PRODUCT DEMAND PREDICTION

By Machine Learning Algorithms

2.1 PROBLEM DESCRIPTION

The problem is to create a machine learning model that forecasts product demand based on historical sales data and external factors. The goal is to help businesses optimise inventory management and production planning to efficiently meet customer needs. This project involves data collection, data pre-processing, feature engineering, model selection, training, and evaluation.

2.2 DATASET INFORMATION

Data Collection: Historical sales data

Dataset Link: https://www.kaggle.com/datasets/chakradharmattapalli/product-demand-prediction-with-machine-learning

Data Pre-processing: Clean and pre-process the data, handle missing values, and convert categorical features into numerical representations

Feature Engineering: Create additional features that capture seasonal patterns, trends, and external influences on product demand.

Model Selection: Choose suitable regression algorithms (e.g., Decision Tree, Random Forest, Gradient Boost) for demand forecasting.

Model Training: Train the selected model using the pre-processed data.

Evaluation: Evaluate the model's performance using appropriate regression metrics (e.g., Mean Absolute Error, Root Mean Squared Error,R-Squared).

2.3 DATASET COLUMNS

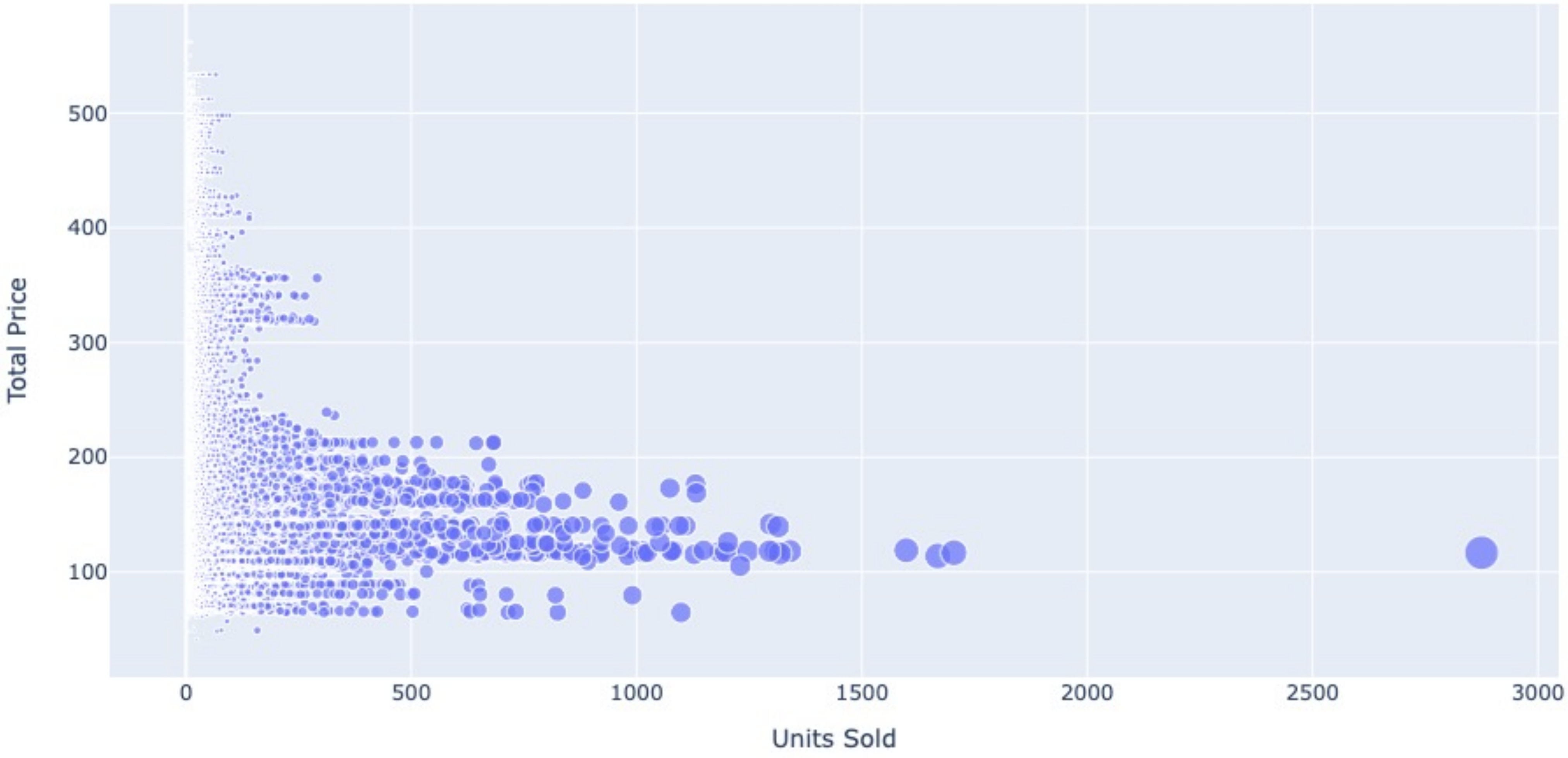
Column Names

1. Product ID
2. Store ID
3. Total Price
4. Base Price
5. Units Sold (Quantity Demanded)

Training Data:

1. **Independent Variables (Features):**
   - Total Price: The total price of a product.
