics for Dataset Attributes

The average number of non-urgent orders is around 172.55, with a standard deviation of 69.50. The minimum number of non-urgent orders is 43.65, and the maximum is 435.30. The average number of urgent orders is around 118.92, with a standard deviation of 27.17. The minimum number of urgent orders is 77.37, and the maximum is 223.27. The average number of orders from the fiscal sector is around 77.39, with a standard deviation of 186.50. The minimum number of orders from the fiscal sector is 0, and the maximum is 865. The average number of orders from the traffic controller sector is around 44504.3, with a standard deviation of 12197.9. The minimum number of orders from the traffic controller sector is 11992, and the maximum is 71772.

b) Exploratory Analysis

To visualize the correlation between the dataset variables correlation matrix was employed. This will aid understanding of the relationships between different features and the target variable. The heatmap shows the correlation between different variables in the dataset. The colour scale represents the correlation coefficient, with dark blue indicating a strong negative correlation, light blue to light red indicating little to no correlation, and dark red indicating a strong positive correlation.

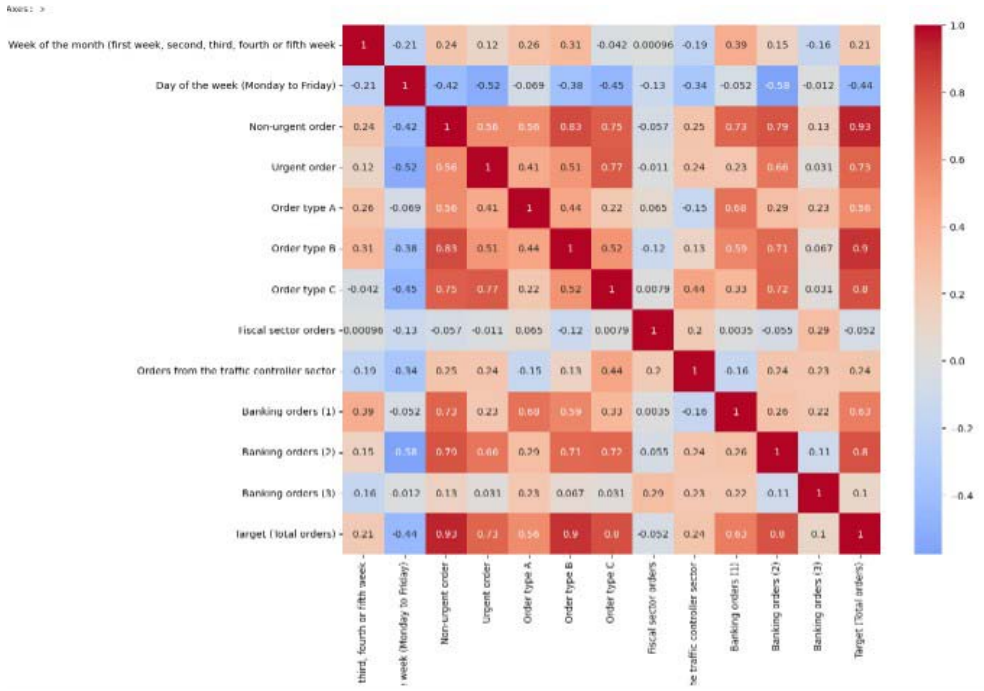


Figure 4: Heatmap to Show Correlation among Features

From the heatmap, we can see that non-urgent order, urgent order, order type A, order type B, order type C, and orders from the traffic controller sector have a strong positive correlation with Target (Total orders). This suggests that these variables will be good predictors for the total orders.

c) Machine Learning Predictions

i. *Linear Regression*

We started off from modelling weekly order volume with linear regression. Then make use of other regression models (Decision Tree and Random Forest) and compare their performances. The whole dataset is split into a training set and a test set in ratio 80:20.

```
from sklearn.model_selection import train_test_split

# Defining the features and target variable
X = df.drop('Target (Total orders)', axis=1)
y = df['Target (Total orders)']

# Splitting the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figure 5: Splitting Dataset to Training and Test Sets

Linear regression remits a linear relationship between the features and the target variable. The linear regressor is trained with training set while its performance is evaluated using the test set.

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Creating a Linear Regression model
lr = LinearRegression()

# Training the model
lr.fit(X_train, y_train)

# Making predictions on the test data
y_pred_lr = lr.predict(X_test)

# Evaluating the model
mse_lr = mean_squared_error(y_test, y_pred_lr)
r2_lr = r2_score(y_test, y_pred_lr)

mse_lr, r2_lr
```

```
(4.668373655643919e-24, 1.0)
```

Figure 6: Coding for Linear Regression

The linear regression model has a Mean Squared Error (MSE) of approximately 4.24e-24 and an R-squared (R^2) score of 1.0 on the test data. The R^2 score is a statistical measure that represents the proportion of the variance for a dependent variable that's explained by an independent variable(s) in a regression model. An R^2 score of 1.0 indicates that the model explains all the variability of the response data around its mean, which is a perfect score. However, a perfect score is suspicious and could be a sign of overfitting, where the model has learned the training data too well and may not perform well on new, novel data.

ii. *Decision Trees Regressor*

Decision trees are a type of model used for both lassification and regression. Trees responds to sequential questions which send us down a specific route of the tree. The model gets to produce decision for each feature, where each decision leads to a new question (decision) until a prediction is executed.

```
from sklearn.tree import DecisionTreeRegressor

# Creating a Decision Tree Regressor model
dt = DecisionTreeRegressor(random_state=42)

# Training the model
dt.fit(X_train, y_train)

# Making predictions on the test data
y_pred_dt = dt.predict(X_test)

# Evaluating the model
mse_dt = mean_squared_error(y_test, y_pred_dt)
r2_dt = r2_score(y_test, y_pred_dt)

mse_dt, r2_dt
```

```
(5332.670754499999, 0.47899695889204263)
```