   - Base Price: The base price of the product.
2. **Dependent Variable (Target):**
   - Units Sold: The number of units of the product sold.

PLOT OF TOTAL PRICE VS UNITS SOLD



2.4 MODULES/ LIBRARIES USED:

Libraries

- Pandas
- Numpy
- Seaborn
- Matplotlib
- sklearn.model_selection
- sklearn.ensemble
- sklearn.metrics

MODELS USED

Decision Tree Regressor: A scikit-learn regressor implementing a decision tree for regression tasks, predicting target values based on input features.

Gradient Boosting Regressor: A scikit-learn regressor employing gradient boosting for ensemble learning, combining weak learners to improve predictive performance.

Random Forest Regressor: A scikit-learn regressor utilizing a random forest, an ensemble of decision trees, to enhance accuracy and control overfitting in regression tasks.

2.5 TRAIN AND TEST

train_test_split function:

The train_test_split function splits arrays or matrices into random train and test subsets. It takes several parameters, including your features (X) and labels (y), and splits them into four subsets: X_train, X_test, y_train, and y_test.

```
from sklearn.model_selection import train_test_split

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

Model initialization: The machine learning model (Random Forest Regressor in this case) is initialized with specified hyperparameters

TRAIN AND TEST

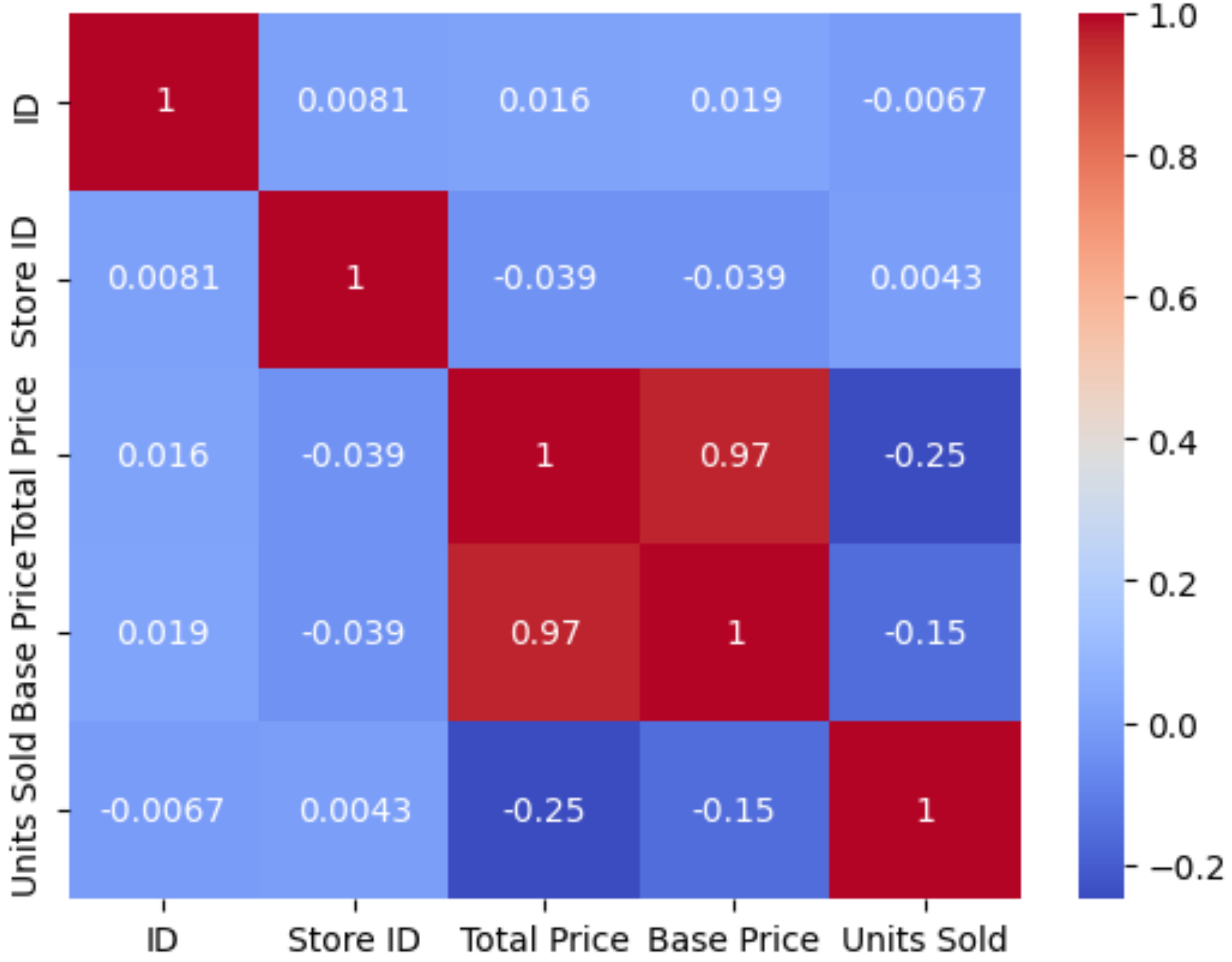
Training the Model: The fit function is used to train the model using the training data (X_train and y_train).

```
#Model Creation
# Create and train a Gradient Boosting Regressor model
model = GradientBoostingRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
```

Making Predictions: The trained model is used to make predictions on the test data (X_test). The resulting predictions are stored in the predictions variable.

```
# Predict a test data with a trained model
y_pred = model.predict(X_test)
```

DATA CORRELATION HEAT MAP



INTERPRETATIONS:

Total Price and Base Price (Correlation: 0.958885) : Total Price and Base Price have a very strong positive correlation. This means that there is a strong linear relationship between the total price and the base price of the products. As the base price increases, the total price tends to increase accordingly.

Total Price and Units Sold (Correlation: -0.235625) : Total Price and Units Sold have a moderate negative correlation. This suggests that as the total price of the products increases, the number of units sold tends to decrease. Customers may buy fewer units when the price is higher.

Base Price and Units Sold (Correlation: -0.140022) : Base Price and Units Sold also have a moderate negative correlation. This indicates that as the base price of the products increases, the number of units sold tends to decrease. This relationship is similar to the one observed with total price.

WHY DOES THE LINEAR REGRESSION MODEL FAIL?

By observing the correlation matrix, high correlation (0.96) among the independent variables and a relatively weak correlation (-0.26) between the dependent and independent variables, linear regression might not be the best choice for modelling this relationship effectively.

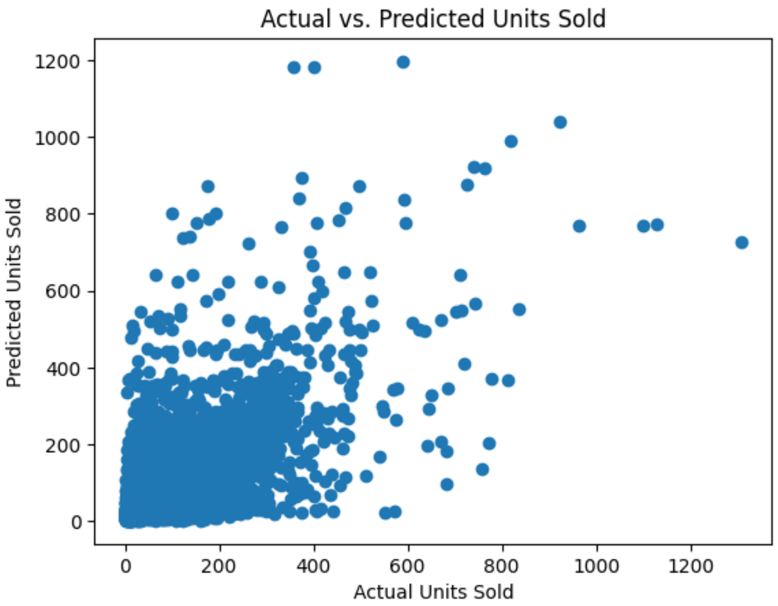
Because:

1. Multicollinearity:
2. Weak Correlation with the Dependent Variable
3. Overfitting

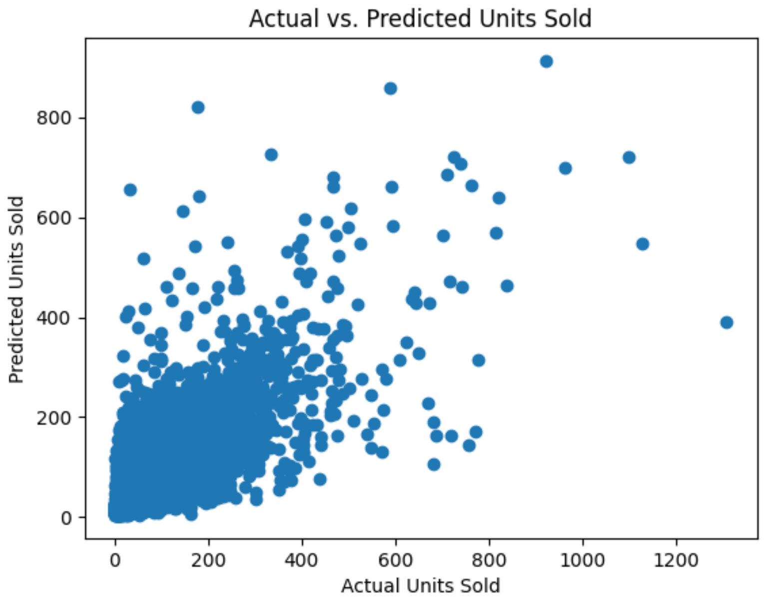
```
R2: 0.1490719889302594
MAE: 32.49815502614977
MSE: 2784.619420940248
RMSE: 52.76949327916886
```

Linear Regression Metrics

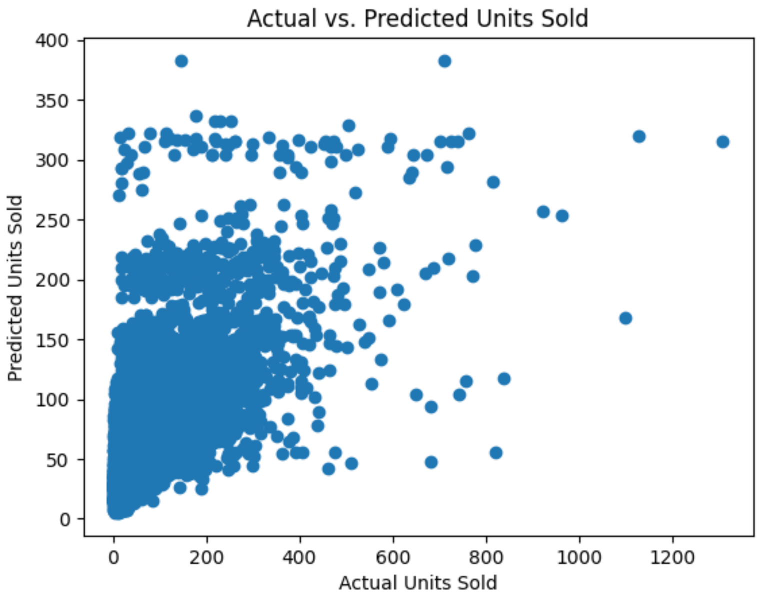
ACTUAL VS PREDICTED UNITS SOLD



Decision Tree Regressor



Random Forest Regressor



Gradient Boosting Regressor

2.7 METRICS:

- R-Squared(R^2)
- Mean Absolute Error (MAE)
- Mean Squared Error (MSE)
- Root Mean Squared Error (RMSE)

	Decision Tree	Random Forest	Gradient Boosting Regressor
R-squared	0.46	0.61	0.43
Mean Absolute Error (MAE)	20.55	18.93	25.56
Mean Squared Error (MSE)	1761.15	1264.28	1873.94
Root Mean Squared Error (RMSE)	41.97	35.56	43.29

2.7 Metrics:

```
R2: 0.46182404986623227
MAE: 20.54729114753958
MSE: 1761.15392034337
RMSE: 41.96610442182321
```

Decision Tree Regressor

```
R2: 0.6136579174937777
MAE: 18.93038515316013
MSE: 1264.28517110497 9
RMSE: 35.5567879750826 1
```

Random Forest Regressor

```
R2: 0.4273579061461198
MAE: 25.55987637616166 8
MSE: 1873.9426544306266
RMSE: 43.28905929251208
```

Gradient Boosting Regressor