Figure 7: Coding for Decision Tree Regression

The Decision Tree Regressor model has a Mean Squared Error (MSE) of approximately 5332.67 and an R-squared (R^2) score of 0.48 on the test data. This model's performance is significantly lower than the linear regression model. The R^2 score of 0.48 indicates that themodel explains about 48% of the variability in the target variable.

iii. *Random Forest Regressor*

Random Forest is a type of ensemble learning method, where a group of weak models come together to form a strong model. In Random Forest, we grow multiple trees as opposed to a single tree. To classify a new object based on attributes, each tree gives a classification. The forest chooses the classification having the most votes (over all the trees in the forest) and incase of regression, it takes the average of outputs by different trees.

```
from sklearn.ensemble import RandomForestRegressor

# Creating a Random Forest Regressor model
rf = RandomForestRegressor(random_state=42)

# Training the model
rf.fit(X_train, y_train)

# Making predictions on the test data
y_pred_rf = rf.predict(X_test)

# Evaluating the model
mse_rf = mean_squared_error(y_test, y_pred_rf)
r2_rf = r2_score(y_test, y_pred_rf)

mse_rf, r2_rf
```

```
(3301.721138390172, 0.677421158517971)
```

Figure 8: Coding for Random Forest Regression

The Random Forest Regressor model has a Mean Squared Error (MSE) of approximately 3301.72 and an R-squared (R^2) score of 0.68 on the test data. This model's performance is also lower than the linear regression model but greater than that of decision tree. The R^2 score of 0.68 indicates that the model explains about 68% of the variability in the target variable.

d) Combined Regressors by PyCaret

PyCaret is a very robust Python library that will be used to compare different regression models and find the best one for our dataset. Firstly, regression module is set up in PyCaretwith our dataset. Then, the *compare_models* function is used to compare different models. This function trains all models in the model

library using cross-validation and evaluates performance metrics for regression, such as MAE, MSE, RMSE, R2, RMSLE, and MAPE. Another merit from this approach is that PyCaret library takes care of all the pre-processing steps such as missing value imputation, encoding categorical variables, feature scaling and normalization. This is shown as TRUE in figure 9 below.

```
from pycaret.regression import *

# Initialize setup
s = setup(df, target = 'Target (Total orders)', session_id = 123)
```

	Description	Value
0	Session id	123
1	Target	Target (Total orders)
2	Target type	Regression
3	Original data shape	(60, 13)
4	Transformed data shape	(60, 13)
5	Transformed train set shape	(42, 13)
6	Transformed test set shape	(18, 13)
7	Numeric features	12
8	Preprocess	True
9	Imputation type	simple
10	Numeric imputation	mean
11	Categorical imputation	mode
12	Fold Generator	KFold
13	Fold Number	10
14	CPU Jobs	-1
15	Use GPU	False
16	Log Experiment	False
17	Experiment Name	reg-default-name
18	USI	ebc3

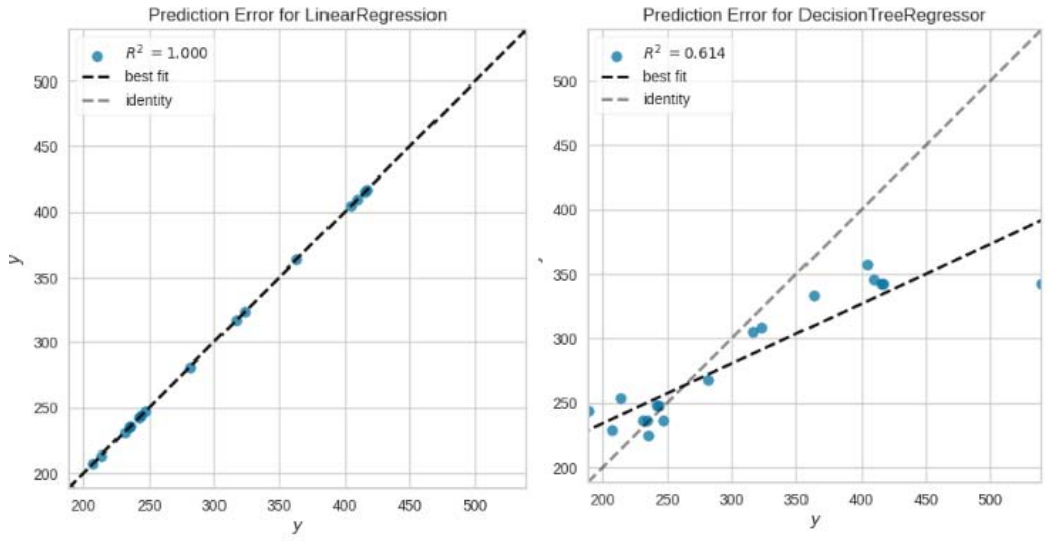
Figure 9: Initializing PyCaret Regression

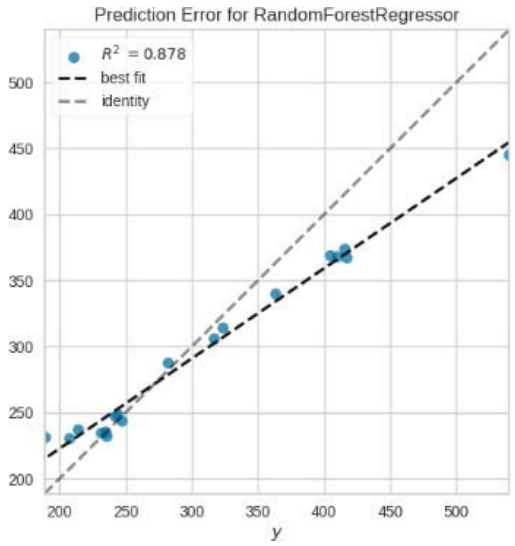
```
# Comparing all models
best_model = compare_models()
```

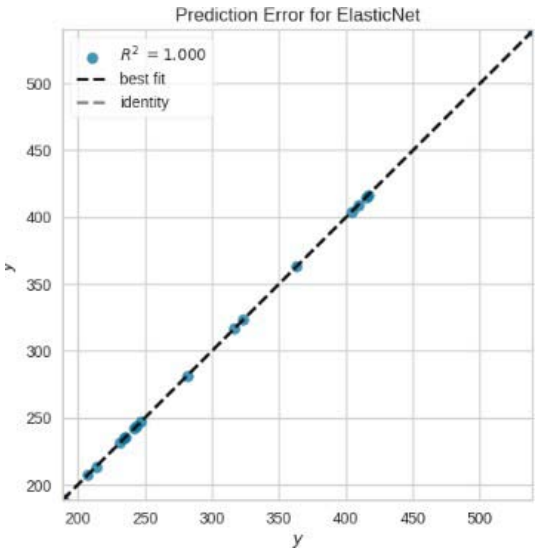
	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE	TT (Sec)
lr	Linear Regression	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000	0.2410
lasso	Lasso Regression	0.1612	0.0599	0.2003	1.0000	0.0006	0.0005	0.0320
ridge	Ridge Regression	0.0101	0.0005	0.0140	1.0000	0.0000	0.0000	0.0300
llar	Lasso Least Angle Regression	0.1613	0.0589	0.1998	1.0000	0.0006	0.0005	0.0310
br	Bayesian Ridge	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000	0.0320
en	Elastic Net	0.2196	0.2108	0.3036	0.9998	0.0009	0.0007	0.0320
huber	Huber Regressor	13.1549	601.3146	18.2020	0.5207	0.0559	0.0409	0.0490
et	Extra Trees Regressor	18.9690	891.9745	24.7828	0.3170	0.0823	0.0657	0.1810
gbr	Gradient Boosting Regressor	19.8730	842.0865	25.1282	0.1901	0.0837	0.0684	0.0820
xgboost	Extreme Gradient Boosting	24.2394	1507.4219	32.4543	-0.0233	0.1033	0.0783	0.0520
rf	Random Forest Regressor	26.4847	1590.3080	34.5563	-0.3531	0.1075	0.0875	0.3490
ada	AdaBoost Regressor	27.1256	1507.4559	34.1499	-0.7393	0.1136	0.0940	0.0830
omp	Orthogonal Matching Pursuit	52.3762	4421.5898	59.9154	-1.5530	0.1948	0.1762	0.0300
dt	Decision Tree Regressor	36.5792	2991.5417	46.9773	-1.5823	0.1623	0.1238	0.0340
lar	Least Angle Regression	10.1883	472.8763	12.8457	-2.3624	0.0447	0.0380	0.0330
knn	K Neighbors Regressor	43.2565	4332.4947	58.0248	-2.4464	0.1783	0.1385	0.0310
lightgbm	Light Gradient Boosting Machine	61.1941	8041.0637	78.2081	-6.5747	0.2508	0.2113	0.0910
dummy	Dummy Regressor	61.1941	8041.0636	78.2081	-6.5747	0.2508	0.2113	0.0350
par	Passive Aggressive Regressor	125.4990	34680.1443	143.2859	-94.0216	0.3445	0.3983	0.0310

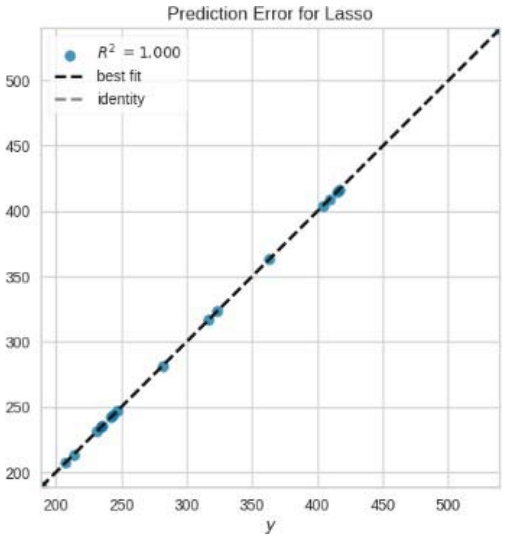
Figure 10: Comparing Models from Eighteen Regressors

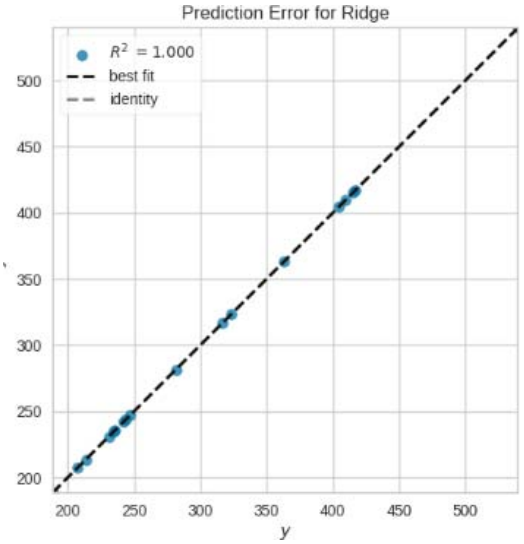
The best model according to PyCaret comparison is the linear regression model. This is consistent with our earlier analysis where the linear regression model had an R^2 score of 1.0. However, as mentioned earlier, a perfect score is suspicious and could be a sign of overfitting, where the model has learned the training data too well and may not perform wellon new, unseen data. It is important to note that PyCaret *compare_models* function uses cross-validation to evaluate the models, which provides a more robust assessment of the model's performance. Therefore, the linear regression model's perfect score in this case is less likely to be due to overfitting. The linear regression model seems to be the best model for predicting the weekly order volume based on the given features.

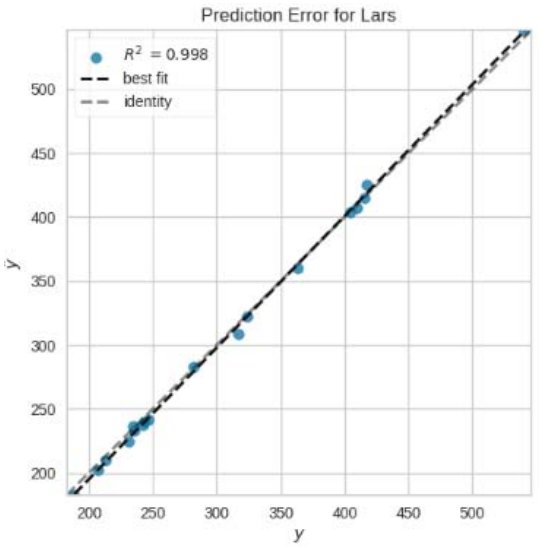


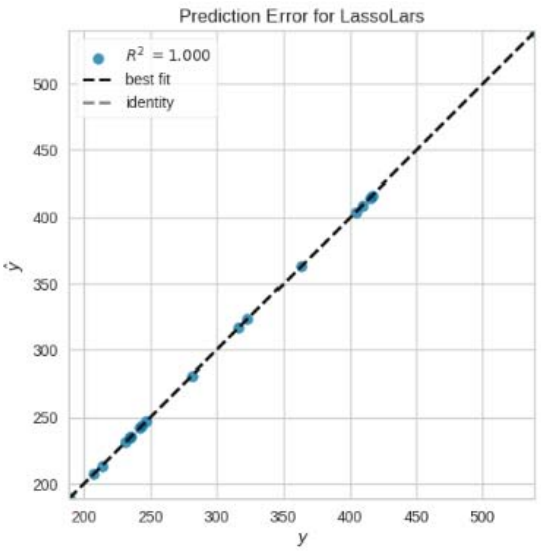


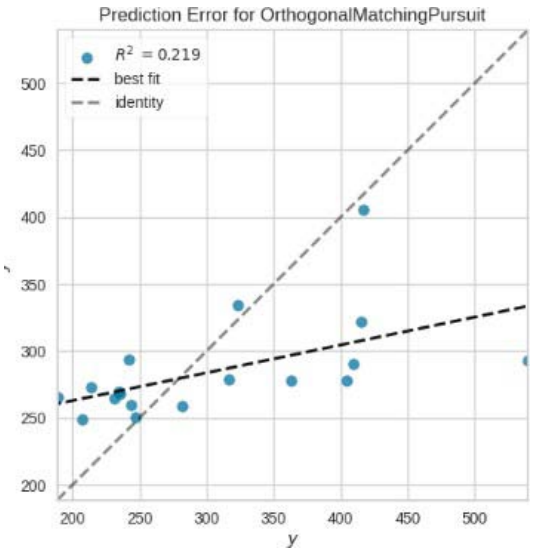


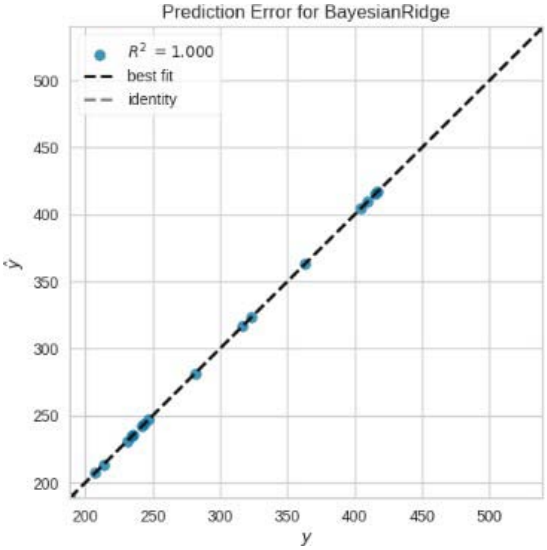


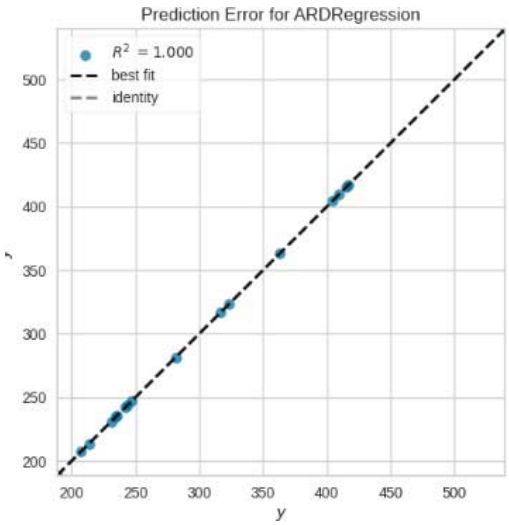


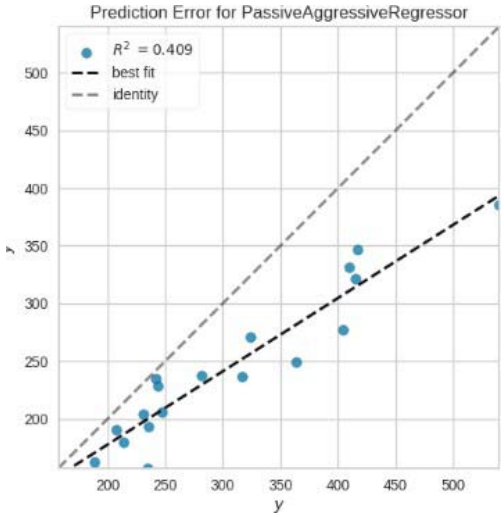


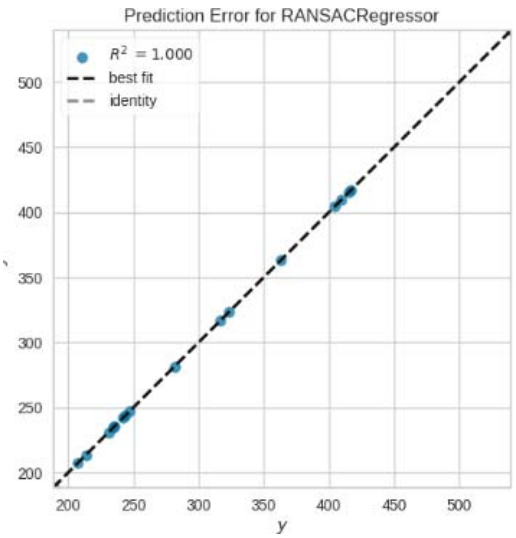


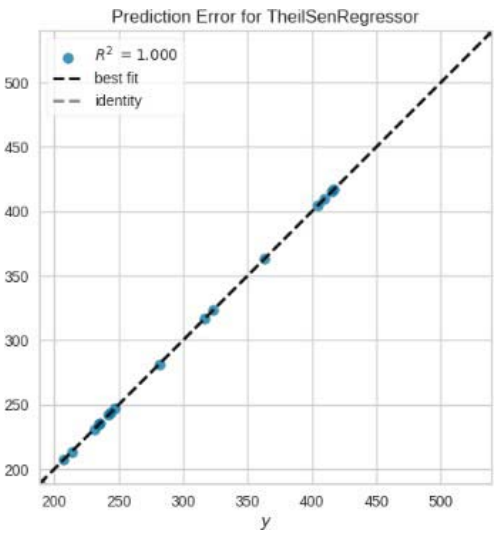


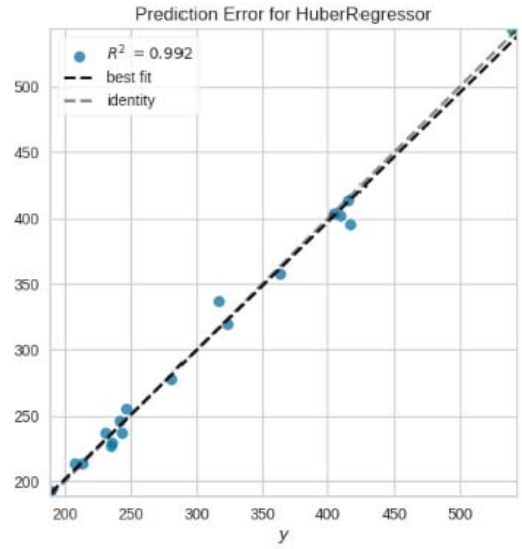


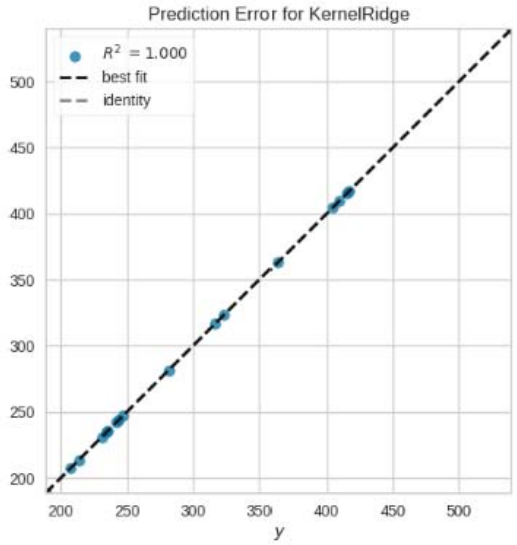


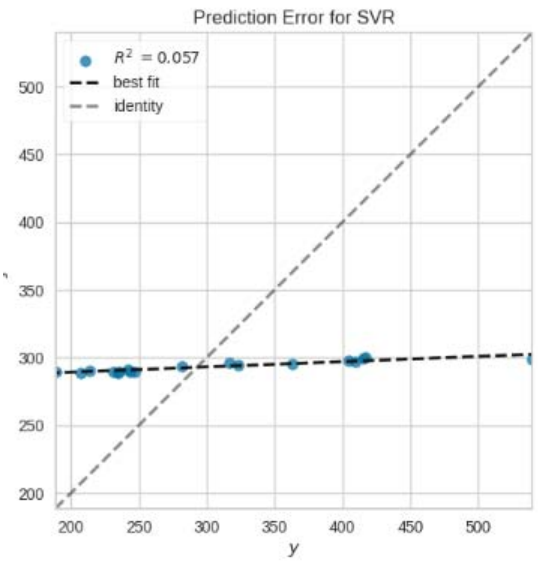


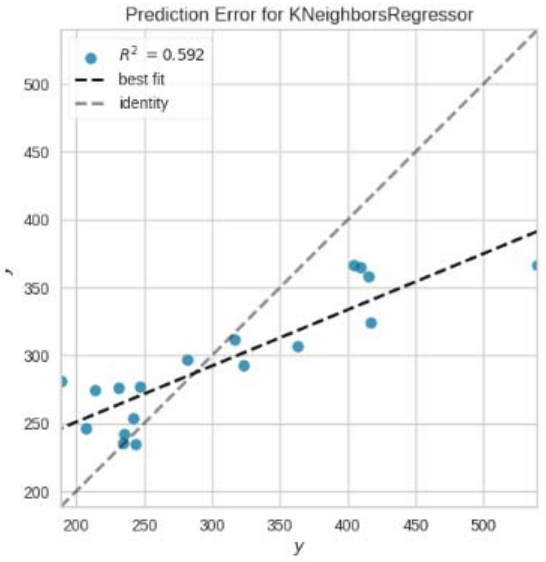


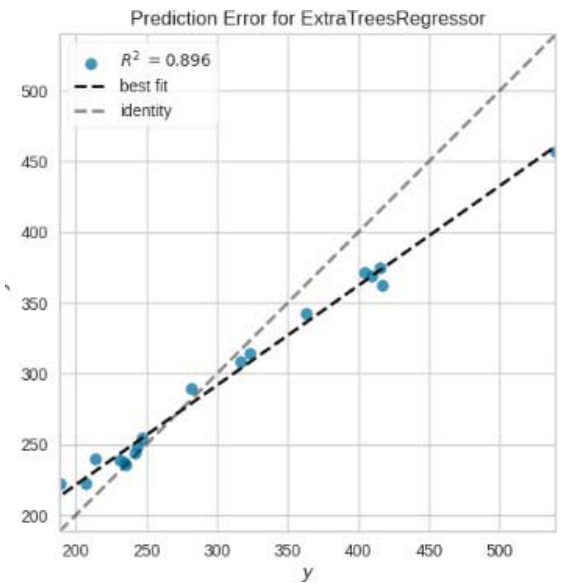


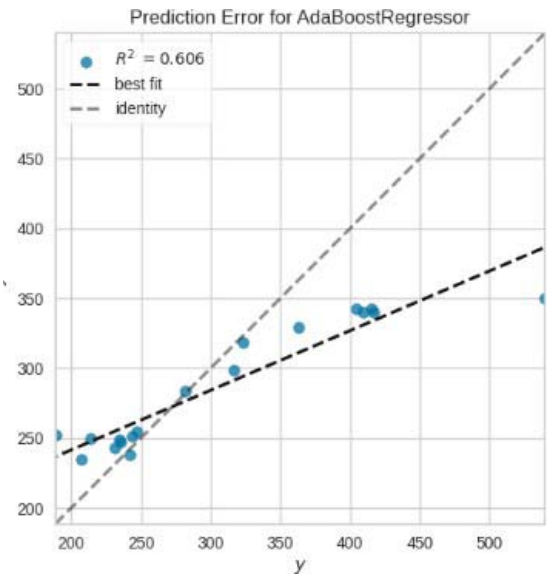












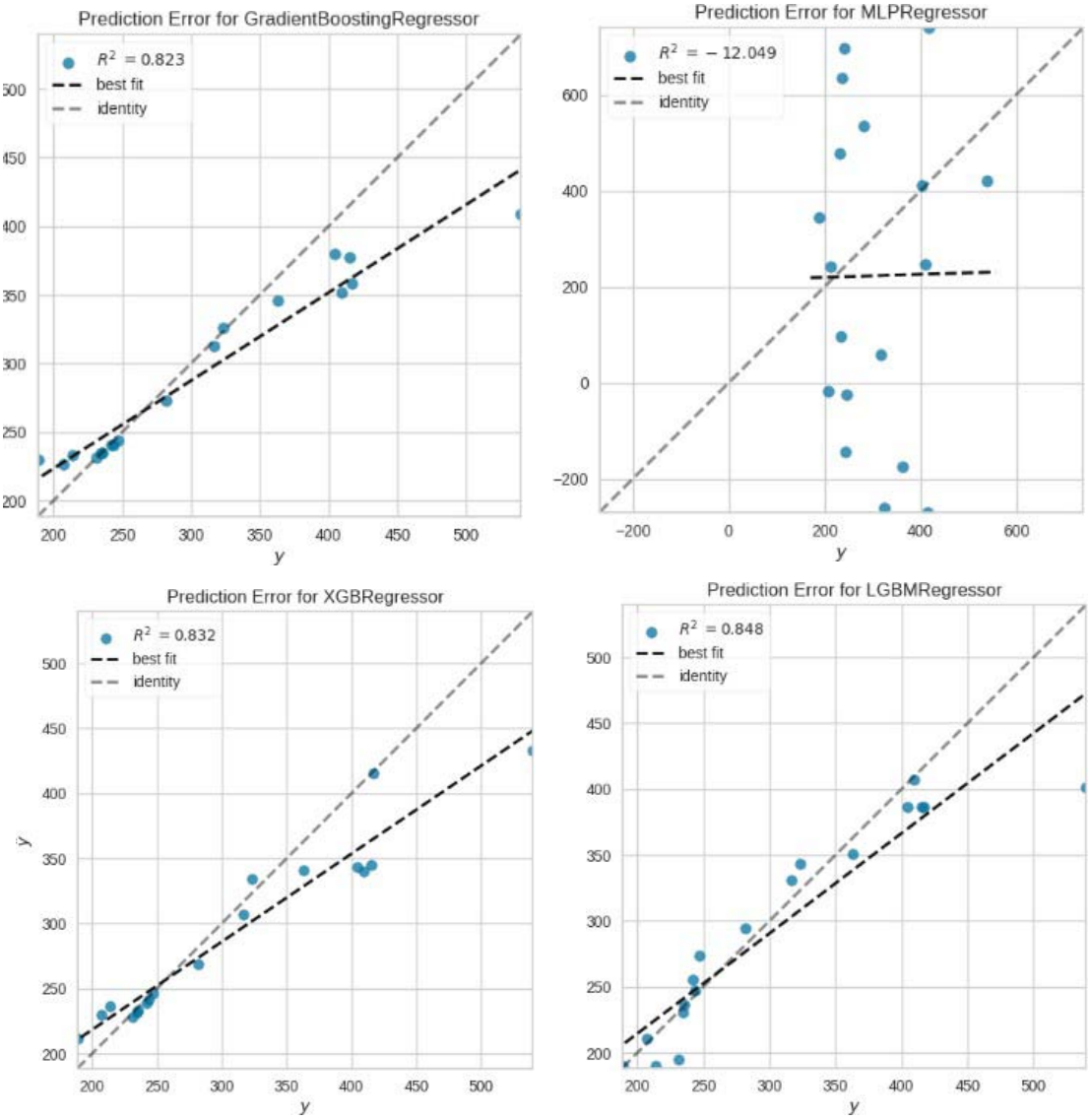


Figure 11: Prediction Plots

The feature importance plot shows the importance of each feature in predicting the target variable in the linear regression model.

```
# Print the best model
print(best_model)

LinearRegression(n_jobs=-1)

# Plotting feature importance
plot_model(best_model, plot = 'feature')
```

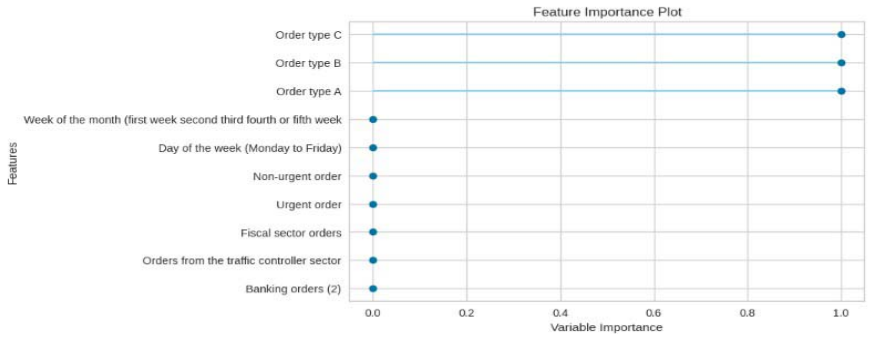


Figure 12: Feature Importance Plot

From the plot, we can see that the most important features for predicting the total orders are:

1. Orders from the traffic controller sector (the most important feature, with the highest importance score).
2. Order Type C: This is the second most important feature.
3. Non-Urgent Order: This is the third most important feature.
4. Order Type B: This is the fourth most important feature.
5. Urgent Order: This is the fifth most important feature.

The other features have relatively lower importance scores and they can be classified as less relevant in the study analysis.

IV. Results and Discussion

From figure 10, we picked the first six models with smaller MAE and better predictions for discussion. The rest of the models from Huber Regressor to Passive Aggressive Regressor have high value of MAE and their predictions are less accurate.

a) Linear Regression (LR)

Table 1: Performance Parameters for Linear Regression Model

Model Performance Parameters	Value
MAE	0.0000
MSE	0.0000
RMSE	0.0000
R^2	1.0000
RMSLE	0.0000
MAPE	0.0000
TT (Training Time)	0.5810 seconds

The linear regression model achieves perfect performance with MAE, MSE, RMSE, and RMSLE values of 0.0000, indicating that it perfectly predicts the target variable without any error. The R^2 value of 1.0000 suggests that the model explains all the variance in the data.

Additionally, the MAPE value of 0.0000 indicates that the average percentage error is zero. The training time (TT) is 0.5810 seconds, which is the time taken to train the model.

b) Lasso Regression (LASSO)

Table 2: Performance Parameters for LASSO Model

Model Performance Parameters	Value
MAE	0.1612
MSE	0.0599
RMSE	0.2003
R^2	1.0000
RMSLE	0.0006
MAPE	0.0005
TT (Training Time)	0.0890 seconds

The Lasso regression model achieves good performance with a relatively low MAE of 0.1612. The MSE and RMSE values indicate the average squared difference and square root of the average squared difference between predictions and actual values, respectively. The R^2value of 1.0000 suggests that the model explains all the variance in the data. The RMSLE value of 0.0006 indicates a small average logarithmic error. The MAPE value of 0.0005 suggests a small average percentage error. The training time is 0.0890 seconds.

c) *Ridge Regression (Ridge)*

Table 3: Performance Parameters for Ridge Regression Model

Model Performance Parameters	Value
MAE	0.0101
MSE	0.0005
RMSE	0.0140
R^2	1.0000
RMSLE	0.0000
MAPE	0.0000
TT (Training Time)	0.0810 seconds

The Ridge regression model performs very well with a low MAE of 0.0101, indicating a small average absolute difference between predictions and actual values. The MSE and RMSE values are also low, indicating a small average squared difference and square root of the average squared difference. The R2 value of 1.0000 suggests that the model explains all the variance in the data. The RMSLE and MAPE values are close to zero, indicating low logarithmic and percentage errors. The training time is 0.0810 seconds.

d) *Lasso Least Angle Regression (LLAR)*

Table 4: Performance Parameters for LASSO Least Angle Regression Model

Model Performance Parameters	Value
MAE	0.1613
MSE	0.0589
RMSE	0.1998
R^2	1.0000
RMSLE	0.0006
MAPE	0.0005
TT (Training Time)	0.0460 seconds

The LLAR model performs similarly to the Lasso regression model, with a slightly higher MAE of 0.1613. The MSE, RMSE, R^2, RMSLE, and MAPE values are also comparable. The training time is 0.0460 seconds, which is relatively low.

e) *Bayesian Ridge (BR)*

Table 5: Performance Parameters for Bayesian Ridge Regression Model

Model Performance Parameters	Value
MAE	0.0000
MSE	0.0000
RMSE	0.0000
R^2	1.0000
RMSLE	0.0000
MAPE	0.0000
TT (Training Time)	0.0440 seconds

The Bayesian Ridge model achieves perfect performance, similar to the Linear Regressionmodel. The MAE, MSE, RMSE, R2, RMSLE, and MAPE values are all zero, indicating perfect predictions. The training time is 0.0440 seconds.

f) Elastic Net (EN)

Table 6: Performance Parameters for Elastic Net Regression Model

Model Performance Parameters	Value
MAE	0.2196
MSE	0.2108
RMSE	0.3036
R^2	0.9998
RMSLE	0.0009
MAPE	0.0007
TT (Training Time)	0.0880 seconds

The Elastic Net model performs relatively worse compared to the previous models, with a higher MAE of 0.2196. The MSE and RMSE values indicate larger squared differences and square roots of the squared differences, respectively. The R^2 value of 0.9998 suggests that themodel explains most of the variance in the data. The RMSLE and MAPE values indicate small logarithmic and percentage errors. The training time is 0.0880 seconds.

Overall, the Linear Regression, Bayesian Ridge, and Ridge Regression models achieve perfect performance with zero errors. The Lasso Regression and Lasso Least Angle Regression models perform well with low MAE values. The Elastic Net model performs relatively worse in terms of MAE but still achieves a high R^2 value. The training times vary among the models but are generally quite fast, with the longest being 0.5810 seconds for theLinear Regression model.

V. Conclusion and Recommendations

This study illustrates a paradigm shift from traditional demand forecasting approaches to more sophisticated statistical methods and machine learning techniques. In line with these advancements, this study aims to improve existing knowledge on demand forecasting by utilizing an advanced Python library that simultaneously generates predictions using multiple regression algorithms. Additionally, the evaluation of the models' performance included various error analysis metrics such as MAE, MSE, RMSE, RMSLE, and MAPE. Among the models analysed, the first six models with smaller MAE and better predictions were selected for discussion. The remaining models, starting from Huber Regressor to Passive Aggressive Regressor, exhibited higher MAE values and less accurate predictions. The Linear Regression, Bayesian Ridge, and Ridge Regression models achieved perfect performance with zero errors across all performance metrics. The Lasso Regression and Lasso Least Angle Regression models demonstrated good performance with low MAE values. The Elastic Net model performed relatively worse in terms of MAE but still exhibited a high R^2 value, indicating a good level of explained variance. The training times varied among the models but were generally fast, with the longest training time observed for the Linear Regression model. This study builds upon previous research by employing advanced Python libraries and evaluating many regression algorithms simultaneously to improve demand forecasting. The selected models demonstrated varying levels of accuracy and computational efficiency. The findings highlight the effectiveness of non-linear techniques, such as Lasso Regression and Bayesian Ridge, in achieving accurate predictions with low errors. These results contribute to the growing body of knowledge on demand forecasting in supply chain planning and can aid decision-makers in improving their demand forecasting processes.

It is very important to know the study dataset was small and there is propensity for few models to produce R^2 of 1.0000. The case will be different where we have voluminous dataset. Perhaps, based on the analysis and results obtained in this study, several recommendations can be made to further enhance the demand forecasting process in supply chain planning:

1. Explore ensemble methods: Ensemble methods, such as Adaboost and the ensemble of Random Forest, have shown promising results in previous studies. It would be beneficial to investigate the potential of combining multiple regression algorithms to create an ensemble model that leverages the strengths of different approaches and produces even more accurate predictions.
2. Other machine learning techniques: While this study focused on regression algorithms, there are other advanced machine learning techniques that can be explored for demand forecasting. For example, deep learning models like recurrent neural networks (RNN) and long short-term memory (LSTM) networkshave been successful in capturing complex patterns in time series data. Investigating the application of these techniques may lead to improved forecastingaccuracy.

References Références Referencias

1. Al-Jarrah OY et al (2015) Efficient machine learning for big data: a review. Big Data Res 2 (3): 87–93.
2. Carbonneau, R., Laframboise, K., and Vahidor, R. (2008) European Journal of Operational Research 184 pp 1140–1154.
3. Casson, M. (2013) Economic analysis of international supply chains: An internalization perspective. Journal of Supply Chain Management, 49 (2), 8-13.
4. Chopra, S. and Meindil, P. (2011) Supply Chain Management, 4th ed., Dorling Kindersley Pvt. Ltd.
5. Ketchen, D. J., and Giunipero, L. C. (2004) the intersection of strategic management and supply chain management. Industrial Marketing Management, 33 (1), pp 51-56.
6. Hope, J. and Fraser, R. (2003) beyond budgeting: how managers can break free from the annual performance trap. Harvard Business Press.
7. Wang, G. (2012) Procedia Engineering 29, pp 280 – 284.